PhD Dissertation

International Doctorate School in Information and Communication Technologies

DIT - University of Trento

Fractional Lambda Switching
- Node Design and Time-blocking Analysis -

Viet-Thang Nguyen

Advisors: Prof. Renato LoCigno and Prof. Yoram Ofek

Università degli Studi di Trento

March 2007

Abstract

Streaming and real-time traffics are dramatically booming on the Internet. Innovative and successful business models (such as Skype, Youtube, IPTV and the forth coming Joost project) may bring new dimensions into revenues of dot-com and telecommunication industry; however, they also introduce a threat to ISPs and carriers in terms of scaling their networks so that they can handle traffic more efficiently and utilize the bandwidth provided by the DWDM transmission technology. Moreover, to ensure that customers who pay more get more priority for their traffic, QoS guarantees are vital parts of bandwidth management. When more real-time traffic is injecting in networks, handling bursty traffic and jitter is increasingly important. However, as pointed out in several researches, when 35% of link capacity is streaming traffic, current QoS Internet mechanisms start degrading network utilization. Since current QoS mechanisms do not fully address these issues, there is a need for new approaches to solve these problems.

The thesis focuses on important aspects of Time Driven Switching (TDS)-Fractional Lambda Switching (FλS). TDS-FλS architecture provides guaranteed QoS, and it is highly scalable and compatible to the current Internet architecture. In TDS-FλS, a common time reference (CTR) and pipeline forwarding are used to deliver traffic flows end-to-end. Thus, the header processing overhead is eliminated and no or few buffers (for enabling scheduling delay) at switching nodes are required. TDS-FλS is a novel technology offering a networking paradigm with no congestion, no jitter, no packet loss, but many open issues still need investigations.

In the first part of the thesis, we discuss the use of tunable laser and propose some related architectures to realize all-optical switches that enable

sub-wavelength granularity switching. The design objective is minimizing node's hardware complexity while maximizing node's scheduling flexibility so that a designed node has a low or null space blocking. Notably, we prove that one design is strictly non-space blocking with hardware complexity (in terms of counting switching elements) equivalent to that of a Clos network, known to be the minimal complexity non blocking architecture.

The remaining part of the theoretical work of this thesis is dedicated to the analysis of the blocking performance. Thought it has been conjectured that TDS-FλSs yield low blocking under high load conditions, no formal proof has been produced so far. The objective of this research track is a comprehensive blocking analysis under various contexts and dependent parameters such as load, hop-length, possible scheduling delay, time-cycle size, etc. As the time dimension is critical in TDS-FλS, blocking in time-domain (or time-blocking) is a subject of interest. Since a switch can have a space-blocking fabric, both space and time blocking should be analyzed jointly. However, the twist of space and time blocking makes it extremely complex to analyze. We present in this work the case where switches are strictly non-space blocking so that we need to tackle only time-blocking.

We start presenting a complete analysis of time-blocking probability of a stand-alone strictly non-space blocking switch under all possible scheduling delay schemes under given load assumptions. This initial analysis helps to obtain some fundamental combinatorial observations and results that are later used to study the time-blocking probability of a multi connected switches.

When a number of strictly non-space blocking switches are connected and under zero scheduling delay scheme, it is almost straightforward to derive a closed form formula of blocking probability. On the other hand, for non-zero scheduling delays, the exact solution is possible based on the stationary solution of the Markov chain. However, for large systems, the

computation is impractical because of extreme complexity. Using some relaxations, we successfully derive upper and lower bounds, which can be used to capture a closed approximation of time-blocking probability. To evaluate the approximation, we compare various numerical results and simulations. Since in FλSs, a link comprises multiple optical channels, we extend the analysis for this case.

Finally, a TDS prototype and its FPGA-based controller is introduced. The prototype implemented from off-the-self components confirms the practical aspect of TDS technology and its properties in terms of being very scalable and offering loss-jitter-congestion-free networks.

Keywords

[time-driven switching, fractional lambda switching, tunable laser, optical networks, optical networking, blocking analysis, combinatorics]

Acknowledgement

I express my sincere esteems to my two advisors, Prof. Renato LoCigno and Prof. Yoram Ofek, from whom I always find excellent advices and discussions during my research work. There have been all interesting moments whenever having open and useful discussions with either of the two advisors, or both at the same time. I have been encouraged and pushed, questioned and answered at right times. I have been taught, guided thoroughly through thick and thin.

A special thank I would like to send to Prof. Telek Miklos from Budapest-Hungary. Though it is not a long since we met, I find out very interesting to discuss with him about all things rather than just only research work. He also appeared at the right time and provided me some very useful hints that help me to finish this thesis.

Prof. Mario Baldi would be another one I wish to thank and spend more time working with him. He and Prof. Ofek are the first to suggest the investigation the use of tunable laser into designing work.

I wish to thank Prof. H.Q. Ngo (SUNY Buffalo). Though we have not met, some few exchanged emails about combinatorics had helped me somehow.

I do also want to acknowledge the great time spending with various colleagues (G. Fontana, M. Corra, G. Marchetto, T.H. Truong, O. Zadedyurina, D. Agrawal) while working for IP-FLOW project. It has been a time where I experienced various working and life aspects. Without them, there would not have been a successfully experimental results of the test-bed partially introduced in this work.

My thanks also go to DIT secretaries, colleagues of the NetMob group (E. Salvadori, R. Cassella, D. Carra, D. Severina and others) and various

friends in Open Spaces at DIT, who have been always willing to help me whenever I had problems staying in Trento.

Special blessing goes to my long-year roomies Dawa Dorje and Son M.Dao. Trento is small-but-charming city where we have enjoyed a lot during many excursions, but it would have been much less fun to spend time without you fellows.

This work is dedicated to my family and beloved ones.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Bandwidth-hungry applications are growing with the faster pace than ever. These are applications ranging from academic and academic purposes to recreational ones. Scientists seeks fast communication, exchange, back-up, and distribution of huge amount of raw data gathering from biology sciences, to astronomical events, to nuclear tests, to grid computing. Citizens seek out multiple media centers for their entertainments. Recreational applications are abundant and streaming-oriented: high quality streaming music and clips; online and interactive video game; live streaming sport events; video contents. Looking back through the years, we can clearly see that online streaming is becoming more widespread than ever.

Recreational applications alone create scaling problems to MAN/WAN networks. The problems of MAN/WAN networking do not reside in the network transmission technology as abundant dark fibers are available, and each fiber can carry up to multiple $Tbit/s$ of traffic. On the one hand, the problems somehow still stay in the last-mile (or first-mile) networking as most access technologies are lag behind the optical ones in terms of speed and available bandwidth. On the other hand, the problem lies in switching and routing architectures than can not be scaled to meet the growth of bandwidth demand.

The picture gets even worse as QoS traffics are also exploding. To ensure that customers who pay more get more priority for their traffic, a strict QoS guarantee must be a vital part of bandwidth management. When more real-time traffic is injected in networks, handling bursty traffic and jitter is increasingly important. However, as pointed out in several researches, when 35% of link capacity is streaming traffic, current QoS Internet mechanisms start degrading network utilization. Since current QoS mechanisms do not fully address these issues, new approaches are needed to solve these problems.

Specifically, current switching and routing architectures are about to reach the limit because of heavy overhead remaining in systems. The problem is more severe if multiple streams are mixed and thus complex queueing mechanisms must be set up to handle various levels of priority. Moreover, at optical layer where the DWDM technology mature, sub-wavelength switching is still not ready, thus bringing more burdens to packet switching nodes. This is because various end users may have to share one optical channel (a wavelength) in the sense that their traffic must be statically multiplexed/demultiplexed at ingress/egress nodes, and must be handled by switching nodes in core networks.

This thesis aims at presenting some research contributions for a promising sub-wavelength technology, which utilizing time helps removing the overhead of header processing. The technology is called time-driven switching (TDS) in general. In the optical domain, the technology is known as Fractional Lambda Switching (FλS). FλS is suitable for both MAN and WAN networks.

1.1 Problems, solutions and thesis outlines

Sub-wavelength switching is not a new concept in the area of optical networking. For examples, optical burst switching (OBS) [51, 69], optical packet switching (OPS) [62] and some others time slot interchange (TSI) [8, 9, 12, 15, 25, 28, 29, 32, 56, 71, 64, 65, 77] architectures have been proposed in the past years. However, as will be discussed in Chapter 2, these architectures still lacks of some important points that hinder them to be deployed into real networks. A detail introduction to TDS-FλS is also presented in Chapter 2.

Chapter 3 of the thesis focuses on proposing some node designs for FλS. The key point is that tunable lasers are used to allow easy and fast wavelength swapping. The combination of swapping wavelength in space domain and switching in time provides specific characteristics of FλS.

Since TDS-FλS uses time as a main criteria for guiding traffics end-to-end, analyzing blocking performance in time-domain is a mandatory issues. Chapters 4 and 5 discuss and derive results on this challenging issue. In Chapter 4, the problem of time-blocking probability in TDS switches is formulated and analyzed. The main result of Chapter 4 is the time-blocking probability analysis for a stand-alone switch as a function of the number of possible scheduling delay z and the loads (K, b_i) and (K, b_o). In addition, the thorough combinatorial analysis allows some fundamental observations and results that are later used to study the time-blocking probability of multi connected switches (i.e., multi-hop) in Chapter 5.

In Chapter 5, a closed formula of time-blocking probability for zero scheduling delay is presented. On the other hand, for nonzero scheduling delay, we explain that an exact solution requires a stationary solution of Markov chain, which is feasible only for a small number of time-frames per time-cycle. For large numbers, we present the upper and lower bound analysis. The average of the two bounds finally yields a very good approximation of time-blocking probability.

Proving that TDS-FλS technology is feasible and can be really deployed in future networks is the last topic of this thesis. We report in Chapter 6 the successful experiment of the first prototype TDS switching node using off-the-shelf components. While building a prototype requires an integration of various lab works, this chapter focuses on the implementation of a FPGA-based controller - the brain of the switching node.

Chapter 7 concludes and discusses the thesis and the 3-year research work. We also highlight some extensions for the future of FλS networking and researches.

Finally, the appendices delivers proofs of some complex formulas presented in the thesis.

Chapter 2

Sub-wavelength switching

In this chapter, we first introduce to some key optical components and devices, such as tunable lasers, wavelength converters, switching elements. These are used in designing switch nodes developed later in this work. Besides, a bunch of other optical components such as multiplexing (MUX), de-multiplexing (DeMUX), star coupler, splitter and combiner, etc should be discussed as well. However, they are matured and commercial products are well set. References to these devices are flourish. Thus we omit to discuss them in detail in this work.

Next, we present a detail introduction to fractional lambda switching (FλS), which utilizes global time Universal Time Coordinated (UTC) for network synchronization. FλS is a novel proposal for the management of optical networks with sub-wavelength granularity. FλS reduces the complexity of switching and eliminates the need for header processing, which is a major open problem in realizing all-optical networks. Consequently, the dynamic all-optical networking with FλS is viable with state of the art optical components.

Finally, the chapter ends up with brief reviews of other related works for sub-wavelength switching in optical networks. Those are OBS, OPS, and several time-slot wavelength techniques.

2.1 Optical components and devices

2.1.1 Tunable transceiver

Transceivers play vital roles in optical networks. In the past, when tunability was not possible due to immature technology, fixed transceivers were used. However, recently, advanced technologies in optics industry allow transceivers to tune their working wavelengths in dynamic ranges aligned with international telecommunication union (ITU)-T grids [1]. The deployment of tunable transceivers helps to significantly reduce operation costs of networks, and to boost the deployment of a dynamic configuration networks. For example, instead of placing multiple fixed transceivers for backing up multiple working wavelengths, it may be possible to use only a single tunable transceiver with its tuning range covering all working wavelengths, thus, saving cost significantly. In reality, a tuning transceiver is an integrated device where its components are a tunable laser (for transmission) and a tunable filter (for receiving).

Fast tuning transceivers are desired for advancing to sub-wavelength switching networks such as FλS and later OPS, where tuning time is bounded by ms or ps. Pertaining to our switching node design for FλS networks, in this section, we aim at introducing promising techniques that allow fast tunability.

Tunable laser

Plainly, a tunable laser is a laser that can tune its transmitted wavelength in a bounded dynamic range through some simple control functions. A tunable laser has the basic structure as that of a fixed laser. Thus, tuning mechanisms can be categorized as following:

- Electrical tuning: the current is injected into the wavelength selective reflector of the laser to change the refractive index, resulting in the change of the peak wavelength for transmission. Electrical tuning allows the tuning speed of about few ns.

- Thermal tuning: the laser sample is heated to change the refractive index. The thermal response is about 0.1 nm/K for an InP telecom laser emitting around 1,550 nm. Due to the slow thermal response, the tuning speed of thermal tuning is in the ms time scale. Intel is known to support external-cavity laser (ECL) thermal tunable laser that tuning speed is 10 s [49].

- Mechanical tuning: the emitted wavelength is changed by mechanically changing the cavity length or the angle of incidence of the light to the reflector. The tuning speed is bounded in a scale of ms. Mechanical tuning is the best among three mechanisms in terms of large tuning range. However, the drawback of this mechanism is the complex fabrication and packaging due to micro-mechanical sensitivity.

Besides tuning speed, there are other specifications to compare performances of tunable lasers. These parameters are optical power, wavelength accuracy, relative intensity noise (RIN), side-mode suppression ratio (SMSR), power dispersion, tuning range, packaging issue, etc. Neglecting tuning mechanisms, there are various technologies to implement tunable lasers, such as monolithic distributed Bragg reflector (DBR), distributed feedback (DFB) array, and ECL [3, 43]. Excellent categorizations, reviews of these technologies and commercial products can be found in [14, 38, 59].

In [63], a widely tunable fast-switching laser (electrical tuning) module that accesses 64 ITU channels with 50 GHz spacing and switches wavelength in under 50 ns was reported. In [78], a less than 6.8 ns tuning time of a tunable laser based on adjusting different bias currents of Fabry-Perot (FP) lasers was presented. Though these fast tuning tunable lasers still face some limits such as unbalanced output power for different wave-

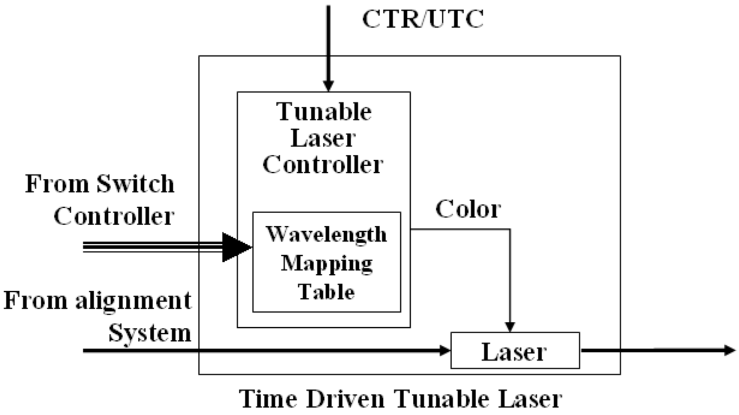


Figure 2.1: No bit-stream stopping with tunable laser controlled by UTC

lengths and narrow tuning range, high power dispersion, wavelength drifting, etc. Scientists are working on to overcome these issues. It is believed that wide and accurate tuning lasers will soon meet the same specifications of fixed-wavelength lasers.

The use of tunable lasers (see Fig. 2.1) in [79] is similar to that of FλS node designs in Chapter 3. There is no bit stream stopping in the design. A part of the incoming signal is tapped to drive and control the tunable laser. Another similar usage of tunable lasers in designing OPS nodes was reported in [36].

Tunable filter and tunable photodetector

As tunable lasers and wavelength converters, tunable filters are important for the deployment of all-optical networks in future. Tunable filters act as wavelength selectors at optical add-drop multiplexer (OADM) and egress nodes. Conventional techniques to implement tunable filters are based on FP and Mach-Zehnder (MZ) interferometers, acoustooptical tunable filter (AOTF) and fiber Bragg grating (FBG) [60].

In connection to sub-wavelength switching networks such as OBS, OPS and fractional lambda switching (FLS), fast tuning filters are strongly required. There have been rich researches in developing fast tuning filters, notably are digitally tunable optical filter (DTOF) techniques based on using thin film filter (TFF), torsional tunable filter (TTF) [42], semiconductor optical amplifier (SOA), arrayed waveguide grating (AWG), or the combination of them. A category of filtering techniques and their characteristics can be found in [22].

Besides tunable filters, tunable photodetectors are also desired to realize fast sub-wavelength switching at the receiver sides (i.e., egress nodes). Several researches recently exposed the implementation of fast and wide-range tunable photodetectors [16, 21, 70].

2.1.2 Wavelength converter

An all-optic wavelength converter is a device capable of mapping the information from a given incoming wavelength to a desired outgoing wavelength without stopping the bit-stream and without the E-O-E conversion. There are various implementations of wavelength converters including: optical gates comprised of a photodiode and electroabsorption modulator (EAM) [80]; cross-phase modulation (XPM) in SOA and fiber [61]; and cross-absorption modulation (XAM) of EAM [19, 30]; cross-gain modulation (XGM) [11]; four-wave mixing (FWM) in SOA or in semiconductor-fiber ring (SFR) [41].

Conversion technique using XGM has an advantage in terms of compactness since it can be implemented in a single SOA. However, it faces several problems such as low conversion speed (determined by gain recovery time), limited extinction ratio, relatively large spectral chirping and inverted coding operations. Meanwhile, wavelength converters using XPM in SOA have several advantages: very fast conversion speed, possible up/down conversions without degrading the extinction ratio. A brief review of these techniques can be found in [73].

Some of experiments show remarkable achievements, for example, the experiment in [80] showed a 25 nm dynamic conversion range with remarkable speed (in the order of ps), though the power penalty of this wavelength converter is high (up to 2.1 dB). However, like tunable lasers, wavelength conversion techniques still face some serious issues such as power penalty level, packaging, power consumption, stable and accurate tuned-wavelengths, etc. Usually, there are tradeoffs between various design requirements. Photonics physicists and manufacturers are carrying on extensive researches and efforts to realize commercial products for future optical networks.

Finally, we note that switching nodes proposed in Chapter 3 of this work use tunable lasers as one of a major components in their designs. However, once wavelength converter's industry matures, all tunable lasers should be replaced by wavelength converters to realize all-optical FλS.

2.1.3 Switching element

A simple all-optical switching element can be implemented based on SOA as following. An SOA gate is an array of devices monolithically integrated on the same substrate and used as a gate. When injected current in an SOA is high, it passes light through with some amplification. When injected current falls to near zero, it blocks the light. On the other words, the simplest method to control an SOA gate is by turning the device current ON or OFF. The great advantage of SOA gates is that they can be integrated to form gate arrays. Thus, an SOA array can act as a switching module.

The switching time of a SOA gate is of the order of 100 ps. For instance, in [24], the technique (called preimpulse step-injected current PISIC) was reported. By applying PISIC, ON/OFF switching time of bulk SOAs was reported to reduce to 0.2 ns, and can be further reduced to the order of tens of *ps*. In [37], the ultra-fast optical gate monolithically integrating a uni-traveling-camer photodiode and a traveling-wave electroabsorption modulator (TW-EAM) was reported, exhibiting a minimum gate opening time of 2.3 *ps* with an extinction ratio of 14 *dB* and 3.0 *ps* with 19 *dB*, respectively.

In general, the basic problem of all-optical switching element is not the switching speed but the packaging issue, since constructing a switching node may require thousands of switching elements. The integration of large amount of SOA-based switching elements into photonic integrated circuit (PIC) devices requires more researches and progress in photonics physic.

In the strictly non-space blocking design for FλS later proposed in this work, a number of active ON/OFF switching elements are required. It is also believed that such basic switching elements are essential components for designing OPS nodes in future all-optical networks.

2.2 Fractional lambda switching

Multi-wavelength optical networks have been widely deployed. Wavelength-routed networking [54] has been the subject of research for many years. However, the typical optical switching bandwidth granularity has been the entire optical channel – i.e., the whole lambda (λ). Consequently, with such design it is only possible to allocate the whole optical channel (λ) capacity or nothing. Switching a whole optical channel is often (very) inefficient, since each optical channel has a capacity ranging from 2.5 *Gbit/s* to 100 *Gbit/s* and can accommodate a very large number of conventional IP sessions/connections. Thus, it is more bandwidth efficient if an optical channel can be partitioned into a number of sub-lambda or fractional lambda channels.

A single wavelength can carry a huge bandwidth that is much larger than the bandwidth demand of a single user. Grooming the traffic from multiple users at the end-points of optical channels is required to improve the throughput of wavelength routed networks and obtain an efficient use. However, traffic grooming is an expensive operation, it introduces additional delay, and it is complex since traffic rivulets come from different sources with different constraints (e.g., quality of service (QoS) requirements). One possible solution is the implementation of robust asynchronous IP-packet switching at each core node. This seems to be an unattractive approach toward the goal of realizing an all-optical networking, since optical-electronic-optical conversion is required. IP-packet switching architectures require huge buffering and introduce large delays. Therefore, there are real needs for some practical implementations of sub-wavelength switching.

Fractional lambda switching (FλS) (also known as TDS) [6, 7, 27, 48] is a novel network architecture for the management of optical networks with sub-wavelength granularity. FλS allows dynamic switching fractions of wavelength in heterogenous and meshed networking environment. FλS offers deterministic performances with low implementation complexity, hence scalability.

In FλS networks, every working wavelength is partitioned into time-frames, grouped into time-cycle, which are coordinated by using a common time reference (CTR). Given the demand for a connection between a source and destination a fractional lambda pipe (FLP) occupying an appropriate number of time-frames should be scheduled to satisfy the bandwidth request. At each FλS network node, time-frames can be switched from input channels (wavelengths) to desired output channels, but no or very limited buffering is possible. Therefore, FλS can be seen as an ideal architecture to realize all-optical networks.

The principle underlying FλS networks is pipeline forwarding, a method known to provide optimal performance independent of specific implementation and widely used in manufacturing and computing. The necessary condition for pipeline forwarding is having a CTR, which in the context of this work is global time-of-day or UTC with proper accuracy.

2.2.1 Timing principle

UTC (coordinated universal time, a.k.a. Greenwich Mean Time - GMT) provides phase synchronization and time-of-day with identical frequencies everywhere. UTC can be easily obtained for a low cost from satellite systems such as GLONASS, GPS or Galileo.

FλS requires phase synchronization (i.e., time-of-day), which is entirely different than the very accurate frequency synchronization required by SONET/SDH. What is required for FλS is that the time at any two points around the globe will be within maximum deviation of a few microseconds. Specifically, FλS utilizes the UTC second that is partitioned into a predefined number of time-frames (time-frames). Time-frames can be viewed as virtual containers for multiple variable-length IP packets that are switched as a whole at every TDS switch. The manners in which IP packets within time-frames are switched from inlet to outlet depend on UTC. Namely, for every time-frame within the UTC second there is a well defined switch configuration (i.e., inlet/outlet permutation), which does not drift in time, and consequently, enables deterministic performance and low implementation complexity.

A UTC second is partitioned into time-frames. Time-frames are the basic for scheduling data unit throughout the FλS network as discussed later. A group of K contiguous time-frames forms a time-cycle (TC). L contiguous time-cycles are grouped into a super cycle that is equal to one UTC second.

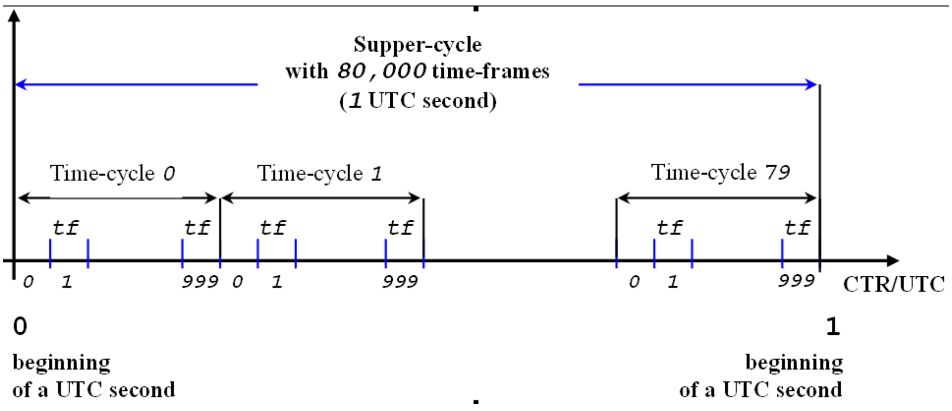


Figure 2.2: Division of an UTC second in TDS/FλS.

An example of UTC time partitioning is shown in Fig. 2.2, where $K = 1000$ and $L = 80$.

In FλS, all time-frames are aligned with UTC at the inlet ports prior to switching. After alignment, the delay between inlets of any pair of switches is an integer number of time-frames, which is the necessary condition for pipeline forwarding.

2.2.2 Forwarding schemes: zero vs. nonzero scheduling delay

Typically, three types of delay present in FλS networks:

- propagation delay: this is common delay in any communication network as signals traverse through communication links with limited velocity.

- alignment delay: since the propagation delay is not an integer number of time-frames, a UTC alignment subsystem is used to round up the delay to an integer number of time-frames at every FλS switch. Thus, the alignment delay is one time-frame duration.

- scheduling delay: this delay varies depending on forwarding mechanisms (discussed below) and a buffering scale deployed at each switching node.

FλS defines two types of time-frame forwarding mechanisms with the corresponding maximum scheduling delays:

Immediate forwarding (IF): — upon the arrival of each time-frame to a TDS switch, the content of time-frame (i.e., IP packets) is scheduled to be "immediately" switched and forwarded to the next switch, as shown in Fig. 2.3. Hence, excluding the alignment delay and the propagation delay, IF requires zero scheduling delay. Henceforth, we use two terms, IF and zero scheduling delay, interchangeable.

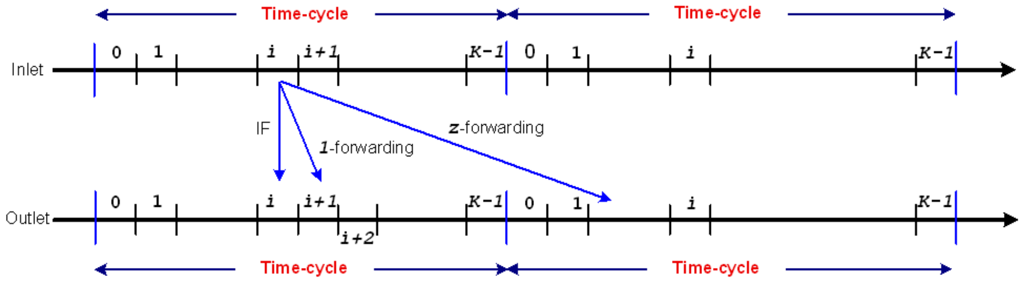


Figure 2.3: Illustration of IF and NIF in the time domain.

Non-immediate forwarding (NIF): — which requires that the content of time-frame is delayed one or more time-frames at the FλS switch (i.e., non-zero scheduling delay). Let us assume that, at each switch inlet there is a buffer for z time-frames. (Note that each buffer can be either an optical delay line or a solid state memory). Thus, the content of each time-frame arriving to the TDS switch can be buffered for an arbitrary number k_z time-frames ($0 \leq k_z \leq z$) before being forwarded to the next switch, as shown in Fig. 2.3, consequently, the maximum scheduling delay is z time-frame durations. (Note that NIF does not exclude IF.) Henceforth, we use two terms interchangeable: NIF and nonzero scheduling delay.

In FλS, prior to data transmission between a source node and a destination node, a FλP must be established between them. A FλP p is defined as a predefined schedule for switching and forwarding certain number of time-frames per time-cycle (or super cycle) along a path of subsequent FλS switches. The FλP capacity is determined by the number of pre-allocated time-frames in every time-cycle (or super cycle). Note that in NIF we imply "arbitrary" only for FλP establishment phase (i.e. while searching a schedule for a given FλP).

2.2.3 Alignment and switching

To obtain a simpler switching control and higher switching fabric utilization that is independent of data unit format and switching technology, all switching data units have the same size (i.e., time-frame duration) and are aligned to UTC. This allows the transfer of all switching data unit from inlets to outlets starts and ends concurrently.

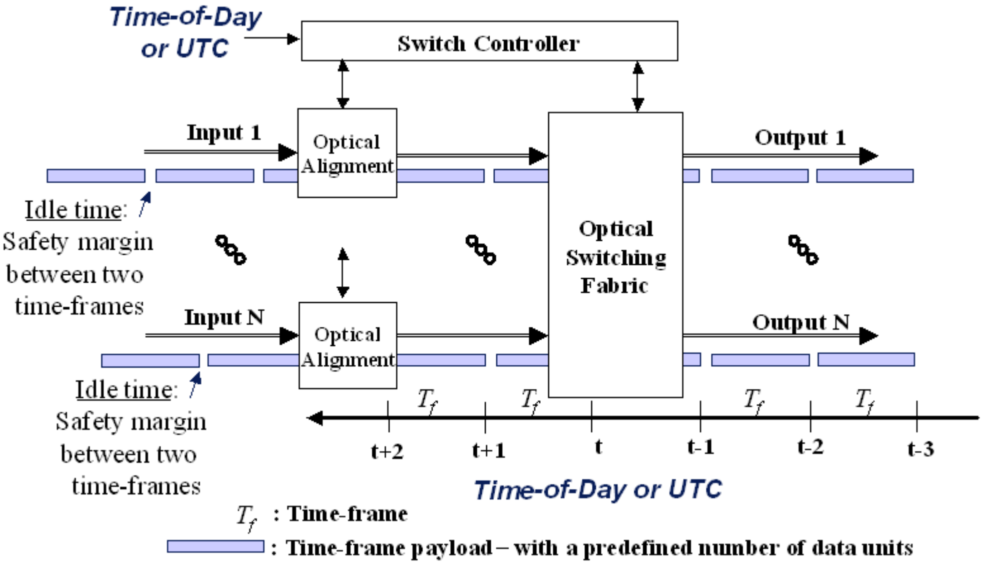


Figure 2.4: Optical alignment subsystem in a FλS node.

Though time-frames are aligned to UTC at transmission sides of all nodes, time-frames arriving at inlets of a FλS switch are usually not aligned. This is simply because the propagation delay across links between nodes are not the integer number of time-frames duration. Therefor, arriving time-frames at inlets of a switch must be aligned to UTC prior to being transferred through switching fabric as depicted in Fig. 2.4. The other way to cope with this issue is to arrange link spans connecting FλS nodes to be equal to a multiple of the time-frame duration.

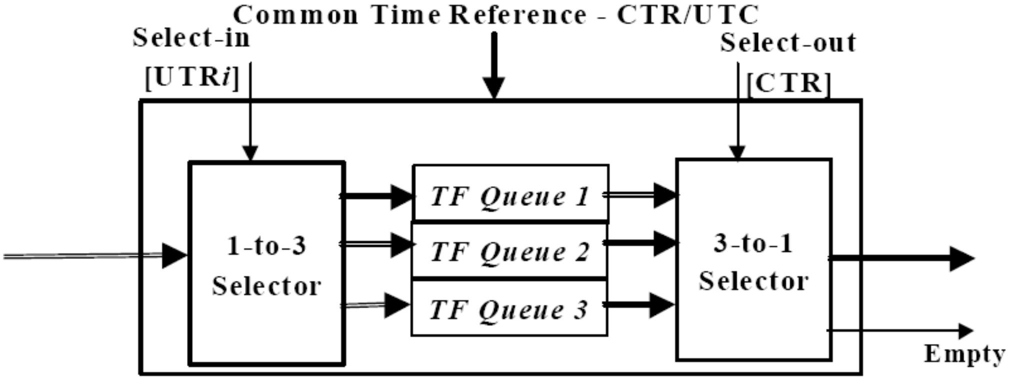


Figure 2.5: An alignment subsystem using three FIFO queues [7].

As shown in Fig. 2.5, an alignment subsystem [7] can be realized using three FIFO queues with mutually read and write access. With this alignment subsystem, the control is simple and no memory access speedup is required since a buffer is never read or written at the same time.

The switching control is simple since switching configuration change once per every time-frame, and the pattern reoccurs after a time-cycle. The switching pattern is also known in advance as the set of all schedules at the switch for established FλPs going through it, thus, no switching speedup is required and there is no output contention.

2.3 Sub-wavelength switching in optical networks

In this section, we review other architectures attempting sub-wavelength switching, including OBS, OPS, time and wavelength interleaving approaches like time-domain wavelength interleaved network (TWIN), time slot interchange (TSI).

2.3.1 Optical packet switching

With almost a half of century development (the packet switching concept was first discovered by Paul Baran in the early 1960's, which then stepped toward the ARPANET, the first packet switching network), packet switching technologies have archived astonishing results as now the world is fast and highly connected by packet switching infrastructures. However, it seems that electronic packet switching almost reaches its limits as does the Moore's law in electronic domain.

In the mean time, what is fast now is forseeably slow in future as the matter of fact that traffic injections into the Internet keep blooming exponentially because of the pressures of wide deployments of VoD, interactive games, IPtv, grid computing applications, etc. While DWDM technology allows up to hundreds of $Tbit/s$ per transmission link, the bottleneck is obviously the limit of electronics packet switching technologies. They are not fast enough to catch up the DWDM transmission technology and their limits are seen. Thus, an OPS network [62] is the ultimate goal for all-optical networking.

As its name, OPS is not something strange and far different from its ancestor - the packet switching in electronic domain. All what is different is that OPS - an asynchronous network aims at processing everything at the photonic level. For instance, in stead of using electronic random access memory (RAM) to store packets awaiting for processing and/or forwarding, OPS uses optical random access memory (ORAM). Packet header processing will be also carried out at the optical level. All electronic processor and devices are going to be replaced by corresponding optical ones in order to avoid electronic limits.

Ideally, all advances in asynchronous packet switching can be straightforwardly applied to OPS and advances pay off complexity levels. Unfortunately, if ever do photonic technologies not mature enough, the complexity is multiplied. And this seems to be the picture of OPS for substantial number of years, for some tens or twenty more years or even longer. At least, two key technological hurdles must be overcome: realizing large asynchronous ORAM and asynchronous optical packet header processing, while ensuring adequate optical power budget and signal to noise ratio.

If these two key technologies mature, there is still a question postured such as how to completely stay away of electronic domain while the controls of all optic devices are done by electronics. And the controller of OPS switch fabrics, will it be all optical processing or electrical/electronic processing?

2.3.2 Optical burst switching

While consistently waiting for the birth of pure OPS networks, optical burst switching (OBS) [51, 69] was proposed as a middle stage. OBS can be seen as a special OPS where "large packets" (or bursts) are used. OBS, thus, is also an asynchronous switching technology. A burst accommodate several (terms of hundreds) of packets from different sources. In OBS, control packets are forwarded in a control channel to configure switching nodes before the arrival of corresponding data bursts, hence, reducing the requirement of optical buffers.

Though OBS is interesting and some protocols were defined for it [20, 74], the behavior of burst switching as an asynchronous switching system makes it hard to implement and control switching fabrics when the traffic load is moderate to high. This consequently leads to high loss or low throughput networks as reported in many researches [34, 39, 40, 50, 67].

Besides, grooming traffic into bursts [23, 50] at ingress nodes of OBS networks and contention resolutions [26, 75] for bursts inside the OBS networks are two other difficult issues.

All in all, slotted OBS/OPS [5, 52] were also studied. However, they all lacked of discussion of timing issues (e.g., how to obtain accurate synchronization through out network wide). Moreover, they were not pipeline forwarding, thus high overhead is another issue that make them not scalable.

2.3.3 Time-domain sub-wavelength switching

In the past ten years there were a number of works on combining WDM with time division multiplexing (TDM) [25, 28, 29, 32]. None of these works was using UTC with pipeline forwarding, as discussed in Section 2.2.1, and did not provide the necessary detailed analysis of critical timing issues. Specifically, regarding the accumulation of delay uncertainties or jitter and clock drifts, which is solved by using UTC with pipeline forwarding, as discussed in Section 2.2.2.

In [29], an optical time slot interchange (TSI) utilizing sophisticated optical delay lines is described with no detailed timing analysis. In [32] and [25] two experimental optical systems with in-band master clock distribution and optical delay lines are described, with only limited discussion about timing issues. In [28] a system with constant delays and clocks is described, which can be viewed as a close model of what we define immediate forwarding (in Section 2.2.2), however, no timing analysis and no consideration of non-immediate forwarding (see Section 2.2.2) were presented.

Following are some other related works, which used tunable laser in their network node design or implicitly utilize UTC in their network synchronization.

TWIN

More recently, the idea of utilizing UTC in order to forward bursts of data in optical networks was proposed in the time-domain wavelength interleaved network (TWIN) architecture [56, 71]. TWIN proposed to use fast tunable lasers at the network edge nodes while the core switching nodes are selective wavelength routers. Each edge node is equipped with a unique wavelength receiver. When one edge node transmits to another edge node it tunes its tunable laser to the unique wavelength receiver of that node.

The TWIN architecture requires network-wide scheduling algorithms in order to ensure that each unique tunable receiver receives only one (burst) transmission at a time. Consequently, TWIN has limited wavelength reuse and can only efficiently accommodate bursts that are larger than the end-to-end propagation delay. Thus, TWIN may be suitable for local area networks. The link delay issue becomes particularly significant if wide-area networks (WANs) are considered.

It is also very worthy to note that if the TWIN architecture operates with near zero propagation delay and source-destination route length is two (i.e., only one core node), it will be equivalent to our second FλS tunable laser switch design (called WR-FλS) presented in Section 3.4 - Chapter 3).

WONDER

RINGO [9, 15] and WONDER [8, 12] dedicated to the design and experiments of slotted OPS rings for future metropolitan networks. Nodes in RINGO/WONDER and TWIN share the same designing characteristics: (*i*) each node is identified by a specific wavelength λ_i and it is the unique node able to receive signal on this wavelength; (*ii*) each node is equipped with a tunable transmitter (or tunable laser), which can be fully tuned to any among all working wavelengths so that it can communicate to all the other nodes of the network. Thus, in general TWIN and RING/WONDER are much similar.

However, RINGO/WONDER differs from TWIN in points of network synchronization and of MAC layer. While TWIN implicitly use UTC for synchronizing nodes and nodes transmit signals based on predefined schedules, RING/WONDER use conventional methods: ; *a*) nodes are synchronized by using a dedicated wavelength for conveying synchronization signals transmitted by the master node; *b*) no predefined schedules are required and nodes must have ability to sense busy/free states of all wavelengths to avoid transmission collisions.

TSI-WDM

The concept of time slot interchange (TSI) WDM networks appeared in some papers [64, 65, 77]. In TSI-WDM, a wavelength is partitioned in a number of time-slots so that multiple source-destination pairs can share one wavelength.

First, in these works no practical node design is shown. There is no or lack of discussion on synchronization issues. Second, a TSI device is capable of rearranging the order of the time-slots passing through it. TSI is originally stemmed from conventional TDM switching technologies. Thus, optical TSI requires optical random access memory (ORAM) to store time-slots. This is not practical since ORAM is not available. Moreover, a TSI-WDM network is a subcase of FλS where full forwarding is deployed (i.e., z-forwarding with $z = K - 1$).

Chapter 3

Switch Designs Using Tunable Lasers

This chapter presents three novel switch designs that are based on the use of tunable lasers (which can be replaced in the future with wavelength converters). The analytical results of *scheduling feasibility*, which measures the total number of possible different schedules for each switch design, are discussed. Then it is shown that the architecture with the *highest scheduling feasibility* is *strictly non blocking* in the space domain.

In addition, we present measures of the switching hardware complexity, which, for the strictly non-blocking architecture, has the same switching complexity as Clos interconnection network, i.e., $O(N'\sqrt{N'})$ where N' is the number of optical channel.

3.1 Tunable Laser Principle – Wavelength Swapping

We focus on FλS with tunable lasers introduced in Section 2.1.1 - Chapter 2. In general, the way tunable lasers are used in this work is to change the wavelength (color) of time-frames that contain IP packets at every FλS node. When wavelength converters will be available they may replace the tunable lasers.

This operation can be viewed as wavelength swapping of packets. Namely, packets are transmitted with λ_1 over the first optical link, then with λ_2 over the second optical link and so on. The operation of swapping wavelength (color) is equivalent to label swapping. Obviously, as in label swapping, packets of different connections (FλPs) should not have the same color (label) when being transmitted over the same optical link and having the same time index within the time-cycle.

Note that there is no “stop” of bit streams at switching nodes as we discussed the usage of tunable lasers in our design is somewhat similar to [36, 79] (see Section 2.1.1 - Chapter 2).

3.2 Designing criteria

The goal of a switching architecture is keeping complexity and cost at a minimum level while providing high performance and low blocking probability for incoming new flows. We introduce three tunable laser based FλS switches and discuss their hardware cost and complexity, as well as their suitability for deploying flexible routing strategies.

The performance of flow-based switching is measured by blocking, which is due to two different phenomena in time driven switching. External-or time-blocking-is the impossibility of finding a time-frame on a suitable optical channel on the proper output port to set up a FλP across the switch. [1] Internal-or space-blocking-is instead the impossibility of setting up the FλP due to internal constraints of the switch although the proper resources at the output port are available.

The different tunable laser switch architectures are compared using:

- *i*) the hardware complexity;
- *ii*) the performance in terms of scheduling feasibility as defined below in Def.3.2.1.

[1] The time-blocking issue is formally formulated and discussed in Chapter 4.

The scheduling feasibility directly influences space-blocking, although there is no explicit mathematical relationship between the two; in Section 3.7 we demonstrate that the architecture with the highest scheduling feasibility is strictly non-space-blocking.

In order to give consistent and convenient descriptions of the different switch architectures, the following notations are used:

- C is the link capacity in terms of the number of optical channels (colors) per optical fiber, which is associated with each input/output port;
- N is the number of input/output ports (or in-ports/out-ports for short) per switch;
- $r = C/N$ is the internal connection ratio. For simplicity it is assumed that r is integer.
- R_T is the tuning range of a tunable laser;
- K is the size of time-cycle in number of time-frames;
- h is the route length of a FλP in number of hops.

Additionally we use the following acronyms to identify the building blocks of the architectures:

- MUX and DEMUX are wavelength multiplexors and de-multiplexors; they operate between optical fibers with WDM channels and the in-/out-ports;
- tunable laser is a device with tuning range R_T that operates the λ swapping; TL(n, c) means the tunable laser connected to the c-th optical channel of in-port n;
- WR is a static wavelength router with fixed permutation pattern;

- SC is a star coupler, i.e., one-to-n broadcast device; SC(n, c) is the star coupler connected to the c-th tunable laser of in-port n;
- OO is an ON/OFF switching element; OO(n, c, n') is the ON/OFF switching element connecting in-port n with out-port n' using the tunable laser c;
- TuF is a tunable filter; TuF(c, n') filters the output of a star coupler c toward the out-port n'.

Definition 3.2.1 (Scheduling feasibility)**.** — For a generic FλS the scheduling feasibility is the number of *distinct schedules* that are available using time and wavelength swapping. The scheduling feasibility is function of the forwarding method (immediate forwarding (IF) or non-immediate forwarding (NIF)), K, C and N, on a given route with h hops (where h is not a variable for feasibility measure).

A *schedule* is defined as a possible allocation of time-frames and wavelength swapping along a given route so that a FλP can be set up. In fact, scheduling feasibility indicates a relative (not absolute) measure for how flexible the scheduling can be for each tunable laser switch architecture.

A feasible schedule is not guaranteed to be available at the time of FλP setup due to the space- or time-blocking (e.g., switching fabric limitation, contention between multiple setups); however, it is clear that the more the available schedules are, the less is the chance that it is not possible to find a non-blocked schedule. The switch architectures studied in this chapter have four key common parts:

1. WDM de-multiplexers on the in-port side;
2. WDM multiplexers on the out-port side;
3. Tunable lasers at the output of the WDM de-multiplexers;

4. A connection network between the tunable lasers and the WDM multiplexers at the out-ports, which is in essence what distinguishes the switch architectures discussed in this chapter.

We define the following three switch architectures:

- *Tunable laser with fixed connection network* (FC-FλS): The fixed connection network consists of point-to-point links from tunable lasers to out-port MUXs.
- *Tunable laser with static wavelength router* (WR-FλS): The static wavelength router does not change its configuration over time.
- *Tunable laser with broadcast and select* (BS-FλS): The broadcast and select operation is time dependent and the connection configuration can change every time-frame.

For the sake of simplicity, we do not show in figures how to implement buffering. In principle, a tunable laser behaves as an optical-electronic-optical conversion device. Specifically, the incoming optical signal serial-bit-stream is converted to electronic signal that is used directly to modulate the tunable laser, and thereby, convert back to optical signal without "stopping" the serial-bit-stream. Thus, buffering can be done optically with programmable fiber-delay-lines. Note that this is only one possible tunable laser design.

3.3 FC-FλS: a fabric-less design

3.3.1 Design description

Fig. 3.1 shows the simple design of the FC-FλS for C=4, N=2 which uses tunable lasers with a fixed point-to-point connection network. DMUX separates WDM signals into C different wavelengths. Each incoming wave-

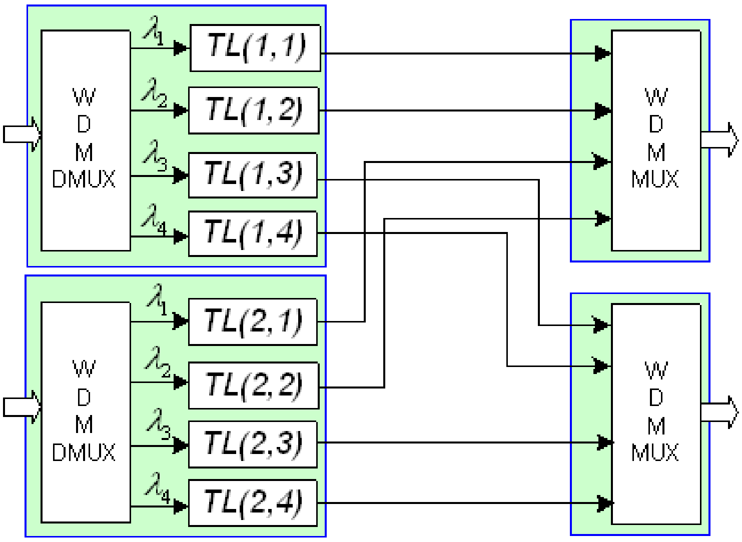


Figure 3.1: An illustration of a 2 × 2 FC-FλS switch with C=4 (TLs are coordinated by UTC time signal, which is not shown)

length is fed to a tunable laser that transmits at any wavelength within its tuning range R_T. The output of each tunable laser is connected to a predefined out-port. The number of fixed connections between an in-port/out-port pair is equal to r, i.e., a switch with N=8 and C=16 has 2 fixed connections between any in-port/out-port pair.

Tunable lasers are tuned every time-frame, where time-frames are derived from Universal Time Coordinated (UTC), such that time-frames are switched from in-ports to out-ports without conflicts at any out-port. Due to the nature of the fixed connection system, the color of a time-frame after switching defines the out-port, and hence, it defines the route it takes.

3.3.2 Hardware complexity and scheduling feasibility

The hardware complexity of this design is CN tunable lasers. Each in-port requires C tunable lasers, corresponding to C channels. The in-port DMUX and out-port MUX devices are not counted in the hardware complexity since they are identical for all the designs described in this chapter.

Scheduling time-frames using FC-FλS is rigid due to the nature of fixed point-to-point internal connection network. To route a time-frame along a predefined route path between source and destination, a tunable laser that receives a signal must tune the output to one wavelength among r. For simplicity, we assume that lasers have full tunable range, that is $R_T = C$. With this assumption, the scheduling feasibilities of this design are given in (3.3.1) for IF, and in (3.3.2) for NIF:

$$S_{FC}^{(IF)} = Kr^h = K\left(\frac{C}{N}\right)^h \tag{3.3.1}$$

$$S_{FC}^{(NIF)} = Kr^h(z+1)^{h-1} = K\left(\frac{C}{N}\right)^h (z+1)^{h-1} \tag{3.3.2}$$

Proof. Eq.(3.3.1): At the 1^{st} hop, to forward a time-frame to the 2^{nd} hop of the defined route, a time-frame must be carried on 1 of r wavelengths; each channel has K different time-frames. Hence, there are Kr scheduling choices for the 1^{st} hop. The following $(h-1)$ hops are all identical and there are only r possible schedules at each hop. Scheduling at all hops is independent. Therefore, the number of possible schedules is given by the product

$$\{Kr\}_{1^{st}} \times \{r\}_{2^{nd}} \times ... \times \{r\}_{h^{th}}$$

of all the possible single hop schedules. $\{*\}_{h^{th}}$ is the contribution of h^{th} hop to the combinatorial result.

Eq.(3.3.2): The 1^{st} hop contribution is equal to that of (3.3.1). For the other contributions, there are more options to forward a time-frame thanks to non-immediate forwarding (NIF). A time-frame can be switched immediately or buffered for up to z time-frames[2], before being switched. Thus, for all hops except the 1^{st} one, there are $r(z+1)$ options to schedule

[2] Note that NIF does not exclude IF, see Section 2.2.2 - Chapter 2.

a time-frame. The final result is given by the product

$$\{Kr\}_{1^{st}} \times \{r(z+1)\}_{2^{nd}} \times \cdots \times \{r(z+1)\}_{h^{th}}$$

thus, we yield $S_{FC}^{(NIF)}$. □

Note that if $z = 0$ is applied to (3.3.2), we obtain $S_{FC}^{(IF)}$ in (3.3.1). This asserts that NIF does not exclude IF.

3.3.3 Robustness and practical issues

Though FC-FλS has a simple design with low cost and low control overhead, a network implemented with FC-FλSs is subject to some disadvantages. First, it is hard to deploy different routing protocols since routing is rigid due to the nature of fixed internal connection network. Second, for the IF scheme the scheduling flexibility of this design strongly depends on the internal connection ratio r, as shown in (3.3.1), requiring many wavelength channels for good performance.

3.4 WR-FλS: a TWIN-like design

3.4.1 Design description

An example of the design using tunable lasers and static wavelength router (WR) is depicted in Fig. 3.2. The idea for this design is built on an optical burst switching (OBS) switch design described in [53]. The key characteristic of this design is that different in-ports use different sets of channels, whose size is r and depends on the permutation pattern, to reach the same out-port. More specifically, in order to switch a time-frame received by TL(n, c) to out-port n', TL(n, c) must tune to one among r channels defined by the designed permutation pattern so that the transmitted time-frame can reach MUX(n, n'). Two common types for the selection of fixed

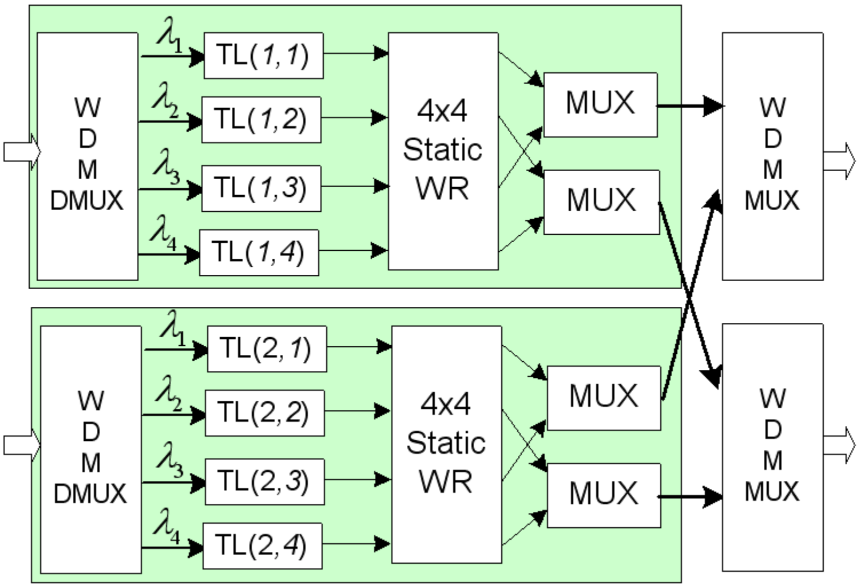


Figure 3.2: An example of 2 × 2 WR-FλS switch where UTC time signal is not shown.

permutation pattern are *contiguous wavelength selection* and *randomized wavelength selection* [53].

Note that if the WR-FλS switch architecture is distributed, namely, if the tunable lasers are connected to the WRs by long optical links, those tunable lasers can be seen as edge nodes. Such tunable laser are similar to edge nodes in time-domain wavelength interleaved network (TWIN) [56][72]. Moreover, if out-ports of WR-FλS are also connected to a center WR by long optical links, a node that is similar to a core TWIN node is formed. Thus, we can infer that TWIN and a modified version of WR-FλS are similar.

3.4.2 Hardware complexity and scheduling feasibility

WR-FλS requires CN tunable lasers, N modules of $C \times C$ static WRs, and N^2 multiplexers at the output of the WRs. The scheduling feasibility of WR-FλS for both IF and NIF schemes are given in (3.4.1) and (3.4.2):

$$S_{WR}^{(IF)} = KCr^{h-1} = K\left(\frac{C}{N}\right)^{h} N \tag{3.4.1}$$

$$S_{WR}^{(NIF)} = KC\{r(z+1)\}^{h-1} = K\left(\frac{C}{N}\right)^{h} (z+1)^{h-1}N \tag{3.4.2}$$

Proof. The proof can be done following the same scheme used to prove (3.3.1) and (3.3.2). Using WR-FλS, there are always KC options to select a time-frame for the 1^{st} hop, since no constraint on routing exists. For the 2^{nd} to h^{th} hops, an incoming time-frame has only r options to reach a desired out-port, assuming again $R_T = C$. Therefore, the product of all hop-based components is given as

$$\{KC\}_{1^{st}} \times \{r\}_{2^{nd}} \times ... \times \{r\}_{h^{th}}$$

for IF and

$$\{KC\}_{1^{st}} \times \{r(z+1)\}_{2^{nd}} \times ... \times \{r(z+1)\}_{h^{th}}$$

for NIF. Therefore, we obtain (3.4.1) and (3.4.2). Again note that applying $z = 0$ into (3.4.2) we yield (3.4.1) as NIF includes IF. □

3.4.3 Robustness and practical issues

Networks using WR-FλS have no constraints on routing, since time-frames coming to an in-port can reach any out-port. The scheduling feasibility is still limited by r, which is a strong constraint to the scalability. Although routing is not limited, space-blocking is possible in this architecture.

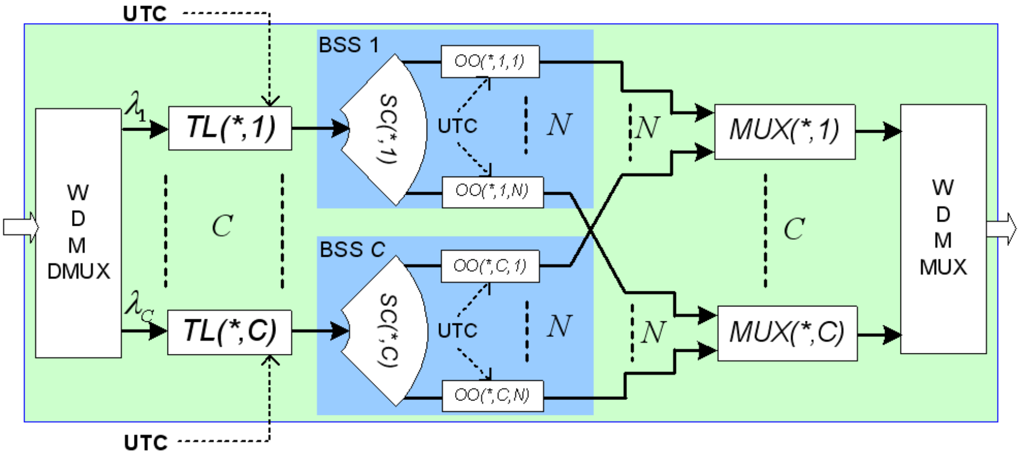


Figure 3.3: BS-FλS, a strictly non-space-blocking architecture

3.5 BS-FλS: a strictly non-space blocking design

3.5.1 Design description

The illustration of BS-FλS design is shown in Fig. 3.3. This design uses one tunable laser and one broadcast-and-select switching (BSS) component per channel. A BSS is composed by the combination of a single 1-to-N star-coupler (SC) and N simple ON/OFF switching elements.

TL(n, c) receives the signal of λ_c and then transmits using any channel in its tunable range. The transmitted signal from a laser is broadcast to all out-ports using the star-coupler SC(n, c) and it is allowed to reach a single out-port enabling the corresponding ON/OFF switching element to that port. The BSS design also enables multicasting. All tunable lasers and ON/OFF switching elements are controlled and coordinated using the UTC signal.

The BS-FλS design allows a tunable laser to transmit time-frames to all out-ports. Moreover, BS-FλS has the advantage over WR-FλS that a tunable laser can transmit time-frames to any out-port using the full channel range C, assuming $R_T = C$, while WR-FλS only allows using the small fixed set of channels r. Thus, compared to WR-FλS, BS-FλS has a much larger scheduling feasibility.

3.5.2 Hardware complexity and scheduling feasibility

The hardware requirements for BS-FλS design are: CN tunable lasers, CN star-coupler modules, CN^2 programmable ON/OFF switching elements. The scheduling feasibility of BS-FλS design for both IF and NIF schemes are given in (3.5.1) and (3.5.2):

$$S_{BS}^{(IF)} = KC^h = K\left(\frac{C}{N}\right)^h N^h \tag{3.5.1}$$

$$S_{BS}^{(NIF)} = KC\{C(z+1)\}^{h-1} = K\left(\frac{C}{N}\right)^h (z+1)^{h-1}N^h \tag{3.5.2}$$

Proof. For the 1^{st} hop, there are KC options to schedule one time-frame, since every channel can be routed following any predefined route. For the 2^{nd} to h^{th} hops, a tunable laser can exploit all the C channels to transmit the signal. In fact, if available time-frames are found at both incoming and outgoing channels, there is a path to schedule the transmission. Therefore, the product of all hop-based components for IF scheme is:

$$\{KC\}_{1^{st}} \times \{C\}_{2^{nd}} \times ... \times \{C\}_{h^{th}}$$

and for NIF scheme it is:

$$\{KC\}_{1^{st}} \times \{C(z+1)\}_{2^{nd}} \times ... \times \{C(z+1)\}_{h^{th}}$$

Note that $S_{BS}^{(IF)}$ and $S_{BS}^{(NIF)}$ are independent from r. The right most expressions in (3.5.1) and (3.5.2) are only for comparison purposes with the other architectures. □

In term of scheduling feasibility, the BS-FλS design gains N^h times compared to the WR-FλS design in both IF and NIF schemes. It is also worthy to highlight the following observations on this design.

Remark 3.5.1 (If less number of SCs used). Using a single SC per in-port, then the scheduling feasibility of the BS-FλS design reduces C times.

Proof. Let us assume that all channels of an in-port share a single SC. SC is a broadcast device, meaning that a signal at a given input is broadcasted to all outputs. At every time-frame strictly one and only one signal can be fed to one of the inputs of SC, otherwise there is conflict. Hence, if all C tunable lasers of an in-port share the same SC, at every time-frame only one of them is allowed to transmit, therefore resulting in the reduction of the utilization of the design by C, compared to the design that deploys a single SC per tunable laser. □

Remark 3.5.2 (If design based on filters). A tunable filter per out-port can be used in replacement of the CN ON/OFF switching elements. In this case the scheduling feasibility is bounded by:

$$KC\left(C'\right)^{h-1} \leq S^{(IF)}_{Filter} \leq K\left(\frac{C}{N}\right)^{h} N^{h}$$

and

$$KC\left(C'\right)^{h-1}(z+1)^{h-1} \leq S^{(NIF)}_{Filter} \leq K\left(\frac{C}{N}\right)^{h}(z+1)^{h-1}N^{h}$$

where $C' = (C - N - 1) \geq 0$.

Proof. Assume that ON/OFF switching elements are removed and outputs of SC devices are connected to tunable filters (TuF), as shown in Fig. 3.4. At a given time-frame, TL(n, c) is scheduled to transmit to out-port n' and TL(m, c) is scheduled to transmit to out-port m', both using channel $\lambda_{c'}$. Consequently, there are conflicts at both inputs of TuF(n', c) and TuF(m', c). Therefore, a given tunable laser must coordinate with all the other $(N-1)$ tunable lasers that are connected to the TuF for transmitting to an out-port. In the worst case, a given tunable laser has only $C' =$

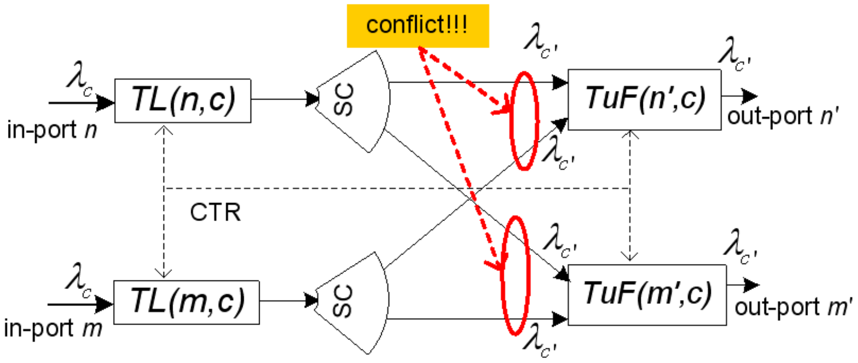


Figure 3.4: One tunable filter replaces N ON/OFF switching elements producing internal conflicts.

$(C - N - 1)$ channel options, since the other $(N - 1)$ channels are used by the other tunable lasers. This yields a lower bound of

$$\{KC\}_{1^{st}} \times \{C'\}_{2^{nd}} \times ... \times \{C'\}_{h^{th}}$$

for IF scheme, and

$$\{KC\}_{1^{st}} \times \{C'(z+1)\}_{2^{nd}} \times ... \times \{C'(z+1)\}_{h^{th}}$$

for the NIF scheme. The internal blocking due to conflicts in the TuF cannot be accounted for with combinatorial analysis, thus we can only give the upper and lower bounds of the scheduling feasibility. □

3.5.3 Robustness and practical issues

BS-FλS is a *strictly non-blocking* design in the space domain (see the proof in Section 3.7). An incoming time-frame always finds the path to be forwarded to a desired out-port if a free corresponding time-frame is found at the outgoing channel. The BS-FλS design also allows deploying multicast and broadcast easily.

3.6 Comparisons between designs

The comparison among the three switch designs is summarized in TABLE 3.1.

Table 3.1: Comparisons between tunable laser-based FλS switch designs for a given h.

Design	Hardware				Scheduling Feasibility		Routing Adapt.
	N_{TL}	N_{WR}	N_{SC}	N_{OO}	IF scheme	NIF scheme	
FC	NC	- -	- -	- -	$K\left(\frac{C}{N}\right)^h$	$K\left(\frac{C}{N}\right)^h (z+1)^{h-1}$	None
WR	NC	N	- -	- -	$K\left(\frac{C}{N}\right)^h N$	$K\left(\frac{C}{N}\right)^h (z+1)^{h-1}N$	Partial
BS	NC	- -	NC	N^2C	$K\left(\frac{C}{N}\right)^h N^h$	$K\left(\frac{C}{N}\right)^h (z+1)^{h-1}N^h$	Full

Parameters to be compared include hardware complexity, scheduling feasibility and optical routing adaptability. Optical routing adaptability indicates the freedom of changing the routing wavelength on the same optical fiber. For instance, the color of a time-frame coming to an in-port of a FC-FλS node will fit a unique next-hop of that time-frame no matter of how the corresponding tunable laser is tuned. For a WR-FλS node, the next-hop of an incoming time-frame can be partially controlled depending on a fixed configuration of internal WRs. With a BS-FλS node, the next-hop for an incoming time-frame is fully controllable.

Design components that are the same in all switch designs, such as WDM-MUX and WDM-DMUX are not shown in this comparison table. N_{TL}, N_{WR}, N_{SC}, N_{OO} stand for the number of TLs, $C \times C$ static WRs, 1-to-N SCs, ON/OFF switching elements, respectively.

Fig. 3.5 shows some plots of the scheduling feasibility $S^{(IF)}$ and $S^{(NIF)}$ of the architectures we introduced. The number of time-frames per time-cycle, K, as well as the optical buffer size z are kept small to avoid numerical problems, since both $S^{(IF)}$ and $S^{(NIF)}$ grows exponentially. First, the graph suggests that the scheduling feasibility may be a good indication of the switch architecture performance in terms of blocking. Though there is

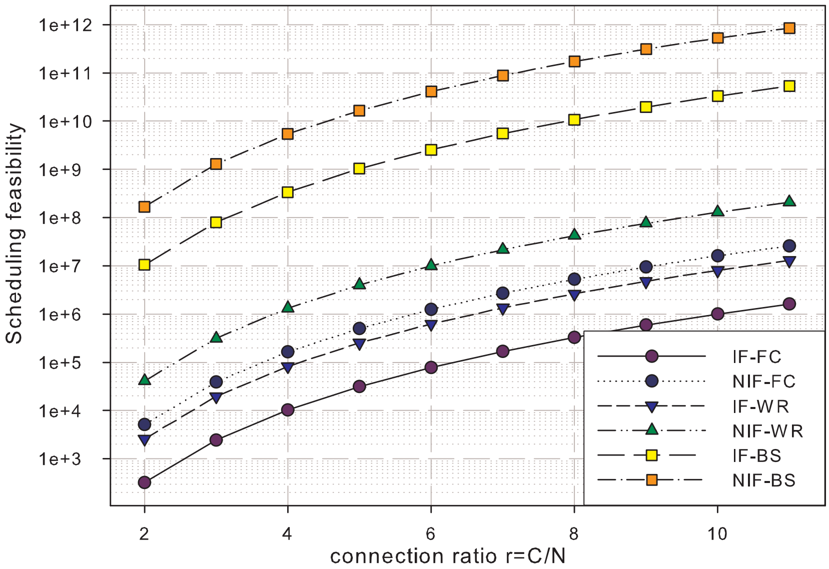


Figure 3.5: Scheduling feasibility vs. connection ratio r when $z = 2$, $K = 10$, $N = 8$ and $h = 5$.

no mathematical relation between scheduling flexibility and blocking performance, it is clear that for a given h, the larger the number of distinct schedules is, the better the chance that a schedule can be found, thus improving the overall blocking performance. Second, the graph highlights the fact that the number of possible FλP schedules in so large that proper signaling and heuristics must be found to exploit the resources of a FλS network.

We have discussed how both the FC-FλS and WR-FλS architectures have limitations in optical routing adaptability, while the BS-FλS can support any routing algorithm. In the next section, we will show that the BS-FλS is strictly non-space-blocking. FC-FλS and WR-FλS, instead, have internal space-blocking.

3.7 A strictly non-space blocking switch for FλS

In this section, we focus on the more general BS-FλS design, since it has been proved to have the highest scheduling feasibility in Section 3.6. We formally prove that this broadcast-and-select design is strictly non-blocking in space domain. The formal definition of a strictly non-blocking FλS design in space domain is given following in (*Def.* 3.7.2). Intuitively, if there is available capacity at both in-port and out-port (i.e. free time-frames to satisfy the IF scheme) but the switch can not configure itself to form a forwarding path (i.e. no more available resource in the fabric), we see it as the blocking event in space domain.

We assume that at anytime there is at most one setup request to forward one time-frame from a given inlet and to a given outlet[3]. For the sake of clarity, we introduce the following notations:

- $tf_{n,c,k}$ denotes a time-frame k belonging inlet c of in-port n.
- $tf'_{n',c',k'}$ denotes a time-frame k' belonging outlet c' of out-port n'.
- $\left\{tf'_{n',c',k+1}\right\}$ denotes the set of all immediate-forwarding positions (i.e., $k' = k+1$) of out-port n', with assumption $R_T = C$.

We give the following definitions:

Definition 3.7.1 (Schedulable time-frame)**.** — A time-frame $tf_{n,c,k}$ is said to be *schedulable* if and only if $tf_{n,c,k}$ is free and at least one time-frame in the set $\left\{tf'_{n',c',k+1}\right\}$ is free. A time-frame $tf_{n,c,k}$ is said to be *schedulable to* $tf'_{n',c',k+1}$ if and only if $tf_{n,c,k}$ is *schedulable* and $tf'_{n',c',k+1}$ is *free.* Note that the definition is valid only for the IF scheme.

[3] It is important to distinguish between an "in-port" and an "inlet", and between an "out-port" and an "outlet". In/out-port indicates the fiber port, whereas inlet/outlet indicates a single wavelength or optical channel.

Definition 3.7.2 (Strictly non-space-blocking FλS switch)**.** — A FλS switching fabric is considered *strictly non-blocking* in space domain if and only if any connection between a given in-port and a given out-port can be established immediately to forward an arbitrary *schedulable* time-frame without interference with any arbitrary existing connection.

Theorem 3.7.1 (Strictly non-space blocking design)**.** *If a time-frame* $tf_{n,c,k}$ *is schedulable to* $tf'_{n',c',k+1}$*, then the forwarding path*

$$fp \quad \models \quad tf_{n,c,k} \rightarrow TL(n,c) \rightarrow SC(n,c) \rightarrow OO(n,c,n') \rightarrow tf'_{n',c',k+1}$$

is always successfully setup during time-frame k, without any interference with existing forwarding paths.

Proof. The proof is obtained by showing that violating the setup postulate, implies that $tf_{n,c,k}$ is NOT *schedulable to* $tf'_{n',c',k+1}$. To setup fp, all devices $\big(\text{TL}(n,c),\ \text{SC}(n,c),\ \text{OO}(n,c,n')\big)$ involved in fp must be available during time-frame k.

Let us denote S_X^k the status of device X during time-frame k, that is:

$$S_X^k = \begin{cases} \text{‘}\texttt{0}\text{’} & \text{if item } X \text{ is } \textit{busy} \text{ during time-frame } k \\ \text{‘}\texttt{1}\text{’} & \text{if item } X \text{ is } \textit{free} \text{ during time-frame } k \end{cases}$$

- Assume $S^k_{TL(n,c)}$ =‘0’ $\Rightarrow$ $tf_{n,c,k}$ is busy, it is not *schedulable* (violate the setup postulate).

- Assume $S^k_{SC(n,c)}$ =‘0’ $\Rightarrow$ $S^k_{TL(n,c)}$ =‘0’ $\Rightarrow$ $tf_{n,c,k}$ is not *schedulable.*

- Assume that during time-frame k, another tunable laser of a certain in-port has been scheduled to forward time-frame on channel c' to out-port n', i.e. $tf'_{n',c',k+1}$ is busy $\Rightarrow$ $tf_{n,c,k}$ is *schedulable* but NOT *to* $tf'_{n',c',k+1}$ (violate the setup postulate).

Therefore, $S^k_{SC(n,c)}$ =‘1’ and $S^k_{TL(n,c)}$ =‘1’, implying that $S^k_{OO(n,c,n')}$ =‘1’ (i.e, available during time-frame k). Thus all elements evolved in forwarding path fp are available during time-frame k. In addition since the default status of ON/OFF switching element is OFF and only the scheduled ON/OFF switching element is ON, setting up fp does not interfere with other existing FλPs. □

A $tf_{n,c,k}$ is *schedulable* only if it is *schedulable to* at least one time-frame belonging to the set $\left\{tf'_{n',c',k+1}\right\}$, the above theorem implies that a BS-FλS is strictly non-space-blocking.

Corollary 3.7.2 (Clos equivalency). *If $N = C$ then it implies that the number of inlets/outlets of the switch is N'=NC=N^2. Therefore the hardware complexity in number of ON/OFF switching elements of the BS-FλS design is CN^2=$N'\sqrt{N'}$, which is the same as a Clos interconnection network [17] with N' inlets/outlets.*

This result is significant since the Clos interconnection network is known to have the lowest switching complexity for strictly non-blocking switch matrices. Note that the equivalence is meant only for the number of active switching elements, since the passive optical broadcast cost cannot be quantified in the sense of switching complexity.

3.8 Discussions

In this chapter we presented three switch architectures for fractional lambda switching paradigm. They use tunable lasers (and in the future wavelength converters). As it was shown, the use of tunable lasers has similar attributes, in the optical domain, to label swapping, in the space domain. Three switch architectures were presented:

1. Fixed Connection (FC-FλS);

2. Wavelength Router (WR-FλS);
3. Broadcast and Select (BS-FλS).

While the second architecture can be seen as an equivalency to TWIN, the first and last architectures are entirely novel and most interesting due to their characteristics. It is noted that the BS-FλS architecture proposed in this work also meets tight bound conditions analyzed in [47] for any strictly non blocking design for WDM switches.

The first architecture, FC-FλS, is fabric-less, since it has no optical switching element. However, FC-FλS is limited as indicated by the scheduling feasibility measure and it does not allow for flexible routing.

The last architecture, BS-FλS, has been shown to be strictly non-blocking with the hardware switching complexity that is equivalent to Clos interconnection network (when $C = N$), which is the minimal complexity for strictly non-blocking architectures. The BS-FλS architecture requires only simple switching elements of 1-by-2.

Furthermore, regarding the optical power budget, the BS-FλS has two desirable attributes: (i) equal power distribution and (ii) low insertion loss, e.g. for $N = C = 32$ - an optical switch with 1024-by-1024 optical channels - the power loss is $3 \log_2 32 = 15$dB. (This is the broadcast loss over the 32-by-32 passive optical star.)

Chapter 4

Time-blocking analysis: stand-alone switch

The BS-FλS design introduced in the previous chapter is one among many possible strictly-non-space-blocking designs. However, even when all strictly-non-space-blocking switches like BS-FλS are used in a FλS network, there are still possibilities that a schedule can not be found to establish an end-to-end FλP. Neglect the case when there is no more capacity to assign schedule, we refer the possibility of no schedule can be found is blocking event in time-domain.

This chapter presents a closed-form time-domain analysis of the blocking probability of time-driven switching (TDS) for the single node case. In this work blocking is defined as the occurrence in which transmission resources are available in both inlet and outlet, but there is no schedule. The main constraints for finding a schedule are: (i) the load and (ii) the maximum scheduling delay between inlet availability and outlet availability. As the maximum scheduling delay (buffering) increases the blocking probability is reduced. The outcome of the analysis in this chapter is the exact blocking probabilities for all possible maximum scheduling delays, under all load conditions.

Section 4.1 examines some related works on blocking analysis. Section

4.2 presents a general discussion on blocking and specialize the problem for FλS. Section 4.3 presents the overall approach used to compute the time-blocking probability. Section 4.4 analyzes the blocking problem for the least complex case when there is only one buffer per inlet. The solution for general cases is then presented in Section 4.5. Sections 4.6 and 4.7 end the chapter with some discussions.

4.1 Review of call-blocking performance

Blocking performance analysis has a long history starting with conventional public phone networks deployment [35]. Traditionally, the term 'call blocking' was used in previous works on blocking analysis (e.g., in [10, 33, 57, 81]). 'Call rejection' is considered as an event when no more network resources (e.g., circuits in telephony or radio channels in wireless) can be allocated to successfully establish a new call. Thus, an analysis of 'call rejection' probability is called 'call blocking' probability analysis. When analyzing 'call blocking' probability, traffic patterns and stochastic distributions are taken into accounted.

However, in this thesis, we do not study the blocking probability at the call level. A blocking in the time-domain (formally defined in Section 4.2.1) occurs even when there are available network resources (i.e., available time-frames) at both inlet and outlet of a TDS switch, but no schedule can be found to properly allocate available resources.

4.2 Time-blocking analysis in FλS systems

In TDS, when a single switch is analyzed, there are two basic blocking issues: blocking in the space domain (*space-blocking*); and blocking in the time domain (*time-blocking*).

Intuitively, if there are schedulable time-frames (e.g., free time-frames that satisfy any chosen forwarding scheme) at a pair of inlet and outlet but the switch cannot be configured to form a forwarding path through the switch fabric (i.e., no available resource in the switch fabric), it is defined as a space-blocking. Space-blocking depends on the architecture of the switching fabric. Naturally, space-blocking can be completely avoided by the deployment of strictly non-space-blocking fabrics [1].

On the other hand, even if there is no space-blocking there still can be blocking in the time domain. For instance, consider the IF scheme when there are free time-frames at both an inlet and an outlet of a switch, but the available (free) time-frames have different time index, then those free time-frames cannot be used for IF and there is time-blocking.

The time-blocking is intrinsic in TDS and can be reduced by using buffers for flexible scheduling as in the NIF schemes. Intuitively, NIF offers a greater flexibility in scheduling time-frames at TDS switches, thus resulting in better performances regarding time-blocking probability. For example, assume that time-frame 5 within the TC is available at the inlet and time-frame 7 within the TC is available at the outlet, then with two buffers (or scheduling delay of two time-frames) it is possible to forward the IP packets within time-frame 5 to the outlet at time-frame 7.

This chapter focuses on a quantitative time-blocking probability analysis. The time-blocking probability analysis in this work is a novel combinatorial approach. The main assumption is that all possible load combinations are equally likely. Namely, if a combination is defined by the distribution of b busy time-frames (out of K possible time-frames in each time cycle - TC) in a given inlet and a given outlet, then all such possible combinations are equally likely.

[1] The BS-FλS presented in Chapter 3 is one example of strictly non-space-blocking designs.

4.2.1 Problem formulation

The switch is part of a large network, we assume independence of each channel (i.e., inlet and outlet), thus we can examine a single channel of the switch. Since the traffic loading the channel comes from other nodes, the resources it uses are defined by the other nodes' constraints, and cannot be assigned freely by the considered node. Assuming independence between nodes, we come to the following model for the traffic load.

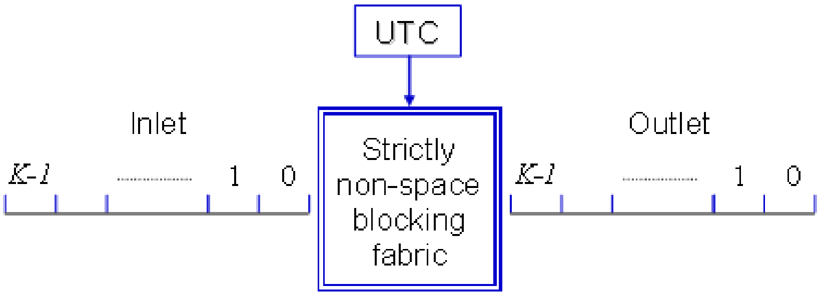


Figure 4.1: Find the time-blocking probability for general NIF schemes.

Load assumptions — The load is defined as the number of busy time-frames per TC per channel. For all channels, the busy time-frames within each TC is assumed to be distributed uniformly. Let b denote the number of busy time-frames per TC. The load of a channel is identified by the pair (K, b) and it is further assumed that all possible combinations are equally likely.

It is further postulated that *i*) the number b is identical for all inlets and outlets, and *ii*) the distribution is independent, i.e., the time-frame distribution of the inlet is independent from the one of the outlet. The later assumption is rather restrictive for small switches, but can be reasonable for large ones. The former assumption is later relaxed without changing much of the analysis. In fact, in Section 4.6 we present modified results to capture the more realistic load assumption (where the load of the inlet and the load of the outlet are different).

To formulate the problem, we further define some notations:

- a denotes the number of free time-frames per TC, $a = K - b$;
- tf_k denotes a generic time-frame k in a TC;
- tf_k^{in} denotes time-frame k of the *inlet*, $0 \leq k < K$;
- tf_k^{out} denotes time-frame k of the *outlet*, $0 \leq k < K$;
- z denotes the number of buffers (or maximum scheduling delay), $0 \leq z < K$;
- symbol '`0`' presents a busy time-frame;
- symbol '`1`' presents an available (or free) time-frame.

Note that a time-frame index has periodic attribute. In other words, if $k \geq K$ then $k = (k \texttt{ mod } K)$ since K time-frames are grouped in a time-cycle.

Definition 4.2.1 (z-forwarding scheme)**.** — A switch is said to be under z-forwarding scheme iff a content of a time-frame, upon its arrival, can be buffered arbitrarily for amount of i time-frames prior to being forwarded, $i = 0, 1, .., z$.

In other words, for the z-forwarding scheme the maximum scheduling delay of an arrival time-frame is equal to z time-frame durations. Note that $z = 0$ means the immediate-forwarding (IF) scheme or zero scheduling delay.

Definition 4.2.2 (A schedulable time-frame)**.** — For a pair of inlet and outlet, a time-frame k of the outlet (i.e., $tf_k^{(ou)}$) is said to be schedulable iff $tf_k^{(ou)}$='`1`' and at least one time-frame in the set $\{tf_{k-i}^{(in)} | i = 0, 1, .., z.\}$[2] is available.

[2] Note that due to periodic attribute of a time-cycle, if $k - i < 0$ then $k - i = K - i + k$.

Definition 4.2.3 (A blocked time-frame)**.** — For a pair of inlet and outlet, A time-frame k of the outlet (i.e., $tf_k^{(ou)}$) is said to be blocked iff $tf_k^{(ou)}$=‘1’ and all time-frames in the set $\{tf_{k-i}^{(in)} | i = 0, 1, .., z.\}$ are busy. We use a symbol ‘1_b’ to denote the blocked time-frame, i.e., $tf_k^{(ou)}$=‘1_b’.

Examples of z-schedulable and z-blocked time-frames are in Fig. 4.2.

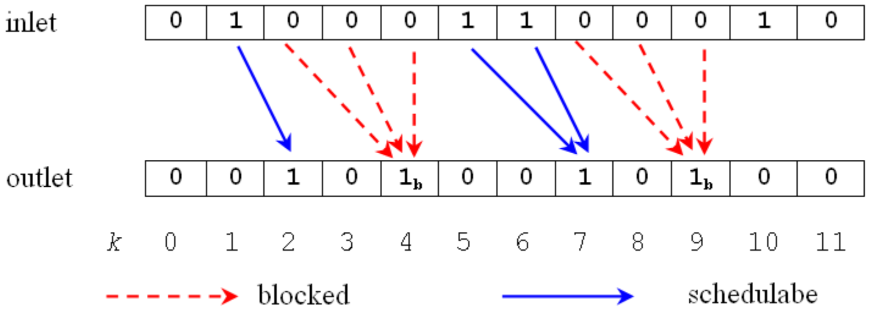


Figure 4.2: Illustration when $K = 12, a = 4, z = 2$: $tf_4^{(ou)}$ and $tf_9^{(ou)}$ are blocked; $tf_2^{(ou)}$, $tf_7^{(ou)}$ are schedulable.

<u>Problem statement</u> — Given a pair of inlet and outlet of a strictly non space-blocking switch operating under z-forwarding scheme, we aim at deriving the probability $p_{z\mathrm{F}}$, the time-blocking probability that all available time-frames of the outlet are found blocked (from the inlet), given the load specified by (K, b).

Let C_{blk} be the number of combinations made by both the inlet and the outlet such that all the a available time-frames of the outlet are found blocked. Let C_{total} be the total number of combinations made by both the inlet and the outlet. The time-blocking probability is the ratio between C_{blk} and C_{total}:

$$p_{z\mathrm{F}} = \frac{C_{blk}}{C_{total}} \tag{4.2.1}$$

In the following, we present a direct computation of the time-blocking probability for all z-forwarding schemes by deriving combinatorial numbers C_{blk} and C_{total}.

4.3 Analysis approach

4.3.1 Run, run-length and blocked positions

Run and run-length

The discussion is focused on different dispositions of the a symbols ‘`1`’ and the b symbols ‘`0`’ in the inlet. A *run* is defined as a group of the same symbols that are positioned consecutively. For examples, runs of `0`’s are ‘`0`’, ‘`00`’, ‘`000`’ and so on.

A number of symbols composing a run is its *run-length*. The minimum run-length for a meaningful run is 1. In between two adjacent runs of `0`’s there is one run of `1`’s, and vice versa.

Runs in a cyclic arrangement

Because of the periodic nature of TDS/FλS, the last time-frame in a time-cycle, for example, can be delayed until first time-frame positions in the next time-cycle if $z > 0$. It means that the last time-frame and the first time-frame are positioned consecutively. This implies that the arrangement of the a symbols ‘`1`’ and the b symbols ‘`0`’ into K time-frame positions reoccur in a cyclical manner. Therefore, in each arrangement the number of runs of `0`’s and the number of runs of `1`’s are equal, excluding the trivial cases of all zeros and all ones.

For instance, in a cyclical arrangement of 4 symbols ‘`1`’ and 8 symbols ‘`0`’ in the inlet shown in Fig. 4.2, there are 3 runs of `1`’s and 3 runs of `0`’s. One special run of `0`’s whose run-length is 2 composing by $tf_0^{(ou)}$=‘`0`’ and $tf_{11}^{(ou)}$=‘`0`’.

Runs in a linear arrangement

In this case it is assumed that the cycle is open, and therefore, in the inlet shown in Fig. 4.2, under the linear arrangement view, there are 4 runs of 0's and 3 runs of 1's.

Note that all notations for runs and run-lengths presented in this chapter are defined for cyclical arrangements. However, in some parts of the combinatorial analysis, it is clearly indicated that the counting is of runs under the linear arrangement (e.g., in Section 4.4.1). We also note that for all linear arrangements discussed, the cycle is broken at the first time-frame position tf_0.

Blocked positions

Observe that for a given z-forwarding scheme, an arrangement of the a available time-frames and the b busy time-frames in the inlet, generates some positions, such that if an available time-frame in the outlet $tf_k^{(ou)}$='1' is "positioned" beneath anyone of these positions, it becomes blocked, i.e., $tf_k^{(ou)}$='1_b'. Thus, such positions are called blocked positions. In order to highlight the concept of blocked position, which is important in the following analysis, let's consider the following examples:

- For $z = 0$ (i.e., the IF scheme), any arrangement in the inlet generates b blocked positions. Obviously, if an outlet's available time-frame is 'positioned beneath' a busy time-frame of the inlet, it is blocked since $z = 0$.

- For $z = 1$, the content of a time-frame can be delayed at most one time-frame duration prior to being forwarded. Fig. 4.3 shows how blocked positions are generated. In fact, for every pair of adjacent '00' symbols the right symbol generates a blocked position. Consequently, if there are $l > 1$ consecutive 0's, then there are $l-1$ blocked positions.

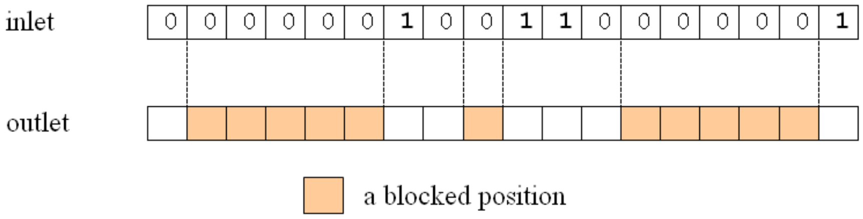


Figure 4.3: Illustration of blocked positions when $z = 1$, given a sample combination of the inlet.

- For $z = 2$, a content of time-frame can be delayed at most two time-frame durations. Fig. 4.4 shows how blocked positions are generated in this case. Only runs whose run-length is greater than two (since $z = 2$), such as, '`000`', '`0000`' and so on, generate blocked positions. Consequently, if there are l consecutive `0`'s and $l > 2$, then there are $l - 2$ blocked positions.

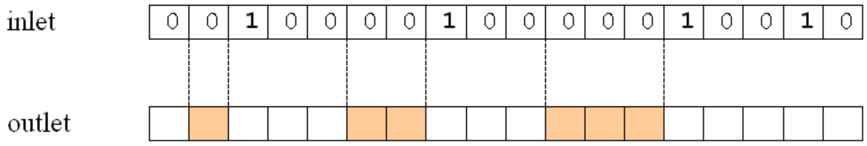


Figure 4.4: Illustration of blocked positions when $z = 2$, given a sample combination of the inlet.

From above illustrations, it is trivial to conclude that:

- The number of blocked positions generated by a given arrangement (of available time-frames and busy time-frames) in the inlet depends on a specific z-forwarding scheme and a given load (K, b) of the inlet.

- For a run of `0`'s, there is a relation between the number of blocked positions generated, its run-length and z. Let l_i be the run-length of run i of `0`'s. Let x_i be the number of blocked positions generated by run i, then:

$$x_i = \begin{cases} l_i - z & \text{,if } \ l_i \geq z \\ 0 & \text{,otherwise} \end{cases} \tag{4.3.1}$$

We are interested in run i such that $l_i \geq z$.

The bound of the number of blocked positions

Given an arrangement in the inlet, let x be the total number of blocked positions generated from all runs of `0`'s, the following is shown. Note that x is non-negative integer, $x \subset \mathbb{Z}^+$.

Lemma 4.3.1 (The bound of x)**.** *For a given load* (K, b)*,* x *is bounded by:*

$$b - za = x_{min} \leq r \leq x_{max} = b - z \tag{4.3.2}$$

Proof. From (4.3.1), we yield $x_{max} = b - z$ when all the b '`0`' symbols form a single run in the inlet, which is obviously the longest possible run.

To compute x_{min}, we further observe that, in a cyclical arrangement, a symbols of '`1`' can split maximum a runs of `0`'s, where every run has the same length of z (i.e., $l_i = z$ for all i) such that no blocked position is generated according to (4.3.1). The remaining number of symbols '`0`' is $(b - za)$. Since no more run of `0`'s can be formed due to running out of symbols '`1`' to split them. Thus, placements of remaining '`0`' symbols finally generate blocked positions. Therefore, $x_{min} = (b - za)$. □

The result (4.3.2) is used to derive the generic form of time-blocking probability in subsection 4.3.2.

The blocked position concept and the definition of blocked time-frame (*Def.* 4.2.3) imply that the time-blocking case happens when all available time-frames of the outlet are 'placed in' blocked positions. Thus, we define each of these occurrences as a time-blocking case.

4.3.2 The general form of the time-blocking probability

For a given value of x satisfying (4.3.2), let $C(x)$ be the number of arrangements found only in the inlet such that each of these arrangements generate exactly x blocked positions. We derive $C(x)$ later in Section 4.4 (for $z = 1$) and in Section 4.5 (for the general case). In essence, counting $C(x)$ is the most difficult part of our analysis and it is done in the subsequent two sections. Given $C(x)$, we have the following result:

Theorem 4.3.2. — *For a stand-alone switch, the time-blocking probability for the general z-forwarding scheme, p_{zF} is given by:*

$$p_{zF} = \sum_{x=\mathsf{max}\{a,(b-za)\}}^{b-z} C(x)\binom{x}{a} \Big/ \binom{K}{b}^2 \tag{4.3.3}$$

Proof. Given x blocked positions generated by the inlet, the number of ways to arrange all a available time-frames of the outlet into blocked positions so that a time-blocking occurs is $\binom{x}{a}$. Thus, the subtotal number of combinations, denoted as C_{sub}, generated by both the inlet and the outlet such that a time-blocking happens is given by:

$$C_{sub} = C(x)\binom{x}{a}$$

Note that if $x < a$, then $\binom{x}{a} = 0$. Thus, we only consider $x \geq a$ (i.e., a case where a time-blocking occurs). From Lemma 4.3.1, observe that:

- if $(b - za) \geq a \Leftrightarrow K \geq (z + 2)a$ then for any combination in the inlet, we have $x_{min} = (b - za) \geq a$.
- if $(b-za) < a \Leftrightarrow K < (z+2)a$ then for some x such that $b-za \leq x < a$, we are not interested in. Thus we set $x_{min} = a$.

Combined with (4.3.2) we have the range of meaningful x for computing time-blocking probability:

$$\mathsf{max}\{a, (b - za)\} \leq x \leq b - z. \tag{4.3.4}$$

The sum of C_{sub} over all meaningful x yields C_{blk}:

$$C_{blk} = \sum_{x=x_{min}}^{x_{max}} C_{sub} = \sum_{x=\mathsf{max}\{a,(b-za)\}}^{b-z} C(x)\binom{x}{a}$$

Meanwhile, total numbers of combinations at the inlet and at the outlet are computed as $\binom{K}{b}$ for each inlet and outlet. Thus, we have C_{total}:

$$C_{total} = \binom{K}{b}\binom{K}{b} = \binom{K}{b}^2$$

Therefore, we obtain $p_{z\mathrm{F}}$ as in (4.3.3). □

Theorem 4.3.2 shows how $p_{z\mathrm{F}}$ is computed once we have $C(x)$. However, the most nontrivial task is at the derivation of $C(x)$, the number of combinations in the inlet generating exactly x blocked positions. The computation is more complicated for z-forwarding schemes such that $z > 1$. Thus, in the next section, we first derive $C(x)$ for $z = 1$. The derivation of $C(x)$ for the general z-forwarding case is presented in Section 4.5.

4.4 Analysis for 1-forwarding case

We separate the analysis of the 1-*forwarding* scheme from the general case, because its simpler mathematics allows for descriptions and explanations that will help in deriving the general case. 1-*forwarding* means there is a single position in the buffer: $z = 1$.

Let u denote the number of runs of `0`'s. For $z = 1$ all runs satisfy $l_i \geq z$. Summing equation (4.3.1) over all runs yields:

$$\sum_{i=1}^{u} x_i = \sum_{i=1}^{u} (l_i - z) = \sum_{i=1}^{u} l_i - uz$$

Since $\sum_{i=1}^{u} x_i = x$ (total number of blocked positions) and $\sum_{i=1}^{u} l_i = b$ (total number of symbols '`0`'), the equation above becomes simply:

$$u = b - x \tag{4.4.1}$$

Eq. (4.4.1) holds only for $z = 1$, and it is the reason why this case can be treated differently from the general one. In this case the computation of $C(x)$ can be done in two different ways. The first one, considering a linear disposition of the symbols, gives the result with a problem decomposition in form of summation. The second one, which will be used also in the general case, considers the cyclic disposition of the symbols and gives the results in form of a multiplicative decomposition that, however, counts the number of possible patterns u times, so that the final result must be divided by u.

4.4.1 Additive decomposition

In a non-cyclic perspective, the patterns into which the b symbols '`0`' and the a symbols '`1`' in an inlet can be disposed falls into one of the following cases:

Case 1: the first and the last symbol of the cycle are different, implying that there are u runs of `0`'s and u runs of `1`'s. Case 1 has two obvious and identical (from the combinatorial point of view) sub-cases: the first symbol is '`0`' and the last one is '`1`', or vice versa.

Case 2: both the first and the last symbol of the cycle are '`1`'. so that there are u runs of `0`'s, and $(u + 1)$ runs of `1`'s.

Case 3: both the first and the last symbol of the cycle are '`0`', so that there are $(u + 1)$ runs of `0`'s and u runs of `1`'s.

It is easy to see that the three cases above form a partition of the set of the dispositions, and this is valid for any given x, so that $C(x)$ can be computed as the sum of the three cases.

Lemma 4.4.1 ($C(x)$ for the case $z = 1$)**.** *For $z = 1$, $C(x)$ is given by:*

$$C(x) = \frac{K}{u}\binom{a-1}{u-1}\binom{b-1}{u-1} \tag{4.4.2}$$

where x is implicit in u as in (4.4.1).

Proof. We sum all the combinations of the three cases defined above, that, forming a partition, contain all and only the distributions of interest, i.e.,

$$C(x) = C_{\text{case 1}} + C_{\text{case 2}} + C_{\text{case 3}}$$

Consider case 1: the number of dispositions is the product of the following terms:

- the number of dispositions of the a symbols '`1`' into u distinct runs such that there will be at least one symbol per run. Basic combinatorics (see Chapter 2 of [68]) yields $\binom{a-1}{u-1}$.
- the number of dispositions of the b symbols '`0`' into u distinct runs such that there will be at least one symbol per run, which is $\binom{b-1}{u-1}$.
- a multiplicative factor of 2 reporting of the two subcases.

Thus, we obtain $C_{\text{case 1}}$:

$$C_{\text{case 1}} = 2\binom{a-1}{u-1}\binom{b-1}{u-1}$$

Following the same counting methods we obtain:

$$C_{\text{case 2}} = \binom{a-1}{u}\binom{b-1}{u-1} = \frac{a-u}{u}\binom{a-1}{u-1}\binom{b-1}{u-1}$$

$$C_{\text{case 3}} = \binom{a-1}{u-1}\binom{b-1}{u} = \frac{b-u}{u}\binom{a-1}{u-1}\binom{b-1}{u-1}$$

Summing together the three cases leads to equation (4.4.2) with trivial algebra. □

Substitute (4.4.2) into (4.3.3), replacing $u = b - x$, $z = 1$ and $a = K - b$, we yield the time-blocking probability for the 1-*forwarding* scheme, $p_{1\mathrm{F}}$:

$$p_{1\mathrm{F}} = \frac{\sum_{x=\mathsf{max}\{a,(b-a)\}}^{b-1} \frac{K}{b-x} \binom{K-b-1}{b-x-1} \binom{b-1}{b-x-1} \binom{x}{K-b}}{\binom{K}{b}^2} \tag{4.4.3}$$

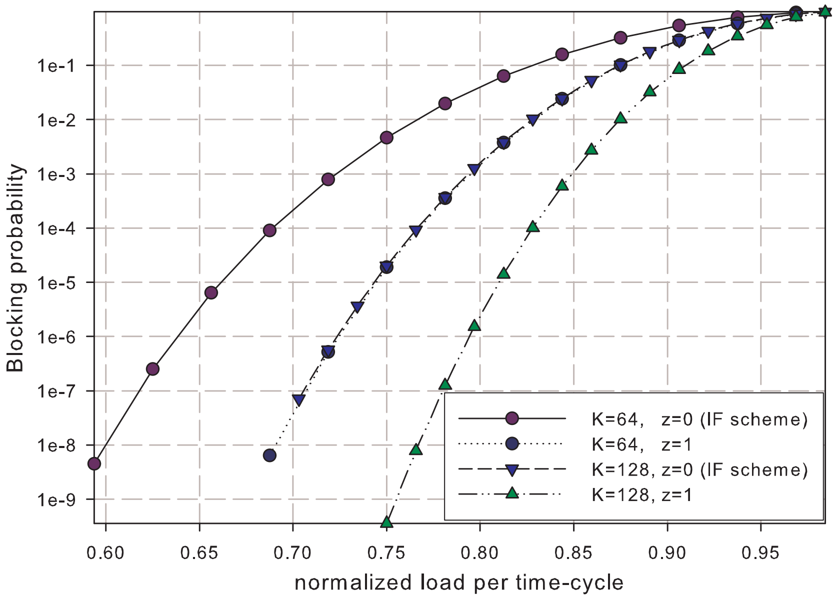


Figure 4.5: Examples of numerical result for $z = 0$ and $z = 1$.

Fig. 4.5 shows numerical examples obtained from (4.4.3) for 1-*forwarding* scheme and results for 0-*forwarding* (IF) scheme (reported later in subsection 4.5.2). In the graph, numerical results for $(K = 64, z = 1)$ and for $(K = 128, z = 0)$ are very close to each other. However, a quick investigation on the actual numbers shows that they are not identical, but only very similar. This can be explained as following. Once one buffer $(z = 1)$ is used, the effect is as comparable as we double both the number of free and the number of busy time-frames in a cycle and not use the buffer $(z = 0)$. Meanwhile, the reverse (i.e., moving from $K = 128$ to $K = 64$ and using $z = 1$) does not hold.

4.4.2 Direct factorization

Considering the space (i.e., the cycle) where the time-frames are disposed in a circle, where the last time-frame is adjacent to the first time-frame, a direct factorization of the counting problem is possible, counting all the possible dispositions of the runs of '`1`' and '`0`', however this leads to counting all the patterns u times, so that the final result must be divided by u. Since this is the technique we use in the general case, we do not repeat it here, but refer to the next section.

4.5 Analysis for general z-forwarding case

Equation (4.4.1) holds only for $z = 1$, since this is the only case where all the runs of `0`'s whose individual run-length satisfies $l_i \geq z$. If (4.4.1) is not valid, there is not a unique relationship between x, u and b, and the scenario becomes more complex.

When the condition $l_i \geq z$ is not satisfied by all runs of `0`'s, these runs are divided into two subsets: those that leads to blocking positions and those that do not. Let's introduce the following notations, that will be used in deriving the results later in the section:

- $\mathbb{U}$ denotes the set of all runs i (of `0`'s) such that run-lengths satisfy $l_i \geq z$. Only runs with $l_i \; > \; z$ produces $x_i \geq 1$, i.e., blocking positions.
- $u = \|\mathbb{U}\|$.
- b_u denotes the number of symbols '`0`' covered by all runs in $\mathbb{U}$.
- $\mathbb{V}$ denotes the set of all runs i (of `0`'s) such that run-length $l_i < z$. That is, no run in $\mathbb{V}$ produces blocked positions.
- $v = \|\mathbb{V}\|$.

- b_v denotes the number of symbols '`0`' covered by all runs in $\mathbb{V}$.
- $\mathbb{A}$ denotes the set of all runs of `1`'s. Thus, $u + v = \|\mathbb{A}\|$ because of cyclical arrangement.

From the above definitions it is immediately clear that the 1-*forwarding* case is the special case where $\mathbb{V} \ = \emptyset$. Table 4.1 summarizes the notation introduced above, together with the others already used elsewhere in this chapter.

One of the key differences between the 1-*forwarding* case and the general case analyzed is the presence of non-valid (u, v) couples, i.e., values of u and v that do not satisfy all the constraints of the problem. This fact forces us to separately count for all and any the valid (u, v) couples, while the simple relation (4.4.1) allowed for a unique computation. Given this additional complexity, partitioning the set of patterns as we did for $z = 1$ becomes excessively cumbersome, so we resort to the analysis considering the cyclic disposition of time-frames.

We now define some general bounds for the parameters of the problem, that will be the upper and lower limits of the indexes used in the formulae derived afterwards. Summing eq. (4.3.1) over all runs in $\mathbb{U}$ yields (with some simple algebra manipulations):

$$0 < b_u = x + zu \leq b \tag{4.5.1}$$

The number of symbols b_v is given by:

$$b_v = b - b_u = b - x - zu \geq 0 \tag{4.5.2}$$

While by construction, we have:

$$1 \leq u + v \leq a \tag{4.5.3}$$

Table 4.1: Summary of the notation used for the general case.

Notation	Explanation
a	Number of symbols '`1`' (i.e. number of free time-frames)
b	Number of symbols '`0`' (i.e. number of busy time-frames)
z	Number of buffers, $1 \leq z < K$
$p_{z\mathrm{F}}$	Blocking probability under z-forwarding scheme
l_i	Run-length of run i
x_i	Number of blocked positions generated by run i
x	Total number of blocked positions generated by all runs of `0`'s in a given arrangement
$\mathbb{U}$	Set of all runs of `0`'s such that $l_i \geq z$
u	Number of runs in $\mathbb{U}$, $u = \|\mathbb{U}\|$
b_u	Total number of symbols '`0`' occupied by all runs in $\mathbb{U}$
$\mathbb{V}$	Set of all runs of `0`'s such that $1 \leq l_i < z$
v	Number of runs in $\mathbb{V}$, $v = \|\mathbb{V}\|$
b_v	Total number of symbols '`0`' occupied by all runs in $\mathbb{V}$
$\mathbb{A}$	Set of all runs of `1`'s, $u + v = \|\mathbb{A}\|$
$C(u,v)$	Number of combinations that generate exact r blocked positions, given a valid pair of (u,v)
$C(x)$	Total number of combinations in the outlet that generates exact r blocked positions, for all valid pairs of (u,v)

Lemma 4.5.1. *The size of* $\mathbb{U}$ *is bounded by:*

$$1 = u_{min} \leq u \leq u_{max} = \mathit{min}\Big\{\Big\lfloor\frac{b-x}{z}\Big\rfloor, a\Big\} \tag{4.5.4}$$

Proof. When there is only one run of `0`'s, we have $u_{min} = 1$.

From (4.5.1) we have $u = \frac{b_u - x}{z}$ and $u = u_{max}$ iff $b_u = b$. $b_u = b$ implies that all symbols '`0`' of the inlet are in runs belonging to $\mathbb{U}$ and $\mathbb{V}$=$\emptyset$, $v = 0$. Setting $v = 0$ in (4.5.3) yields $u \leq a$ so that $u_{max} \leq \mathsf{min}\{\lfloor \frac{b-x}{z} \rfloor, a\}$.

Note that $u = 0$ is not considered since it means there is one run of `0`'s with length smaller than z, or $b < z$. In this case we do not have time-blocking. □

Lemma 4.5.2. *For* $1 < z < K$*, the size of* $\mathbb{V}$ *is bounded by:*

$$\lceil \frac{b_v}{z-1} \rceil = v_{min} \leq v \leq v_{max} = \mathit{min}\{(a-u), b_v\} \tag{4.5.5}$$

Proof. We have $v = v_{min} = \lceil \frac{b_v}{z-1} \rceil$ when all runs in $\mathbb{V}$ have the maximum allowed length $l_i = (z-1)$.

The upper bound depends on the ratio between b_v and the number of symbols '`1`' not used to separate runs in $\mathbb{U}$ that can separate runs in $\mathbb{V}$. That is $(a-u)$.

- if $b_v > (a-u)$ then $v_{max} = (a-u)$.
- if $b_v \leq (a-u)$, we can split all b_v symbols '`0`' in runs of length one, so that $v_{max} = b_v$.

Therefore, $v_{max} = \mathsf{min}\{(a-u), b_v\}$. □

4.5.1 Deriving the combinatorial number $C(x)$

Equations (4.5.1)-(4.5.5) define the limits of (u, v) for a given value of blocking positions x satisfying (4.3.4).

Recall that in a time-cycle, the last time-frame tf_{K-1} is considered to be adjacent to (on the left) the first time-frame tf_0, so that no real "beginning" or "ending" of the cycle exist and no run is "split" as it happens considering the linear disposition with the cycle's beginning and ending.

Theorem 4.5.3. *Given a valid pair of (u, v), the number of patterns, denoted as $C(u, v)$, that exactly generates x blocked positions is:*

$$C(u, v) = \frac{KC_{uv}C_aC_{b_u}C_{b_v}}{u+v} \tag{4.5.6}$$

where r is implicit in b_u, b_v, u, v given the relations (4.5.1)-(4.5.5). The factors C_{uv}, C_a, C_{b_u}, and C_{b_v} are defined in (4.5.7)-(4.5.10) of the proof, respectively.

Proof. The goal is computing the total number of possible patterns distributing the b_u symbols '`0`' into $\mathbb{U}$ runs, the b_v symbols '`0`' into $\mathbb{V}$ runs, and the a symbols '`1`' into runs in $\mathbb{A}$. To obtain this we show that there exists a factorization of the problem that counts $(u + v)$ times the total number of patterns. The factorization starts counting the possible dispositions of the runs themselves given u and v, then counts the dispositions of the symbols in the runs in different sets $\mathbb{A}$, $\mathbb{U}$, and $\mathbb{V}$, finally all possible K cyclic shifts of the above patterns are counted showing that each pattern is thus counted exactly $(u + v)$ times.

C_{uv}: number of dispositions of the u runs in $\mathbb{U}$ within the total number of possible runs $(u + v)$ of $\mathbb{U} \cup \mathbb{V}$. Trivial combinatorics yields:

$$C_{uv} = \binom{u+v}{u} \tag{4.5.7}$$

C_a: number of dispositions of the a symbols '`1`' into the $(u+v)$ distinct runs such that each run has at least one symbol. Basic combinatorics [68] yields:

$$C_a = \binom{a-1}{u+v-1} \tag{4.5.8}$$

C_u: number of dispositions of the b_u symbols '`0`' into the u distinct runs such that each run has at least z symbols. The counting method consists in first placing $(z-1)$ symbols into every run $\in \mathbb{U}$, then distributing the remaining $b_u - (z-1)u$ symbols in all the u runs such that each run has at least one symbol. Using the same combinatoric result used for C_a we have:

$$C_{b_u} = \binom{b_u - (z-1)u - 1}{u-1} \tag{4.5.9}$$

C_{b_v}: number of dispositions of the b_v symbols '0' into the v distinct runs such that each run has at least one symbol and no run has more than $(z-1)$ symbols:

$$C_{b_v} = \begin{cases} \sum_{i=0}^{v} (-1)^i \binom{v}{i} \binom{b_v - i(z-1) - 1}{v-1} & \text{if } v > 0 \\ 1 & v = 0 \text{ or } b_v = v \end{cases} \tag{4.5.10}$$

Deriving (4.5.10) is a long and cumbersome and we present it in Appendix A.

The time-cycle boundary can be at any time-frame, thus there are K possible shifts for each disposition counted so far. The total number of possible dispositions given a valid pair (u, v) is then $KC_{uv}C_aC_{b_u}C_{b_v}$. However, each combination is actually counted $(u+v)$ times and the number $KC_{uv}C_aC_{b_u}C_{b_v}$ must be divided by $(u+v)$ to eliminate multiple countings, thus resulting in (4.5.6).

The proof of the multiple counting is presented in Appendix B to streamline reading. The rationale is that each of the $C_{uv}C_aC_{b_u}C_{b_v}$ can be transformed into exactly $u+v$ other patterns by shifting it circularly of an appropriate number of time-frames. □

Theorem 4.5.4. *The total number of dispositions $C(x)$ that generates exact x blocked positions is given by:*

$$C(x) = \sum_{u=1}^{min\{\lfloor \frac{b-x}{z} \rfloor, a\}} \left\{ \sum_{v=\lceil \frac{b_v}{z-1} \rceil}^{min\{a-u, b_v\}} C(u,v)\big|_{u+v \leq a} \right\} \tag{4.5.11}$$

Proof. A pair of (u, v) is valid iff u and v jointly satisfy (4.5.3), (4.5.4) and (4.5.5). Since $C(u, v)$ is computed through eq. (4.5.6) for any valid pair of (u, v), the sum of $C(u, v)$ over all valid pairs of (u, v) leads to the total number of dispositions $C(x)$ in the outlet that generates exact x blocked positions. □

Finally, summing (4.5.11) over all valid values of x then multiplying with $\binom{x}{a}$ fulfills the numerator of (4.3.3) and finally the closed form solution of the time-blocking probability for a single node case.

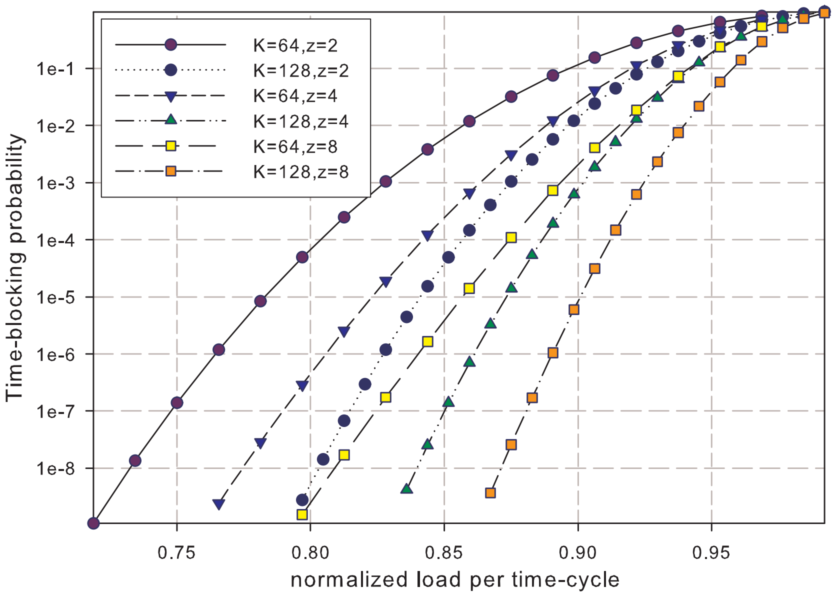


Figure 4.6: Examples of numerical result for $z > 1$.

Examples of numerical results for various z and K values are shown in Fig. 4.6. One interesting property is the reduction in time-blocking probability as K increases for a given normalized load. While it was an easy prediction that time-blocking probability would decrease exponentially increasing the buffering capability z, a similar decrease simply increasing K was not an easy prediction. The phenomenon is similar to the classic result that gives smaller and smaller call blocking probability for a given load as the granularity of the calls decreases.

4.5.2 Sanity checks

The result for the general case presented above is rather complex and might be appalling. Here we discuss some limit cases where the exact result can be easily obtained with heuristic reasoning.

Immediate forwarding

This is the case when $b \leq K-1$, $z=0$. For any combination in the outlet, we always have $x = x_{min} = x_{max} = b$, and $C(x) = \binom{K}{b}$. Thus, the equation (4.3.3) shrinks to:

$$p_{0F} = \binom{K}{b}\binom{b}{a} \Big/ \binom{K}{b}^2 = \binom{b}{a} \Big/ \binom{K}{b} \tag{4.5.12}$$

Trivial combinatorics also reache the same result.

Non-immediate forwarding with $z = 1$

This is the case when $b \leq K-1$, $z=1$. For $z=1$, we have $\mathbb{V}$=$\emptyset$ or $b_v = 0$, and $b_u = b = x + u$. Letting $v = 0$ in the formulae of theorem 4.5.3 yields

$$C_{(u,v=0)} = \frac{K}{u}\binom{a-1}{u-1}\binom{b_u-(z-1)u-1}{u-1} \tag{4.5.13}$$

which, as claimed in Section 4.4.2, is equivalent to (4.4.2) remembering that $z = 1 \Leftrightarrow b = b_u;\ v = 0;\ x = b - u$.

Arbitrary (full) time-frame forwarding

This is the case when $b \leq K-1$, $z = K-1$. Replacing $z = K-1$ into (4.3.2) yields $x_{max} = b - z = b - (K-1) \leq 0$ since $b \leq K-1$. This implies that there is no single combination where we can find $x \geq a$, or $p_{z\mathrm{F}}$= 0 for $z = K-1$. The intuition of zero blocking for this special case is confirmed.

4.6 Load assumption relaxation

The results presented in previous sections can be modified to accommodate the different load assumptions where the load of the inlet and the outlet are different. Let (K, b_i) and (K, b_o) denote the load for the inlet and for the outlet respectively. And let a_i and a_o be the number of available time-frames at the inlet and the outlet, respectively. Thus, eq.(4.3.4) becomes:

$$\mathsf{max}\{a_i, (b_i - za_i)\} \leq x \leq b_i - z. \tag{4.6.1}$$

and the modified version of (4.3.3) is:

$$p_{z\mathrm{F}} = \left\{ \sum_{x=\mathsf{max}\{a_i, b_i - za_i\}}^{b_i - z} C(x) \binom{x}{a_o} \right\} \Big/ \left\{ \binom{K}{b_o} \binom{K}{b_i} \right\} \tag{4.6.2}$$

Note that $C(x)$ is given by:

$$C(x) = \sum_{u=1}^{\mathsf{min}\{\lfloor \frac{b_i - x}{z} \rfloor, a_i\}} \left\{ \sum_{v=\lceil \frac{b_v}{z-1} \rceil}^{\mathsf{min}\{a_i - u, b_v\}} C(u, v)\big|_{u+v \leq a_i} \right\} \tag{4.6.3}$$

where

$$b_v = b_i - b_u = b_i - x - zu \geq 0 \tag{4.6.4}$$

Eq. (4.5.1), (4.5.7), (4.5.8), (4.5.9), (4.5.10) and (4.5.6) are reused without any change.

4.7 Discussions

The problem of time-blocking probability in TDS switches has been formulated and analyzed in this chapter. It has been shown that time-blocking is greatly reduced when a small number of (optical) buffers (used for enabling scheduling delays that are measured in time-frames) are added to each inlet.

Consequently, the main result of this chapter is the time-blocking probability analysis as a function of the number of possible scheduling delay z and the loads (K, b_i) and (K, b_o).

The time-blocking probability analysis in this chapter is a novel combinatorial approach. The main assumption is that all possible load combinations are equally likely. Namely, if a combination is defined by the distribution on b busy time-frames, out of K possible time-frames in each time-cycle, in a given inlet and a given outlet, then all such possible combinations are equally likely. Some concrete numerical examples presented in the chapter clearly illustrate that only a small number of buffers (or short scheduling delay) is required to obtain low blocking probability under high load conditions.

However, the analysis presented in this chapter is suitable to comprehend time-blocking behaviors only for a single strictly non space-blocking switch. Thus, in the next chapter, we extend the blocking analysis for multiple hops with both immediate forwarding (IF) and non-immediate forwarding (NIF).

Chapter 5

Time-blocking: multi hop cases

In chapter 4, we analyze the time-blocking behavior of a stand-alone strictly non-space blocking FλS switch. In fact, when one single switch is considered, it is equivalent to a route path of 2 hops (i.e., source $\rightarrow$ one-switching-node $\rightarrow$ destination).

Obviously, more than two-hop cases come into reality more often. In this chapter, we aim at analyzing the time-blocking probability when multiple consecutive switches are connected to form a multiple-hop path on which a FλP can be established. More specifically, the time-blocking along the route consisting of H hops will be analyzed. The load assumption remains as presented in the previous chapter, i.e., all combinations are equally likely given b (the number of busy time-frames at hop h) out of total K time-frames of a time-cycle.

5.1 Multi-hop scenarios

A path from a source to a destination node is connected by $H - 1$ consecutive switches. Switches are similar in terms of being strictly non space blocking and having the same z-forwarding scheme. Hops and switches are indexed as shown in Fig. 5.1.

Without the loss of generality, a delay between hops is disregarded (i.e,

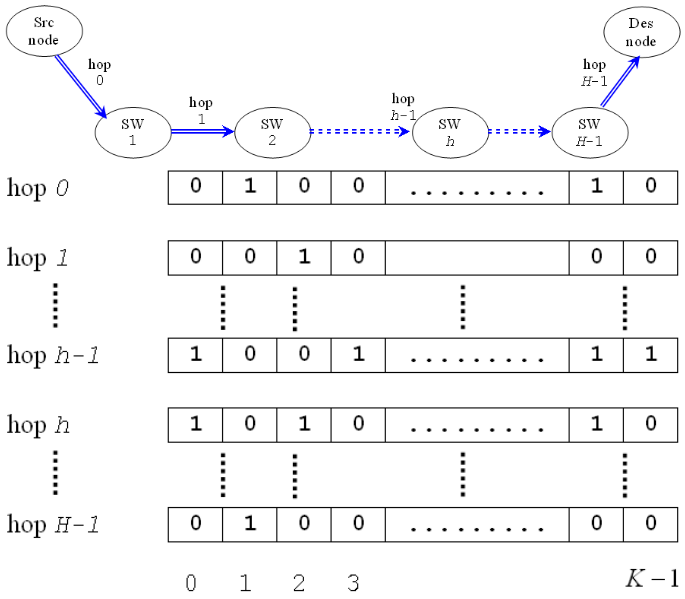


Figure 5.1: H hops from source (src) node to destination (des) node, $H-1$ switches in between. There are K time-frames per hop and b out of K are busy.

the time-cycle of every hop is aligned to each other). Recall that symbol '1' represents an available time-frame, and symbol '0' represents a busy time-frame. For every hop, there are b symbols '0' and a symbols '1'. All symbols are uniformly distributed.

We aim at finding the probability that after H hops there is no *schedule* to form a FλP from source to destination. Recall that a *schedule* is defined as a possible allocation of time-frames following z-forwarding schemes at all switching nodes along a route of H hops so that a FλP[1] can be setup.

5.1.1 Definitions and notations

Throughout the paper, for any positive integer z, let $[z]$ denote the set of integers $\{0, 1, 2, \cdots, z\}$. Let $tf_k^{(h)}$ denote time-frame k at hop h, we have following definitions, which are the extensions of Def. 4.2.2 and Def. 4.2.3.

[1] Note that we consider a FλP that requires only one time-frame at a time.

Definition 5.1.1 (A schedulable time-frame at hop h). — A time-frame k at hop h, $tf_k^{(h)}$ is said to be schedulable iff $tf_k^{(h)}$=‘$\texttt{1}$’ (i.e., available) and at least one time-frame in the set $\{tf_{k-i}^{(h-1)}|i \in [z]\}$ is available (i.e., $tf_{k-i}^{(h-1)}$=‘$\texttt{1}$’ for at least one $i \in[z]$).

Note that for the fist hop, $h = 0$, all available time-frames are schedulable.

Definition 5.1.2 (A blocked time-frame at hop h). — A time-frame k at hop h, $tf_k^{(h)}$ is said to be blocked, $tf_k^{(h)}$=‘$\texttt{1}_b$’, iff $tf_k^{(h)}$=‘$\texttt{1}$’ and all time-frames in the set $\{tf_{k-i}^{(h-1)}|i \in [z]\}$ are either blocked or busy (i.e., $tf_{k-i}^{(h-1)}$=‘$\texttt{1}_b$’ or $tf_{k-i}^{(h-1)}$=‘$\texttt{0}$’ for all $i \in[z]$).

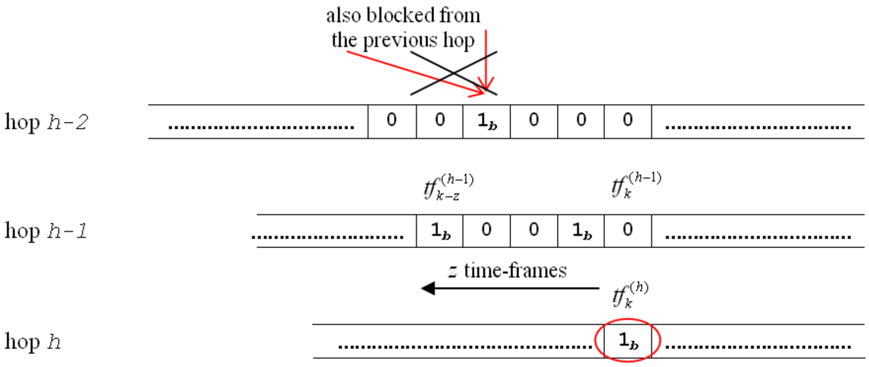


Figure 5.2: Illustration of blocked time-frames ‘$\texttt{1}_b$’.

Blocked time-frames are useless in the process of searching for a schedule for setting up a FλP. An illustration of a blocked time-frame (Def. 5.1.2) is shown in Fig. 5.2. Note that the Def. 5.1.2 is repetitively applied for all available time-frames of all hops excluding the first hop. There is no blocked time-frames at the first hop since any available time-frame can start a schedule to form an FλP. In other words, $tf_k^{(0)}$=‘$\texttt{1}_b$’ does not exist.

Definition 5.1.3. — A time-frame *position* in a time-cycle at hop h is said to be *unblocked* iff it is in a forwarding range of one schedulable time-frame of the previous hop $h - 1$.

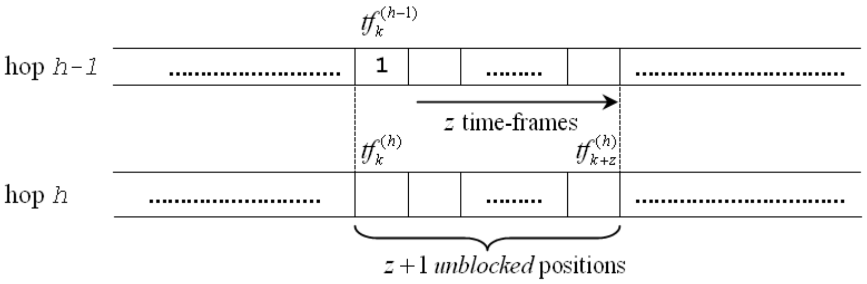


Figure 5.3: The *unblocked* positions.

In other words, if $tf_k^{(h-1)}$=‘1’, then all time-frame positions $tf_{k+i}^{(h)}$, $i \in [z]$, are *unblocked*. If an available time-frame ‘1’ is placed in an *unblocked* position, it becomes a schedulable time-frame.

Contrary, there are also *blocked* positions. By Def. 5.1.2, if $tf_{k-i}^{(h-1)}$=‘0’ or $tf_{k-i}^{(h-1)}$ = ‘1_b’ for all $i \in [z]$, then $tf_k^{(h)}$ is a *blocked* position. If an time-frame ‘1’ is placed in a *blocked* position, it becomes blocked, ‘1_b’.

Remark 5.1.1. At hop h, let x be the number of *blocked* positions,and y be the number of *unblocked* positions, then it is obvious that:

$$x + y = K \tag{5.1.1}$$

At hop $h-1$ and hop h, let:

- α_{h-1} denotes the number of schedulable time-frames ‘1’ at hop $h-1$.
- β_{h-1} denotes the number of *unblocked* positions generated by hop $h-1$ for hop h.
- α_h denotes the number schedulable time-frames ‘1’ at hop h.
- $Pr(\alpha_h = \tilde{a})$ denotes the probability that $\alpha_h = \tilde{a}$.
- $Pr(\alpha_{h-1} = \hat{a})$ denotes the probability that $\alpha_{h-1} = \hat{a}$.

Remark 5.1.2. For the first hop $h = 0$, we have:

$$Pr(\alpha_0 = \tilde{a}) = \begin{cases} 1.0 & \text{, if} \quad \tilde{a} = a_0 \\ 0.0 & \text{, otherwise} \end{cases} \tag{5.1.2}$$

Remark 5.1.3. In general at hop $h-1$, $\hat{a}$ is bounded by:

$$0 \leq \hat{a} \leq a \tag{5.1.3}$$

Remark 5.1.2 and Remark 5.1.3 are trivial by constructions.

Note that if α_{h-1}=0, then β_{h-1}=0. For α_{h-1}=$\hat{a}$> 0, on one extreme case, when all $\hat{a}$ schedulable time-frames at hop $h-1$ form a unique run, then we obtain minimum value of *unblocked* positions:

$$y_{min} = \mathsf{min}\{\hat{a} + z, K\}.$$

On the other extreme case, when each schedulable time-frame forms one run, and two consecutive schedulable time-frames are split by an interval of at least z time-frames, then we obtain the maximum value of *unblocked* positions:

$$y_{max} = \mathsf{min}\{(z+1)\hat{a}, K\}.$$

Thus, we have the following remark.

Remark 5.1.4. At hop $h-1$, given $\alpha_{h-1} = \hat{a} > 0$ then y (the number of *unblocked* positions generated) is bounded by:

$$\mathsf{min}\{\hat{a} + z, K\} \leq y \leq \mathsf{min}\{(z+1)\hat{a}, K\} \tag{5.1.4}$$

5.1.2 Example

An example of multi-hop time-blocking is shown in Fig. 5.4. In the example, $H = 4$ and each switching node use 2-*forwarding* scheme. Each hop contains one channel where $K = 12$ time-frames. In the example, until the third hop, there are two schedulable time-frames. However, all available time-frames of the fourth hop are blocked.

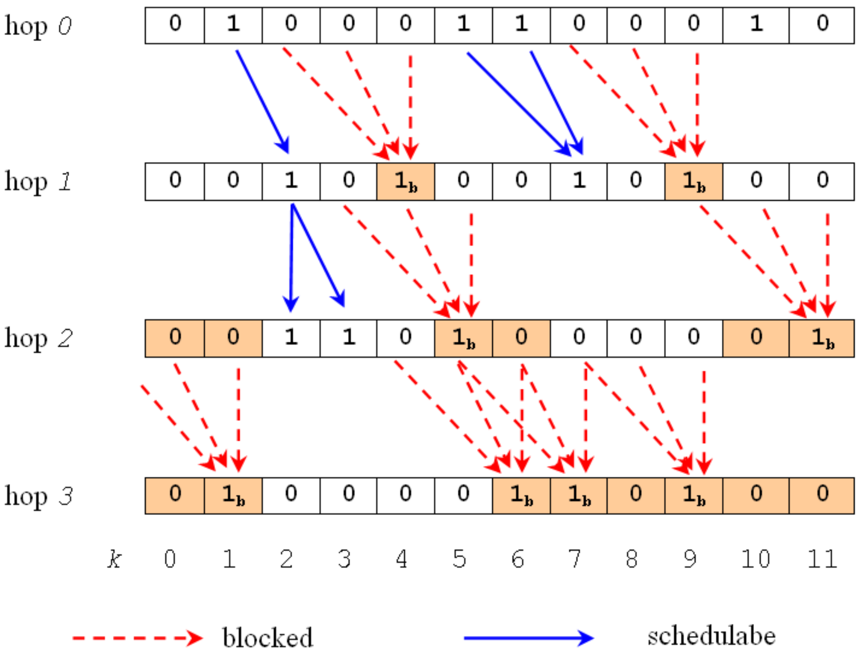

hop 0
0 1 0 0 0 1 1 0 0 0 1 0
hop 1
0 0 1 0 1_b 0 0 1 0 1_b 0 0
hop 2
0 0 1 1 0 1_b 0 0 0 0 0 1_b
hop 3
0 1_b 0 0 0 0 1_b 1_b 0 1_b 0 0
k 0 1 2 3 4 5 6 7 8 9 10 11
blocked
schedulabe

Figure 5.4: An example shows that there is blocking after 4 hops.

5.2 Stochastic ordering background

We present in this section some basic results of stochastic ordering. These results are important to prove the bounds of time-blocking probability that we shall derive in this chapter.

Definition 5.2.1. — Let Y and $\ddot{Y}$ be random variables taking values on the same finite ordered space $E = \{0, 1, 2, \cdots, K\}$ with $\mathbf{B}$ and $\ddot{\mathbf{B}}$ as associated probability distribution vectors:

$$\mathbf{B} \doteq \langle Pr(Y = j) \rangle$$

$$\ddot{\mathbf{B}} \doteq \langle Pr(\ddot{Y} = j) \rangle.$$

Then $Y \leqslant_{st} \ddot{Y}$ (i.e., Y is less than $\ddot{Y}$ in a stochastic sense) iff

$$\sum_{j=y}^{K} Pr(Y = j) \leq \sum_{j=y}^{K} Pr(\ddot{Y} = j) \tag{5.2.1}$$

for all $y = 0, 1, 2, \cdots, K$. Equivalently, we can write:

$$\mathbf{B} \leqslant_{st} \ddot{\mathbf{B}} \tag{5.2.2}$$

For example, we can have two following probability distribution vectors satisfying the above stochastic ordering: $\mathbf{B} = \big\langle 0, 0.22, 0.33, 0.45, 0 \big\rangle$ and $\ddot{\mathbf{B}} = \big\langle 0, 0.20, 0.34, 0.46, 0 \big\rangle$.

Definition 5.2.2. — Let $\mathcal{P}$ and $\ddot{\mathcal{P}}$ be two stochastic matrices. And let $\mathcal{P}_{y,*}$ and $\ddot{\mathcal{P}}_{y,*}$ denote row y of $\mathcal{P}$ and $\ddot{\mathcal{P}}$, respectively. If

$$\mathcal{P}_{y,*} \leqslant_{st} \ddot{\mathcal{P}}_{y,*}$$

for all y, then

$$\mathcal{P} \leqslant_{st} \ddot{\mathcal{P}} \tag{5.2.3}$$

Theorem 5.2.1. *If there are two probability distribution vectors of size K satisfying:*

$$\mathbf{B}_0 \leqslant_{st} \ddot{\mathbf{B}}_0$$

and the two matrices $\mathcal{P}$ and $\ddot{\mathcal{P}}$, whose sizes are $K \times K$, satisfying (5.2.3), then:

$$\mathbf{B}_h = \mathbf{B}_0 \; \mathcal{P}^h \leqslant_{st} \ddot{\mathbf{B}}_h = \ddot{\mathbf{B}}_0 \; \ddot{\mathcal{P}}^h \tag{5.2.4}$$

for any integer $h \geq 1$.

Fundamental concepts and details can be found in [13] and [46]. Stochastic ordering is a strong mathematic tool that is usually used to prove and derive bounds.

5.3 Exact Solution for Zero Scheduling Delay

5.3.1 Single channel per hop

When the scheduling delay is zero (i.e., $z = 0$), it is the IF scheme. For this special case, we have the following result.

Lemma 5.3.1. *For $z = 0$, the probability that there are $\tilde{a}$ schedulable time-frames at hop h is given by:*

$$Pr(\alpha_h = \tilde{a}) = \sum_{\hat{a}=0}^{a} \frac{\binom{\hat{a}}{\tilde{a}}\binom{K-\hat{a}}{a-\tilde{a}}}{\binom{K}{a}} \;\; Pr(\alpha_{h-1} = \hat{a}) \tag{5.3.1}$$

Proof. Let $Pr(\alpha_h = \tilde{a}|\alpha_{h-1} = \hat{a})$ represent the conditional probability that there are $\tilde{a}$ schedulable time-frames at hop h, given $\hat{a}$ schedulable time-frames at hop $h-1$.

An available time-frame of hop h becomes a schedulable time-frame if it is positioned "below" a schedulable time-frame of hop $h-1$. In order to generate exactly $\tilde{a}$ schedulable time-frames for hop h ($\tilde{a} < \hat{a}$), we distribute $\tilde{a}$ available time-frames "beneath" the total $\hat{a}$ possible positions, obtaining the number $\binom{\hat{a}}{\tilde{a}}$.

Remaining $a-\tilde{a}$ available time-frames must be placed under the $(K-\hat{a})$ busy time-frames of hop $h-1$ so that no more schedulable time-frame is generated, yielding the number $\binom{K-\hat{a}}{a-\tilde{a}}$.

Meanwhile, without any constraint, the total number of ways to distribute a available time-frames into K time-frame positions is $\binom{K}{a}$. Thus, we obtain:

$$Pr(\alpha_h = \tilde{a}|\alpha_{h-1} = \hat{a}) = \frac{\binom{\hat{a}}{\tilde{a}}\binom{K-\hat{a}}{a-\tilde{a}}}{\binom{K}{a}} \tag{5.3.2}$$

Taking

$$Pr(\alpha_h = \tilde{a}) \;\; = \sum_{\text{all possible } \hat{a}} Pr(\alpha_h = \tilde{a}|\alpha_{h-1} = \hat{a}) \;\; Pr(\alpha_{h-1} = \hat{a})$$

remembering (5.1.3) results in (5.3.1). □

With the initial condition of $Pr(\alpha_0 = \tilde{a})$ given in (5.1.2), eq. (5.3.1) is used repetitively to obtain the time-blocking probability for the IF scheme.

5.3.2 Multi-channel per hop

In FλS network, a link between two consecutive switches (a hop) usually has at least a fiber of C channels. Assume that we have a perfect load balancing between channels so that the load carried by each channel are equal. Assume further that no wavelength conversion is used. This is equivalent to the case that we have C independent paths (between source and destination) whose individual blocking probability is $p_0 = Pr(\alpha_{H-1} = 0)$. Thus, the C channels time-blocking probability, $p(C)$, is given by:

$$p(C) = \{p_0\}^C \tag{5.3.3}$$

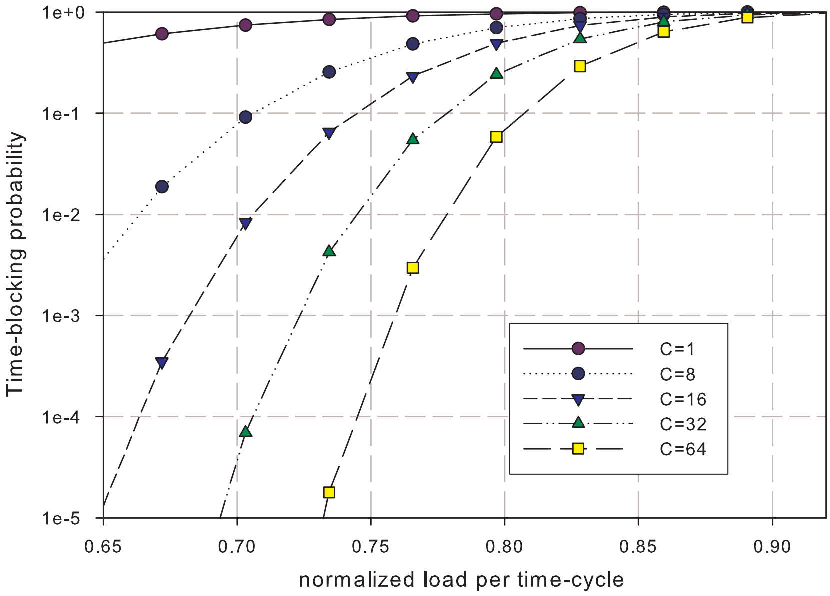


Figure 5.5: Numerical results for 5 hops, $K = 128$, $z = 0$, C varies.

Numerical results (Fig 5.5 - Fig. 5.8) show that even under zero scheduling delay scheme, having multiple channels per link helps reduce blocking probability significantly. Fig. 5.7 and Fig. 5.8 show how blocking probability is heavily affected by hop-length. For example, at 75% time-cycle loaded (Fig. 5.8), the blocking probability increases more than 100 times when hop is increased from 5 to 6, even if we have 64 channels.

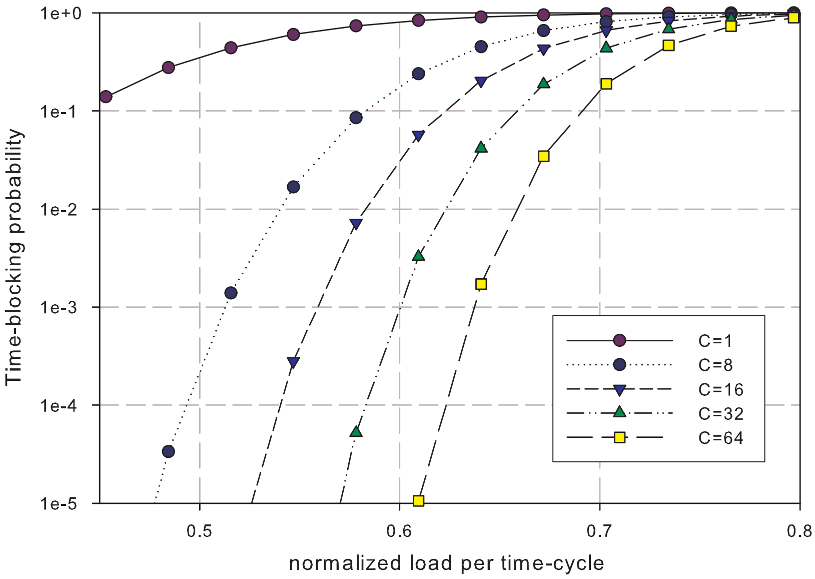


Figure 5.6: Numerical results for 7 hops, $K = 128$, $z = 0$, C varies.

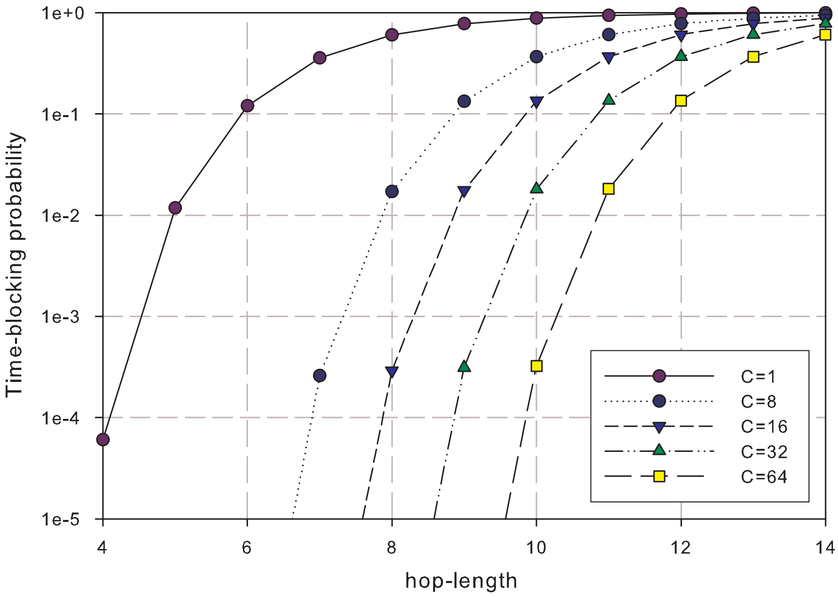


Figure 5.7: Numerical results when normalized load is 50%, hop-length varies.

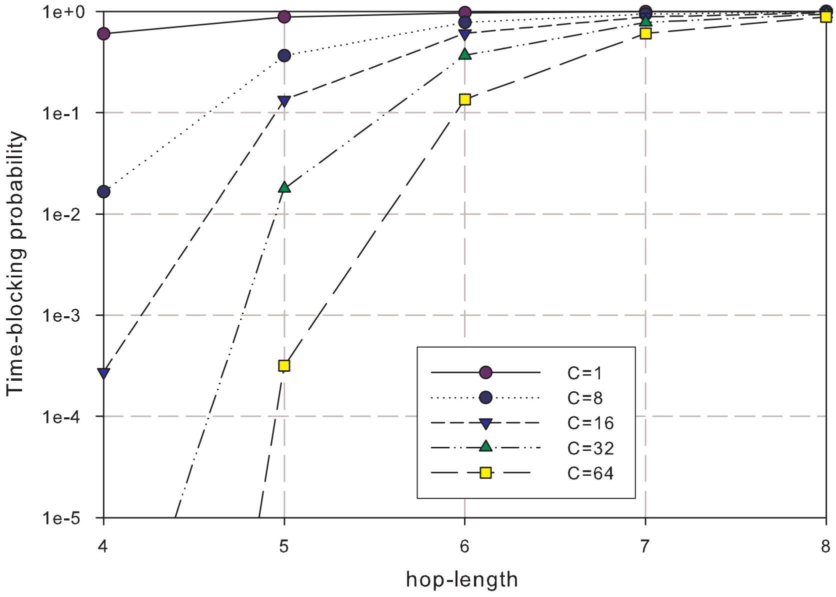


Figure 5.8: Numerical results when normalized load is 75%, hop-length varies.

5.4 Exact Solution for Nonzero Scheduling Delay

The analysis for zero scheduling delay is rather straightforward. The idea is that we can derive the quantity $Pr(\alpha_h = \tilde{a})$ using a hop-based computation. In this section, we apply this approach to analyze cases of nonzero scheduling delay schemes (i.e., $z \geq 1$).

Let us introduce two following conditional probabilities:

- $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$ denotes the probability that β_{h-1}=y, given that α_{h-1}=$\hat{a}$.
- $Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y)$ denotes the probability that α_h=$\tilde{a}$, given that β_{h-1}=y.

Basically, the following result holds for general cases.

Theorem 5.4.1. *Following coupled equations can be repetitively used to* *compute* $Pr(\alpha_h = \tilde{a})$ *for any hop* $h \geq 1$*:*

$$Pr(\beta_{h-1} = y) = \sum_{\hat{a}=0}^{a} Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a}) \;\; Pr(\alpha_{h-1} = \hat{a}) \qquad (5.4.1)$$

$$Pr(\alpha_h = \tilde{a}) = \sum_{y=0}^{y_{max}} Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y) \;\; Pr(\beta_{h-1} = y) \qquad (5.4.2)$$

where $y_{max} = \mathit{min}\{(z+1)a, K\}$.

Proof. The theorem is obvious according to probability theory and running parameters given in (5.1.3)-(5.1.4). □

According to Theorem 5.4.1, we can obtain the time-blocking probability if we are able to compute two conditional probabilities $Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y)$ and $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$. More specifically, the time-blocking probability after H hops is computed by eq. (5.4.2) for $\alpha_{H-1} = 0$.

Lemma 5.4.2. $Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y)$ *is computed by:*

$$Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y) = \begin{cases} \frac{\binom{y}{\tilde{a}}\binom{K-y}{a-\tilde{a}}}{\binom{K}{a}} & \text{,}\mathit{if}\;\; \tilde{a} \leq y \;\&\; a - \tilde{a} \leq K - y \\ 0 & \text{,}\mathit{otherwise} \end{cases} \qquad (5.4.3)$$

Proof. In order to have $\tilde{a}$ schedulable time-frames at hop h, we distribute $\tilde{a}$ available time-frames among y *unblocked* positions generated by hop $h-1$, which yields $\binom{y}{\tilde{a}}$. To block the other $(a - \tilde{a})$ available time-frames, they must be arranged among $(K - y)$ *blocked* positions, which yields $\binom{K-y}{a-\tilde{a}}$. Meanwhile, without any constraint, the total number of dispositions is $\binom{K}{a}$. Thus, we derive $Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y)$ as in (5.4.3). □

Though $Pr(\alpha_h = \tilde{a}|\beta_{h-1} = y)$ is easily computed as in Lemma 5.4.2, it is more challenging to find $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$.

In fact, we observe the inclination that schedulable time-frames tend to form "batches" rather than being uniformly distributed among K positions. The exact computation of $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$ depends not only on a certain *z-forwarding* scheme but also on the detail distribution of $\hat{a}$ schedulable time-frames, which is no longer uniform due to the allowance of nonzero scheduling delay.

Therefore, an exact solution for time-blocking probability is possible if we are able to compute probabilities associated with all possible distribution of $\hat{a}$ schedulable time-frames. We demonstrate the process to obtain the exact time-blocking probability by examining one simple example.

5.4.1 An example for small K

We perform the exact solution for a set of small parameters: $K = 6, a = 2, z = 1$. Following (5.1.3) and (5.1.4) we have $0 \leq \hat{a} \leq 2$ and $y \in \{0, 2, 3, 4\}$.

Table 5.1: All possible patterns and y values for $K = 6, a = 2, z = 1$.

State	$\hat{a}$	Pattern	y
q_0	0	000000	0
q_2	1	100000	2
q_3	2	110000	3
q_{41}	2	101000	4
q_{42}	2	100100	4

For each possible $\hat{a}$, we consider all possible patterns taking into account the ordering of run-lengths of `0`'s and of `1`'s. For example, by allowing shifting the two following patterns are interchangeable: $101000 \equiv 100010$. All patterns and their corresponding $\hat{a}$ and y are given in Table 5.1. The number of patterns is small. Transition probabilities between patterns can be easily computed as shown in Table 5.2.

Table 5.2: Transition probability between patterns.

	q_0	q_2	q_3	q_{41}	q_{42}
q_0	1	0	0	0	0
q_2	6/15	8/15	1/15	0	0
q_3	3/15	9/15	2/15	1/15	0
q_{41}	1/15	8/15	3/15	2/15	1/15
q_{42}	1/15	8/15	2/15	2/15	2/15

Let $\mathbf{q}_h$ be the vector associated with the probability that hop h is at the state q_*: $\mathbf{q}_h \doteq \langle Pr(q_*) \rangle$. Let $\mathbf{Q}$ be the transition matrix given in Table 5.2.

We have $\mathbf{q}_0 \doteq \langle 0, 0, \frac{3}{15}, \frac{6}{15}, \frac{6}{15} \rangle$.

And the vector $\mathbf{q}_h$ is computed by:

$$\mathbf{q}_h = \mathbf{q}_0 \times \mathbf{Q}^h$$

The above equation is used to compute $\mathbf{q}_{H-2}$ then (5.4.2) is applied to compute the blocking probability.

5.4.2 Magnitude of complexity

In principle, the exact solution requires two steps:

- Step 1: acquire knowledge of all combination patterns.
- Step 2: compute a transition probability for every pair of patterns.

Let $\Pi(b, n)$ be the number of partitions of the integer b into n parts. Let N_S be the number of patterns at a load point (K, b). The number of patterns is given by:

$$N_S = \sum_{n=1}^{K-b} \Pi(b, n)$$

In fact, $\Pi(b, n)$ is the number of ways to distribute b identical balls into n identical bins (i.e., bins are not ordered) with no constraint. Note that $1 \leq n \leq a$.

For example, $\sum_{n=1}^{64} \Pi(64, n) = 1,741,360$. Thus, an exact blocking probability at the point of 50% load of a time-cycle $K = 128$ requires a knowledge of $1,741,360$ patterns and the computation of transition probability between any pair of patterns, which requires a matrix of $1,741,360^2$ cells. This computation is impractical.

5.5 A Lower Bound for Nonzero Scheduling Delay

As we discuss in the last section, the exact quantity of $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$ can not be exactly computed if we consider the uniform distribution of $\hat{a}$ schedulable time-frames. However, we can obtain the lower bound of blocking probability based on this uniform distribution assumption.

Now, at hop $h - 1$, given α_{h-1}=$\hat{a}$ let:

- $C(y)$ be the number of combinations that generate exactly y *unblocked* positions;
- C_T be the total number of combinations.

Note that each combination is the disposition of three types of time-frames($\tilde{a}$ schedulable, $a - \tilde{a}$ blocked, and b busy) into K cyclic positions of a time-cycle. We have the following lemma.

Lemma 5.5.1. *$Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$ is given by:*

$$Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a}) = \begin{cases} \frac{C(y)}{C_T} & \textit{,if (5.1.3) \& (5.1.4) \& } \hat{a} > 0 \\ 1 & \textit{,if } y = \hat{a} = 0 \\ 0 & \textit{,otherwise} \end{cases} \tag{5.5.1}$$

Proof. It is obvious that if $\hat{a}$=0, then y=0, thus $Pr(\beta_{h-1} = 0|\alpha_{h-1} = 0) = 1$.

Following Remarks 5.1.3 and 5.1.4, it is clear that if $\hat{a}$ and y do not respectively satisfy (5.1.3) and (5.1.4), then $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a}) = 0$. Otherwise, assume $C(y)$ and C_T are countable (we shall discuss the derivation of $C(y_{h-1})$ and C_T later), we have the combinatorial probability as the first entry of (5.5.1). □

5.5.1 Combinatorial numbers $C(y)$ and C_T

Essentially, we must consider all possible dispositions of three types of time-frames (schedulable, blocked, and busy) into K cyclic positions of a time-cycle under some certain constraints. Before pointing out some challenges that prevent us to derive correct versions of $C(y)$ and C_T, we introduce approximated versions of these two numbers.

One can see that a blocked time-frame '$\mathtt{1}_b$' plays no role in the process of generating *unblocked* positions at hop $h-1$. Based on this observation, we can derive an approximation by treating blocked time-frames '$\mathtt{1}_b$' and busy time-frames '`0`' equally. In other words, we approximate the number $C(y)$ by considering how to dispose:

- $\hat{a}$ schedulable time-frames
- and $\hat{b}$=K−$\hat{a}$ "busy" time-frames

into K time-frame positions so that each combination generate exactly y *unblocked* positions.

Approximating the number C_T

Accordingly, the number C_T can be approximated by:

$$C_T \approx \binom{K}{\hat{a}} \tag{5.5.2}$$

Approximating the number $C(y)$

In order to approximate the number $C(y)$, recall $C(x)$ - the number of dispositions that each generate exactly x blocked positions (defined in Section 4.5 - Chapter 4).

From Remark 5.1.1, it is trivial to see that if $x + y = K$, the two combinatorial are equal:

$$C(y) = C(x)$$

Therefore, instead of approximating $C(y)$, it is equivalent to approximate $C(x)$, by which we can reuse fundamental results presented in Section 4.5 - Chapter 4.

Since $x = K - y$, following (5.1.4) we modify the bound of x:

$$\mathsf{max}\{0, \hat{b} - z\hat{a}\} \leq x \leq \mathsf{max}\{0, \hat{b} - z\} \tag{5.5.3}$$

Applying the same combinatorial analysis that we used in Section 4.5 - Chapter 4, we obtain:

$$0 \leq \hat{b}_u = x + zu \leq \hat{b} \tag{5.5.4}$$

Note that $\hat{b}_u = 0$ when $u = 0$, $x = 0$ or $y = K$.

The number of symbols $\hat{b}_v$ is given by:

$$0 \leq \hat{b}_v = \hat{b} - \hat{b}_u = \hat{b} - x + zu \tag{5.5.5}$$

We have:

$$1 \leq u + v \leq \hat{a} \tag{5.5.6}$$

Also, u is bounded by:

$$0 \leq u \leq \mathsf{min}\{\hat{a}, \lfloor\frac{\hat{b} - x}{z}\rfloor\} \tag{5.5.7}$$

For $1 < z \leq K - 1$, v is bounded by:

$$\lceil\frac{\hat{b}_v}{z - 1}\rceil \leq v \leq \mathsf{min}\{\hat{a} - u, \hat{b}_v\} \tag{5.5.8}$$

For a pair of valid (u, v) satisfies (5.5.6), (5.5.7) and (5.5.8), we have:

$$C(u, v) = \frac{KC_{uv}C_{\hat{a}}C_{\hat{b}_u}C_{\hat{b}_v}}{u + v} \tag{5.5.9}$$

where:

$$C_{uv} = \binom{u + v}{u} \tag{5.5.10}$$

and

$$C_{\hat{a}} = \binom{\hat{a} - 1}{u + v - 1} \tag{5.5.11}$$

and

$$C_{\hat{b}_u} = \binom{\hat{b}_u - (z - 1)u - 1}{u - 1} \tag{5.5.12}$$

and

$$C_{\hat{b}_v} = \begin{cases} \sum_{i=0}^{v} (-1)^i \binom{v}{i} \binom{\hat{b}_v - i(z-1) - 1}{v-1} & ,\text{if } v > 0 \\ 1 & ,\hat{b}_v = v \end{cases} \tag{5.5.13}$$

Thus, $C(x)$ is given by:

$$C(x) = \sum_{u=0}^{\mathsf{min}\{\hat{a}, \lfloor \frac{\hat{b} - x}{z} \rfloor\}} \left\{ \sum_{\mathsf{min}\{\hat{a} - u, \hat{b}_v\}}^{\lceil \frac{\hat{b}_v}{z-1} \rceil} C_{(u,v)}\Big|_{u+v \le \hat{a}} \right\} \tag{5.5.14}$$

Finally, replacing $x = K - y$, $K - \hat{b} = \hat{a}$, we obtain:

$$C(y) \approx \sum_{u=0}^{\mathsf{min}\{\hat{a}, \lfloor \frac{y - \hat{a}}{z} \rfloor\}} \left\{ \sum_{\mathsf{min}\{\hat{a} - u, \hat{b}_v\}}^{\lceil \frac{\hat{b}_v}{z-1} \rceil} C_{(u,v)}\Big|_{u+v \le \hat{a}} \right\} \tag{5.5.15}$$

Note that for $z = 1$, we have a simplified version:

$$C(u) = \frac{K}{u} \binom{\hat{a} - 1}{u - 1} \binom{\hat{b} - 1}{u - 1} \tag{5.5.16}$$

and thus:

$$C(y) \approx \sum_{u=0}^{\mathsf{min}\{\hat{a}, y - \hat{a}\}} \frac{K}{u} \binom{\hat{a} - 1}{u - 1} \binom{\hat{b} - 1}{u - 1} \qquad \text{for } z = 1 \tag{5.5.17}$$

5.5.2 Observation and remarks

An approximate time-blocking probability can be obtained according to Theorem 5.4.1, Lemma 5.4.2, Lemma 5.5.1 once the numbers $C(y)$ and C_T are computed.

A further observation suggests that the more accurate approximation can be achieved. The fact is that by treating '$\mathtt{1}_b$' and '$\mathtt{0}$' equally, we implicitly count some forbidden combination patterns while deriving $C(y)$ and C_T.

Specifically, let $\overline{1_b\{i\}1_b}$ represent a pattern when a series of i consecutive schedulable time-frames (i.e, symbols '$\mathtt{1}$') are "positioned" in between two consecutive blocked time-frames (i.e., symbols '$\mathtt{1}_b$'). For nonzero scheduling delays (i.e., $z \geq 1$), a pattern $\overline{1_b\{i\}1_b}$ such that $i \leq z$ is not allowed to appear. We call these patterns by forbidden patterns.

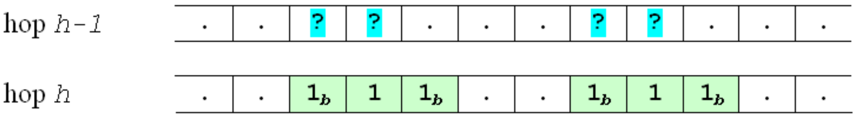

hop h-1
. . ? ? . . . ? ? . . .
hop h
. . $\mathtt{1}_b$ $\mathtt{1}$ $\mathtt{1}_b$ . . $\mathtt{1}_b$ $\mathtt{1}$ $\mathtt{1}_b$ . .

Figure 5.9: For $z = 1$: there is one forbidden pattern.

The case of forbidden patterns is illustrated in Fig. 5.9 and Fig. 5.10. In Fig. 5.9 assume that at hop h, a disposition of all time-frames contain a forbidden pattern (time-frame positions in grey), then no time-frame among three types of time-frame (schedulable '$\mathtt{1}$', blocked '$\mathtt{1}_b$', and busy '$\mathtt{0}$') can be put in a "question mark" position at hop $h-1$ so that the 1-forwarding scheme satisfies.

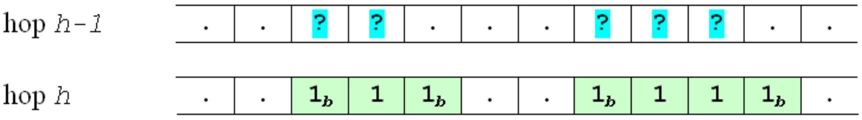

hop h-1
. . ? ? . . . ? ? ? . .
hop h
. . $\mathtt{1}_b$ $\mathtt{1}$ $\mathtt{1}_b$ . . $\mathtt{1}_b$ $\mathtt{1}$ $\mathtt{1}$ $\mathtt{1}_b$.

Figure 5.10: For $z = 2$, there are two forbidden patterns.

The number of possible forbidden patterns increases as a possible scheduling delay z is increased. As in Fig. 5.10 where $z = 2$, there are two possible forbidden patterns compared to only one forbidden pattern when $z = 1$ (in Fig. 5.9). This fact implies that, using approximate results given in (5.5.2) and (5.5.15), we obtain less accurate results if more scheduling delay is allowed (i.e., large z). This remark will be examined later in Section 5.7.

Obviously, a more precise approximation can be obtained if we distinguish '$\mathtt{1}_b$' and '$\mathtt{0}$' in the counting process. In other words, we consider the distribution of three types of time-frames ('$\mathtt{1}$', '$\mathtt{1}_b$', and '$\mathtt{0}$') and eliminate all forbidden patterns while counting $C(y)$ and C_T. Remember that the placements of time-frames into a time-cycle has the cyclic property. The more accurate approximation can be accomplished using Polya counting and the Inclusion-Exclusion principle [68]. However, we believe that it has a high complexity and it deserves a further consideration in the future.

Another remark is that the accuracy of approximation based on (5.5.2) and (5.5.15) degrades as the hop-length increases. Since errors appear in the conditional probability $Pr(\beta_{h-1} = y|\alpha_{h-1} = \hat{a})$, which later transfers the errors to computations of probability for the next hop. Thus, the errors are aggrandized as hop-length increases. This remark is also examined in Section 5.7.

5.5.3 Why lower bound?

The further scrutiny shows that even if all forbidden combination patterns are eliminated while deriving $C(y)$ and C_T, we still obtain a lower bound of time-blocking.

Let $\mathbf{B}_h$ be the vector associated with $Pr(\beta_{h-1} = y)$ for all possible y: $\mathbf{B}_h \doteq \langle Pr(\beta_{h-1} = y)\rangle$, where $Pr(\beta_{h-1} = y)$ is computed using (5.5.1).

Let $\ddot{\mathbf{B}}_h$ be the vector associated with real distribution of $Pr(\beta_{h-1} = y)$.

We conjecture that if all schedulable time-frames are uniformly distributed among K positions, it results in:

$$\ddot{\mathbf{B}}_h \leq_{st} \mathbf{B}_h$$

This means the quantity $Pr(\beta_{h-1} = y)$ computed by (5.5.1) is overvalued for high values of y and undervalued for low values of y. Meanwhile, the more *unblocked* positions there are, the potentially lower blocking it is. Thus the above stochastic ordering implies the approximate probability is the lower bound of time-blocking probability. A formal proof of the above hypothesis needs further study.

5.6 An Upper Bound for Nonzero Scheduling Delay

The analytical approach we use in the previous section is illustrated through the process composed by two dotted arrows in Fig. 5.11.

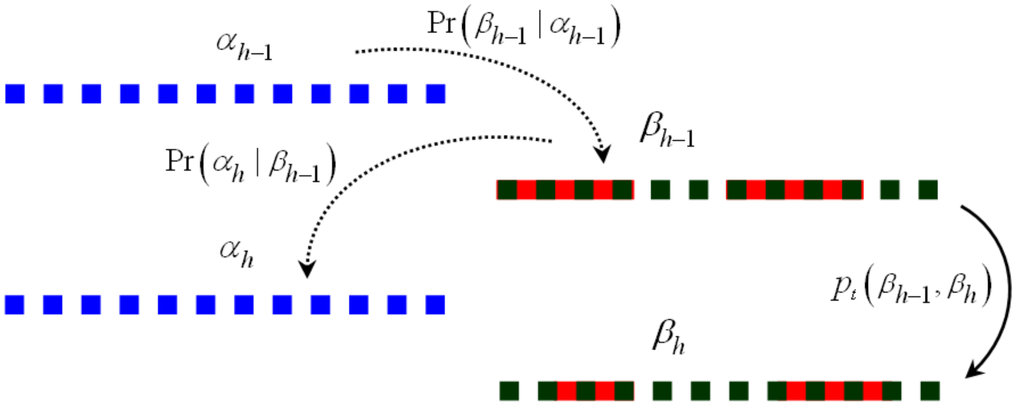

α_{h-1}
$\Pr(\beta_{h-1} \mid \alpha_{h-1})$
β_{h-1}
$\Pr(\alpha_h \mid \beta_{h-1})$
α_h
$p_t(\beta_{h-1}, \beta_h)$
β_h

Figure 5.11: Hop-based computing process.

It is obvious that in order to compute time-blocking probability, we must compute the probability distribution of $\mathbf{B}_h$.

Let $p_t(y, y')$ be the probability $p_t(\beta_{h-1}= y,\beta_h= y')$. For short, we use the notation $p_t(y, y')$ in replacement of $p_t(\beta_{h-1} = y, \beta_h = y')$. $p_t(y, y')$ is the transition probability that there are y' *unblocked* positions generated at hop h, transited from y *unblocked* positions generated at hop $h-1$. $p_t(y, y')$ is illustrated as the solid arrow in Fig. 5.11.

5.6.1 Compact combination patterns

The exact computation of $p_t(y, y')$ requires knowledge of concrete combination patterns at both two hops $h-1$ and h that generate exactly y and y' *unblocked* positions, respectively. For large (K, a), it is impossible (see discussion in Section 5.4.2. However, an approximation of $p_t(y, y')$ can be computed by considering transitions from only "compact" patterns of hop $h-1$.

Given $\hat{a}$ schedulable time-frames and $(K-\hat{a})$ busy time-frames (again we simplify the problem by noting that blocked and busy time-frames are treated equally), let $\ddot{\mathbb{C}}$ be the set of all combination patterns that each generates exactly y *unblocked* positions.

In set $\ddot{\mathbb{C}}$, there is a subset of combination patterns in which an individual pattern is considered "compact". A "compact" pattern is defined as a combination that all y *unblocked* positions are grouped together. In other words, a "compact" pattern at hop $h-1$ has a sequence of continuous y positions that are *unblocked*, and $K-y$ continuous positions that are *blocked*. Examples of compact pattern are illustrated in Fig. 5.12. We

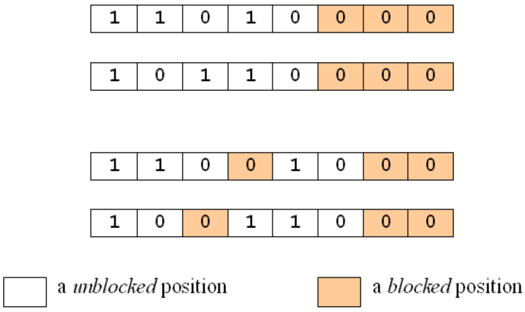


Figure 5.12: $K = 8$, $\hat{a} = 3$ and $y_{h-1} = 5$: the two uppermost patterns are compacted, while the two lowermost are not.

shall focus on deriving the quantity $p_t(y, y')$ considering transitions only from a compact pattern of y.

5.6.2 Transition matrix of $p_t(y, y')$ for $z = 1$

Let $\mathcal{P}$ denote the transition matrix whose entries are probabilities $p_t(y, y')$ for all possible pairs of (y, y'), considering transitions from compact patterns. We now focus on deriving $\mathcal{P}$. Matrix $\mathcal{P}$ will be used for computing the upper bound of time-blocking probability.

In this subsection, we focus on deriving the $p_t(y, y')$ matrix for the case $z = 1$. We shall discuss the extension for cases $z > 1$ later in Section 5.6.5.

Since $y_{max} = y'_{max} = K$, the size of matrix $\mathcal{P}$ is $K \times K$ (see Fig. 5.13).

$$\mathcal{P} = \begin{pmatrix} p_t(0,0) & \cdots & p_t(0,K) \\ p_t(1,0) & \cdots & p_t(1,K) \\ p_t(2,0) & \cdots & p_t(2,K) \\ \cdots & p_t(y,y') & \cdots \\ p_t(K,0) & \cdots & p_t(K,K) \end{pmatrix}$$

Figure 5.13: The transition matrix $\mathcal{P}$ for $z = 1$.

Theorem 5.6.1. *For $z = 1$, the entry $p_t(y, y')$ of the matrix $\mathcal{P}$ is given by:*

$$p_t(y,y') = \begin{cases} 1 & , \textit{if} \quad y = y' = 0 \\ \binom{K-y}{a} / \binom{K}{a} & , \textit{if} \;\; 1 < y \;\&\; y' = 0 \\ C(y,y') / \binom{K}{a} & , \textit{if} \;\; 1 < y \;\&\; 1 < y' \leq y + 1 \\ 0 & , \textit{otherwise} \end{cases} \tag{5.6.1}$$

where

$$C(y,y') = \sum_{n=1}^{\lfloor y'/2 \rfloor} \binom{y'-n-1}{n-1} \binom{y-y'+n+1}{n} \binom{K-y}{\hat{a}-y'+n} \tag{5.6.2}$$

Proof. If there is no *unblocked* position of hop $h-1$, then there is no *unblocked* position of hop h, thus $p_t(0,0)$=1. Also note that $p_t(y,y')$=0 for all entries such that y=1 or y'=1, since these values do not exists.

For $y>1$, in order to obtain $y'=0$, all a available time-frames of hop h must be distributed in *blocked* positions. The total number of *blocked* positions generated by hop $h-1$ is $K-y$. There are $\binom{K-y}{a}$ ways to do so (shown in Fig. 5.14). Meanwhile, without any constraint, the total number of ways to distribute a available time-frames of hop h is $\binom{K}{a}$. Thus, we yield the second entry of eq. (5.6.1).

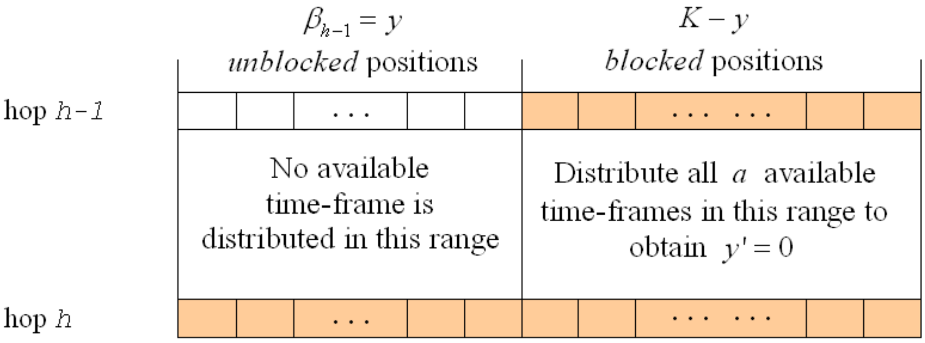

$\beta_{h-1}=y$
unblocked positions
$K-y$
blocked positions
hop h-1
. . .
. . . . . .
No available time-frame is distributed in this range
Distribute all a available time-frames in this range to obtain $y'=0$
hop h
. . .
. . . . . .

Figure 5.14: Computing $p_t(y,y')$ for entries that $y>1$, $y'=0$.

For $1<y'\leq y+1$, we need some basic combinatorial examinations to derive $C(y,y')$. Let n be the number of runs of 1's that constitutes y'. We have the bound of n:

$$1\leq n\leq \lfloor\frac{y'}{2}\rfloor \tag{5.6.3}$$

Let m be the number of symbols '1' that constitutes y' *unblocked* positions of hop h. Since $z=1$, we have the following simple relation:

$$y'=n+m \tag{5.6.4}$$

For a given pair of (m,n) satisfying (5.6.3)-(5.6.4), we need to find the number of ways to distribute the m symbols '1' into the n runs such that no empty is allowed, which yields $\binom{m-1}{n-1}$.

Besides, note that n runs of `1`'s can "float" inside a "compact" range of y positions of hop $h-1$. There must be at least one '`0`' separating two consecutive runs of `1`'s. This is equivalent to the case that there are $n+1$ runs of `0`'s (illustrated in Fig. 5.15). With two extra '`0`' at two ends of y range, the total number of symbols '`0`' that can be distributed into these $n+1$ runs of `0`'s is $y-m+2$. The total number of ways to distribute $y-m+2$ symbols '`0`' into $n+1$ runs where no empty is allowed is $\binom{y-m+2-1}{n+1-1} = \binom{y-m+1}{n}$.

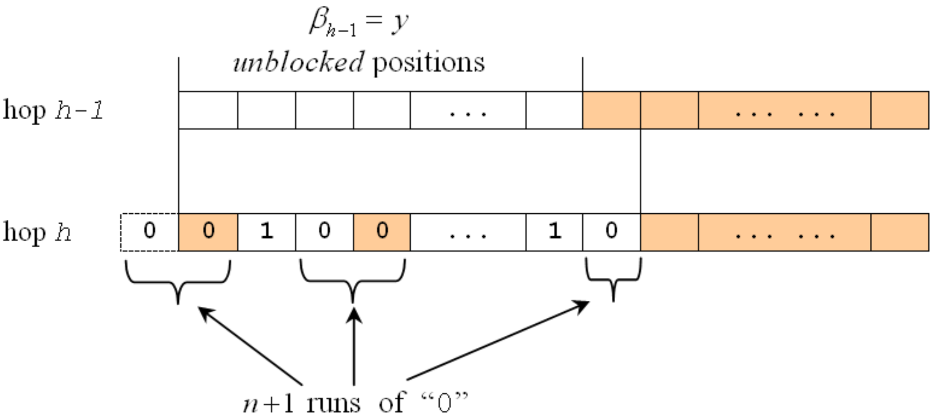


Figure 5.15: There are n runs of `1`'s, then there are $n+1$ runs `0`'s.

Furthermore, the other $(a-m)$ available time-frames are blocked by being distributed into $(K-y)$ *blocked* positions, which yields $\binom{K-y}{a-m}$.

Thus, for a valid pair of (m,n), the number of ways $C(n,m)$ to generate y' *unblocked* positions for hop h is given:

$$C(n,m) = \binom{m-1}{n-1}\binom{y-m+1}{n}\binom{K-y}{a-m} \tag{5.6.5}$$

The sum of (5.6.5) over all valid pair of (m,n) following (5.6.3)-(5.6.4) yields $C(y,y')$ as in (5.6.2), which fulfills the third entry of (5.6.1).

Finally, note that for $y > 1$, if $y' > y+1$, then $p_t(y,y')$=0 since by construction, for $z=1$, transiting from a compact pattern of y *unblocked* positions, the *unblocked* range of hop h can expand at most 1 more position, or $y'_{max} = y + 1$. □

5.6.3 Upper bound computation for $z = 1$

Let

$$\mathbf{B}_0 \doteq \langle Pr(\beta_0) \rangle$$

be the initial vector representing all possibilities of compact combination patterns of the first hop ($h = 0$). Recall the notation

$$\mathbf{B}_h \doteq \langle Pr(\beta_h) \rangle$$

representing a distribution vector of $Pr(\beta_h)$. All vectors have the same size of $K + 1$. We have the following result.

Theorem 5.6.2. *For $h > 1$, $\mathbf{B}_h$ is computed by:*

$$\mathbf{B}_h = \mathbf{B}_0 \times \mathcal{P}^h \tag{5.6.6}$$

where entries of the matrix $\mathcal{P}$ is given in (5.6.1) and the entry $p(\beta_0 = y)$ of the initial vector $\mathbf{B}_0$ is given by:

$$Pr(\beta_0 = y) = \begin{cases} \frac{K}{y-a}\binom{a-1}{y-a-1}\binom{K-a-1}{y-a-1}/\binom{K}{a} & , \textit{if } 2 \le y \le \mathsf{min}\{2a, K\} \\ 0.0 & , \textit{otherwise} \end{cases} \tag{5.6.7}$$

Proof. First, let us derive the initial vector $\mathbf{B}_0$. Note that $Pr(\beta_0 = y)$ is the probability that the first hop generates y *unblocked* positions, or equivalently $x = K - y$ *blocked* positions. For $2 \le y \le \mathsf{min}\{2a, K\}$, eq. (5.6.7) is a direct outcome of eq. (4.4.2) with the note that

$$u = b - x = (K - a) - (K - y) = y - a.$$

Besides, $Pr(\beta_0 = y) = 0$ for $y = 0$ or $y = 1$ or $y > \mathsf{min}\{2a, K\}$, since these values do not exit at the first hop.

Eq. (5.6.6) follows directly as the consequence that all possible transitions from hop $h-1$ to hop h are taken into account in the transition matrix $\mathcal{P}$. □

Corollary 5.6.3. *Obtaining vector* $\mathbf{B}_{H-2} = \big\langle Pr(\beta_{H-2} = y)\big\rangle$*, eq. (5.4.2) is used to compute the upper bound of time-blocking probability* p_u*:*

$$p_u = Pr(\alpha_{H-1} = 0) = \sum_{y=0}^{K} \frac{\binom{K-y}{a}}{\binom{K}{a}} \; Pr(\beta_{H-2} = y) \qquad (5.6.8)$$

5.6.4 Proof of the upper bound for $z = 1$

We now provide the formal proof of the upper bound for the case $z = 1$.

Let $\mathbb{D}$ be the set of possible distributions of y *unblocked* positions at hop $h-1$. Let $d \doteq \|\mathbb{D}\|$. In set $\mathbb{D}$:

- denote κ_0 the compact distribution.
- denote κ_i a random distribution i, $i = 1, 2, \cdots, d-1$.

Let $q(y, \kappa_0)$ be the probability that there are y *unblocked* positions of hop $h-1$, considering only a compact pattern. Thus:

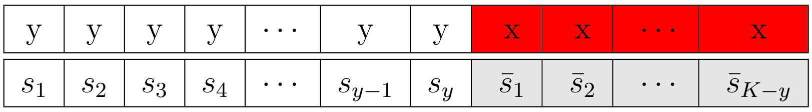
$$q(y, \kappa_0) = 1 \qquad (5.6.9)$$

Let $\ddot{q}(y, \kappa_i)$ be the probability that there are y *unblocked* positions (of hop $h-1$) associated with pattern κ_i, considering all distribution patterns. Thus:

$$\sum_{i=0}^{d-1} \ddot{q}(y, \kappa_i) = 1 \qquad (5.6.10)$$

Let y' be the number of *unblocked* positions generated by a combination

$$Ç' \doteq s_1 s_2 \cdots s_{y-1} s_y \bar{s}_1 \bar{s}_2 \cdots \bar{s}_{K-y-1} \bar{s}_{K-y}$$

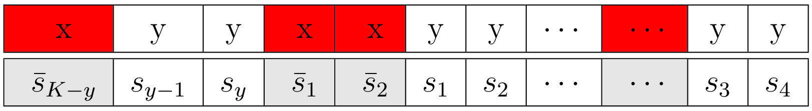


y	y	y	y	$\cdots$	y	y	x	x	$\cdots$	x
s_1	s_2	s_3	s_4	$\cdots$	s_{y-1}	s_y	$\bar{s}_1$	$\bar{s}_2$	$\cdots$	$\bar{s}_{K-y}$

Figure 5.16: A transition from a compact pattern.

of time-frames at hop h , transitioned from y *unblocked* positions of the previous hop considering only a compact pattern. An illustration is shown in Fig. 5.6.4.

Let y'' be the number of *unblocked* positions generated by a combination Ç$''$ at hop h, transitioned from y *unblocked* positions of the previous hop considering a random distributed pattern. An illustration is shown in Fig. 5.6.4.



x	y	y	x	x	y	y	$\cdots$	$\cdots$	y	y
$\bar{s}_{K-y}$	s_{y-1}	s_y	$\bar{s}_1$	$\bar{s}_2$	s_1	s_2	$\cdots$	$\cdots$	s_3	s_4

Figure 5.17: A transition from a random distributed pattern.

There exists the mapping

$$\text{Ç}' \longmapsto \text{Ç}'',$$

so that the order of symbols in the two sequences $s_1 s_2 \cdots s_y$ and $\bar{s}_1 \bar{s}_2 \cdots \bar{s}_{K-y}$ are unchanged, though the symbols in one sequence are distributed and scrambled by symbols of the other sequence. An illustration is depicted through Fig. 5.6.4 and Fig. 5.6.4.

If Ç$'$ is unique then Ç$''$ is unique since the number of symbols of each type (‘`0`’, ‘$\mathtt{1}_b$’ and ‘`1`’) are fixed. Thus, the mapping is 1-to-1: Ç$' \leftrightarrow$ Ç$''$.

Remark 5.6.1. For an individual sequence Ç$' \longmapsto$ Ç$''$ we have $y' \leq y'$ if $z = 1$. If all possible sequences of symbol Ç$'$ are considered, then y' and y' are two random variables taking samples from a finite ordered space $E = \{0, 1, 2, \cdots, K\}$. Therefore:

$$y' \leqslant_{st} y''$$

Remark 5.6.1 associated with (5.6.9) and (5.6.10) implies

$$\mathcal{P}_{y,*} \leqslant_{st} \ddot{\mathcal{P}}_{y,*} \tag{5.6.11}$$

for all rows of $\mathcal{P}$ and $\ddot{\mathcal{P}}$. Thus, by Def. 5.2.2 we have:

$$\mathcal{P} \leqslant_{st} \ddot{\mathcal{P}} \tag{5.6.12}$$

By Theorem 5.2.1, $\mathbf{B}_{H-2}$ mentioned in Corollary 5.6.3 has lower stochastic order than $\ddot{\mathbf{B}}_{H-2}$, which is the real distribution of $Pr(\beta_{H-2} = y$:

$$\mathbf{B}_{H-2} \leqslant_{st} \ddot{\mathbf{B}}_{H-2}.$$

Thus, we obtain the upper bound of time-blocking probability if the computation is based on vector $\mathbf{B}_{H-2}$.

5.6.5 Extension for $z > 1$

Following are two steps to extend the analysis for general $z > 1$:

- Recompute the initial vector $\mathbf{B_0}$. (The results presented in Section 4.5 - Chapter 4 will be reused.)
- Recompute all entries $p_t(y, y')$ of matrix $\mathcal{P}$. (In fact, when $z > 1$, deriving $p_t(y, y')$ is a bit tedious.)

Deriving $p_t(y, y')$ for $z > 1$

Lemma 5.6.4. *The number of runs of '1' constituting y', n is bounded by:*

$$1 \leq n \leq \lfloor \frac{y' - (z-1)}{2} \rfloor \tag{5.6.13}$$

Proof. The counting for $z > 1$ is illustrated in Fig. 5.18. Since the last run of 0's always has run-length $l_i \geq z$, it contributes at least z *unblocked* positions. However, we deduct $z - 1$ *blocked* positions and then consider

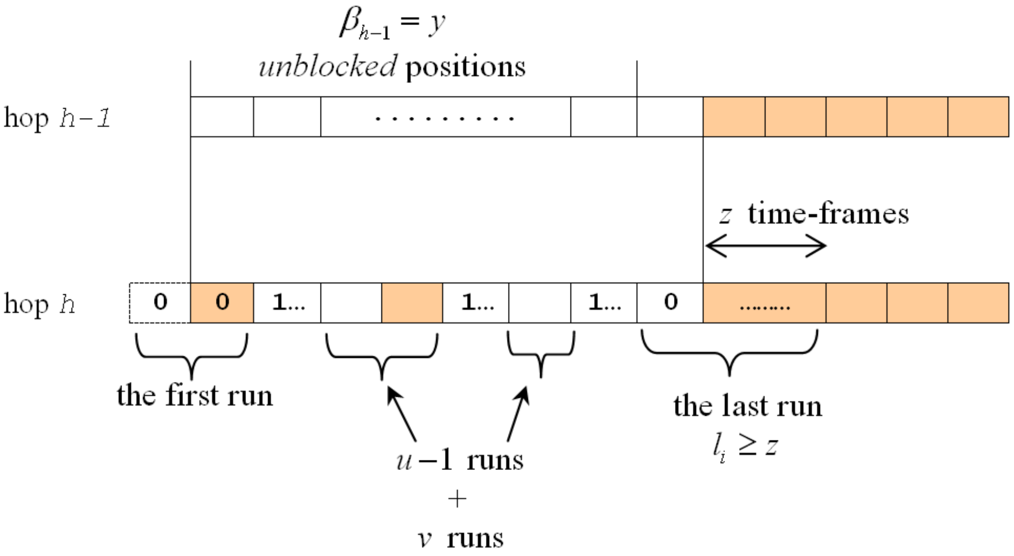


Figure 5.18: Computing $p_t(y, y')$ for general case $z > 1$.

that each run of 1's contributes at least 2 *blocked* positions. Therefore, at most we can have $n_{max} = \frac{y'-(z-1)}{2}$ runs of 1's. Besides, it is obvious that $n_{min} = 1$. □

Lemma 5.6.5. *Number of symbols '1' constituting y'* blocked *positions, m is bounded by:*

$$\mathsf{max}\{y' - nz, n\} \leq m \leq \mathsf{min}\{a, y' - n - z + 1\} \tag{5.6.14}$$

Proof. Let o be the number of '0' constituting y'. Since the last run of 0's always contributes z *unblocked* positions, and each of the other runs of 0's contributes at least one '0' and at most z '0's to y'_0, given n we have:

$$(n-1) + z \leq o \leq nz \tag{5.6.15}$$

Since $m = y' - o$, thus $m_{min} = y' - nz$. However, note that we need at least n symbols '1' to separate n runs of 0's, thus $m \geq n$. Therefore, $m_{min} = \mathsf{max}\{y' - nz, n\}$.

Meanwhile, $m_{max} = y' - n + 1 - z$ but note that $m \leq a$, thus we obtain $m_{max} = \mathsf{min}\{a, y' - n - z + 1\}$. □

As observed and analyzed in Section 4.5 - Chapter 4, when $z > 1$, we must distinguish runs of symbols '`0`':

- set $\mathbb{U}$ of u runs whose run-length $l_i \geq z$;
- set $\mathbb{V}$ of v runs whose run-length $1 \leq l_i < z$;

Using the same analyzing process as used in Section 4.5 - Chapter 4, we obtain some further bounds.

Given m satisfying (5.6.14), then:

$$1 \leq u \leq \mathsf{min}\{n, \lfloor \frac{y' - m}{z} \rfloor\} \tag{5.6.16}$$

and

$$\lceil \frac{b_v}{z-1} \rceil \ \leq v \leq \ \mathsf{min}\{b_v, n - u\} \tag{5.6.17}$$

where

$$b_v = y' - m - uz \geq 0 \tag{5.6.18}$$

is the number of '`0`' belonging to all runs $v \in \mathbb{V}$.

Given a pair of (u, v) satisfying (5.6.16)-(5.6.17), we turn to count the number of ways to:

(i1) distribute m symbols '`1`' into n runs where no empty is allowed, which yields the number $\binom{m-1}{n-1}$.

(i2) distribute b_v symbols '`0`' into v runs such that no empty allowed and each run has maximum $(z-1)$ symbols '`0`'. This is the number C_{b_v} given in eq. (4.5.10) in Section 4.5 - Chapter 4.

(i3) distribute b_u symbols '`0`' into $u + 1$ runs with some clear constraints.

(i4) arrange runs of $\mathbb{U}$ and runs of $\mathbb{V}$.

For items **(i3)** and **(i4)**, we need a little further understanding of the case we are counting. Let us examine Fig. 5.18 where two special runs of `0`'s are shown. The first run does not contribute any *unblocked* position. This run belongs to neither $\mathbb{U}$ nor $\mathbb{V}$. Imagine that there is one virtual '`0`' permanently placed in this run, then this run is never empty while we do counting.

The other special run is the last one that takes extra z positions. This run always belongs to $\mathbb{U}$ since its minimum length is z. Since this run $\in\mathbb{U}$ has a fixed position, the number of ways to do item **(i4)** is $\binom{u-1+v}{v}$.

Next, we find b_u, the number of '`0`' that is later distributed in all runs of $\mathbb{U}$ plus the first run:

$$\begin{aligned} b_u &= (y' - m - b_v) + (y - y') + z + 1 \\ &= y + z + 1 - m - b_v \end{aligned} \tag{5.6.19}$$

We then put $(z-1)$ symbols '`0`' in each run $\in \mathbb{U}$ in advance. The remaining $b_u - u(z-1)$ symbols '`0`' are distributed in $u+1$ runs such that no run is empty. Thus, we obtain the number of ways to do item **(i3)** (after substituting b_u and b_v):

$$C_{b_u} = \binom{b_u - u(z-1) - 1}{u + 1 - 1} = \binom{y - y' + u + z}{u} \tag{5.6.20}$$

Besides, note that the other $(a - m)$ available time-frames of hop h are blocked by distributing them into $(K - y)$ *blocked* positions, yielding the number $\binom{K-y}{a-m}$. Hence, for a pair of valid (u, v), the number of ways to generate y' *unblocked* positions at hop h is given:

$$\begin{aligned} C(u,v) &= \binom{u-1+v}{v}\binom{m-1}{u+v-1} \\ &\times \binom{y - y' + u + z}{u} C_{b_v} \binom{K-y}{a-m} \end{aligned} \tag{5.6.21}$$

where C_{b_v} is given in eq. (4.5.10) (Section 4.5 - Chapter 4).

Summing up $C(u, v)$ for all pairs of (u, v) yields $C(y, y'|n)$ for a given n:

$$C(y, y'|n) = \sum_{\text{all } m} \sum_{\text{all } u} \sum_{\text{all } v} C(u, v)\Big|_{u+v=n}$$

Summing up $C(y, y'|n)$ for all n yields $C(y, y')$:

$$C(y, y') = \sum_{n=1}^{\lfloor \frac{y'-(z-1)}{2} \rfloor} C(y, y'|n) \tag{5.6.22}$$

$$= \sum_{n=1}^{\lfloor \frac{y'-(z-1)}{2} \rfloor} \sum_{m=\max\{y'-nz,n\}}^{\min\{a,y'-n-z+1\}} \sum_{u=1}^{\min\{n,\lfloor \frac{y'-m}{z} \rfloor\}} \sum_{v=\lceil \frac{bv}{z-1} \rceil}^{\min\{b_v,n-u\}} C(u, v)\Big|_{u+v=n}$$

We are now ready to extend Theorem 5.6.1 for the case $z > 1$.

Theorem 5.6.6. *For $z > 1$, the entry $p_t(y, y')$ of the matrix $\mathcal{P}$ is given by:*

$$p_t(y, y') = \begin{cases} 1.0 & \text{, if } y = y' = 0 \\ \binom{K-y}{a}/\binom{K}{a} & \text{, if } y' = 0 \ \& \ z < y \\ C(y, y')/\binom{K}{a} & \text{, if } z < y \leq K \ \& \ z < y' \leq y + z \\ 0.0 & \text{,otherwise} \end{cases} \tag{5.6.23}$$

where $C(y, y')$ is given in (5.6.22).

Deriving vector $\mathbf{B}_0$ for $z > 1$

Each entry of the initial vector $\mathbf{B}_0$ for $z > 1$ is computed by:

$$Pr(\beta_0 = y) = \begin{cases} C(x)/\binom{K}{a} & \text{, if } z + 1 \leq y \leq \min\{a(z + 1), K\} \\ 0.0 & \text{, otherwise} \end{cases} \tag{5.6.24}$$

where $x = K - y$ and $C(x)$ is given by eq. (4.5.11) in Section 4.5.1 - Chapter 4.

The formal proof for the upper bound when $z > 1$ follows the same approach as used in Section 5.6.4. However, the mapping operator is very complex and we skip the presentation for it.

5.7 Evaluations

The computed bounds are evaluated by comparisons between numerical results and simulation results. Simulations are based on the Monte-Carlo approach. Given parameters K and H, a matrix is generated. Then each row of the matrix representing a hop is loaded uniformly until b time-frames out of K are set to be '0'. An attempt to find a schedule from the first hop until the last hop is made with the rule strictly follows a certain z-forwarding scheme that all switches along the route deploy. If no schedule is found, a blocking event is counted. After each attempt to find a schedule, the matrix is reset and reloaded. For each simulated point in graphs, 10×10^6 iterations are tried.

In all graphs, the upper bound of time-blocking probability, p_u, is computed based on the analysis presented in Section 5.6. The lower bound of time-blocking probability p_l is computed using the analysis derived in Section 5.5. We also compute the approximation curves using the following equation:

$$p_{app} = \sqrt{p_u \; p_l} \tag{5.7.1}$$

To obtain high precisions of numerical results, we use MAPM library [55] for computing.

We notice that the gap (we call it error) between a simulation curve and a bound one slightly increases as the more possible scheduling delay is allowed (from $z = 1$ to $z = 3$). However, the load difference between two curves is in a scale of 1% to 2%.

Though the individual error of either upper bound or lower bound increases as either the hop-length or the scheduling delay increases, we still capture a very good accuracy of the approximation computed through eq. (5.7.1). This is because the two errors eliminate each other since they grows in opposite direction.

We also notice that there is a significant improvement of time-blocking probability if there is one or two scheduling delay is allowed ($z = 1$ or $z = 2$ vs. $z = 0$). However, the gain is not as that much if we increase z (e.g., $z = 3$ vs. $z = 2$). This is a consent with the conjecture that only very few buffering is required in TDS-FλS networks.

Fig. 5.22-5.23 show blocking probability performances when there are multiple channels per hop. Obviously, FλS allows to obtain very high throughput network. For example, a good combination of two dimensions (time and wavelength) such as $z = 2$, $C = 16$ is good enough to maintain the blocking performance around 10^{-3} even the normalized load reaches 0.8.

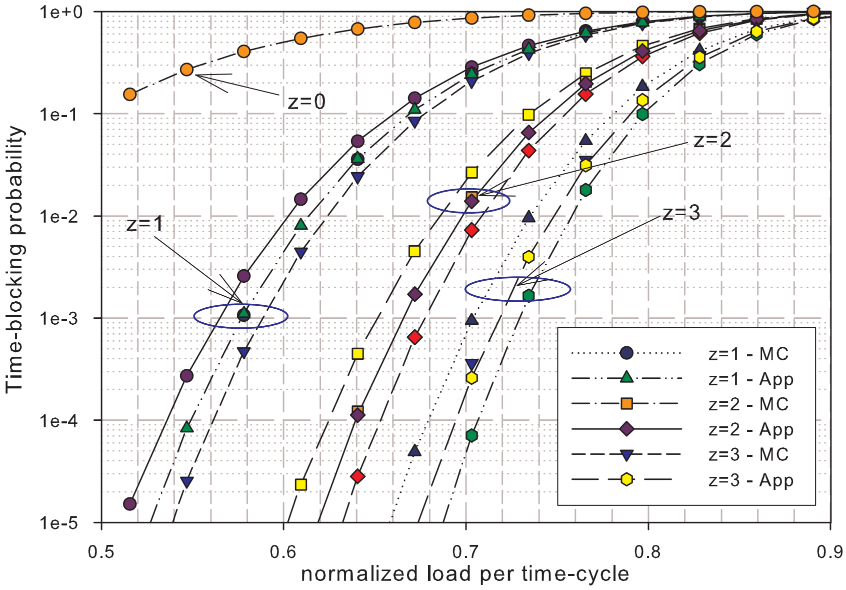


Figure 5.19: Evaluations for 5 hops path, $K = 64$, and z varies.

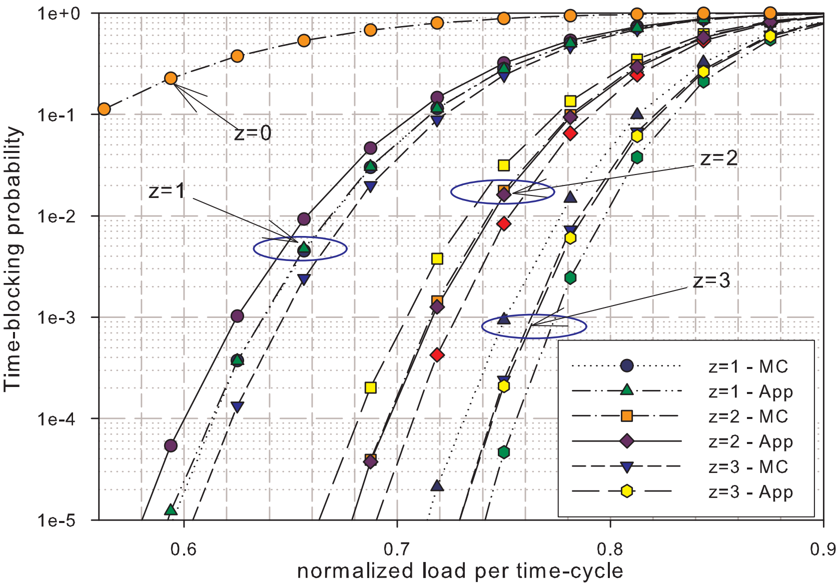


Figure 5.20: Evaluations for 5 hops path, $K = 128$, and z varies.

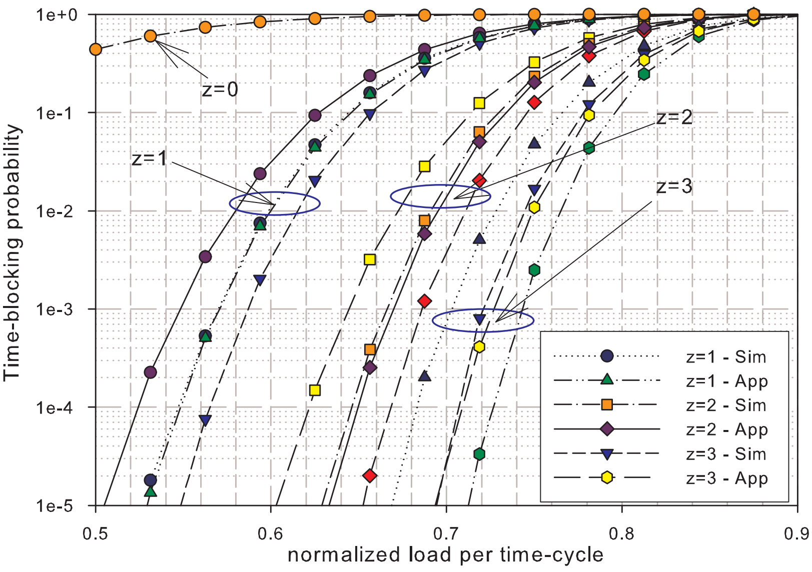


Figure 5.21: Evaluations for $H = 7$, $K = 128$, and z varies.

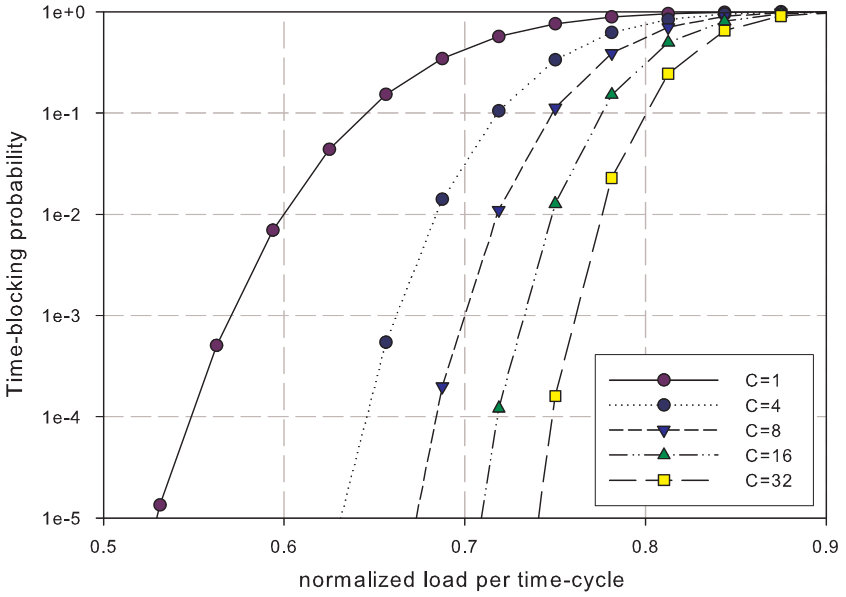


Figure 5.22: Blocking performance for $H = 7$, $K = 128$, and $z = 1$ and number of channels per hop varies.

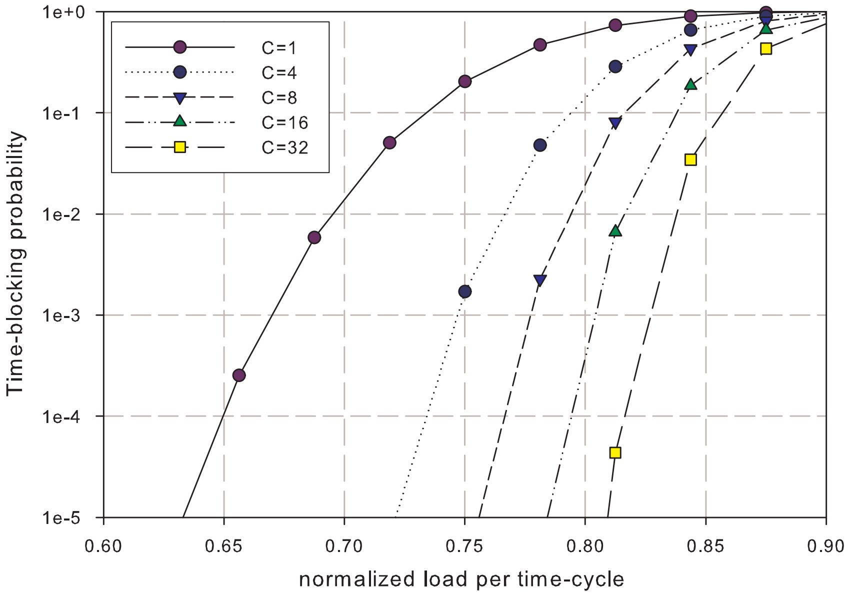


Figure 5.23: Blocking performance for $H = 7$, $K = 128$, and $z = 2$ and number of channels per hop varies.

Chapter 6

Prototype and test-bed

Previous chapters are dedicated for various theoretical works, including some node designs based on the use of tunable lasers; scheduling feasibilities; and a thorough analysis about the time-blocking issue. In this chapter, we close our work by introducing the first time-driven switching (TDS) ever implemented in our lab at DIT-University of Trento.

Particularly, this chapter focuses on UTC-based FPGA (field programmable gate-array) controller that was implemented for controlling the TDS switch prototype. The switch controller facilitates dynamic configuration of ultra scalable switch. The prototype is implemented using off-the-shelf components. Preliminary results point out that TDS architecture is especially suitable to support high capacity streaming media applications over the Internet.

6.1 Field programmable gate-arrays

The field programmable gate arrays (FPGAs) have gone main stream. The newest generation of FPGAs have hit performance and cost goals which allow a much wider spectrum of applications support. Now a day, flexibility and large number of functions are needed in each protocol layer for providing quality of service in data networks, but current networks are far from rich in terms of flexibility. To help remedy this situation, we have designed and implemented a FPGA based switch controller for ultra scalable TDS architecture. A switch controller is the brain of TDS architecture. The function of controller is to dynamically configure the switch.

Bandwidth of optical fiber is growing at faster rate than the speed of silicon therefore switching scalability is an issue. Usually, a network processor (NP) is used for switching purpose in the network. NP acts as a traffic manager, which occupies the space between a network interface and a switch fabric in a switch/router. NPs are specialized CPUs optimized to support the implementation of network protocols at the highest possible speed. The overarching emphasis on speed results in unconventional hardware architectures that create new challenges for the software engineer. For example, Ciscos top-of-the-line router with a novel NP design, the CRS-1, has 640 $Gbit/s$ per chassis [the announcement of 92 $Tbit/s$ should be divided by 2 (for counting input and output separately) and then by 72 chassiss], which represents a factor of 2 improvement after 5 years of development with 500 million dollars of investment. So, if the internet traffic is doubling, say, every 18 months there is a real switching bottleneck on the horizon.

The implemented FPGA based optoelectronic switch controller for TDS switches is a simple and low cost alternative to network processor. The TDS switch architecture guarantees deterministic QoS for streaming media over the Internet and is scalable to speed of 10-100 $Tbit/s$.

Some applications of FPGA in communication network have been reported in the literature. In [4], the integration of FPGA-based controller with optical transponder was suggested. In [45], authors reported a telecommunication oriented FPGA for implementing programmable ATM adapters. A high speed serial transceiver running at sub-nominal rate to recover data is presented in [66]. A complex FPGA-based controller for contention resolution in an optical packet router was reported in [76]. In fact, the lack of optical random access memory (ORAM) and all-optical processing technologies and the use of fiber delay link for storing and forwarding caused the complexity of the work in [76].

In this chapter we present the FPGA-based controller for dynamic configuration of ultra scalable TDS switch fabric. Details of other parts of prototype are out of scope of the chapter, though the complete prototype has been briefly described in subsequent sections.

The test-bed presented in this chapter aims to realize the timing and forwarding principles of TDS/FλS, which is thoroughly discussed in Section 2.2 of Chapter 2. Section 6.2 describes scalable TDS switching architecture with emphasis on FPGA based switch controller, Section 6.3 presents experimental demonstration of the switch prototype, Section 6.5 discusses the presented work and open issues.

6.2 Ultra scalable switch architecture

The functional diagram of a TDS is shown in Fig. 6.1. It has three major parts: a global positioning system (GPS) time receiver; a switch controller (or simply controller); and a switch fabric.

The switch fabric is a set of interconnected switching boards, which can be connected in various forms (e.g., matrix, single stage, multistage etc). Operations of the switching fabric is controlled by the switch controller.

GPS receiver is connected to GPS antenna, which is mounted externally facing the open sky (not shown in the Fig. 6.1). The communication between controller and switching boards can be either parallel or serial.

There are multiple control signals such as I/O enable, switching board status and so on. However, for the sake of simplicity, we only show three important classifications: address, data path and strobe signal.

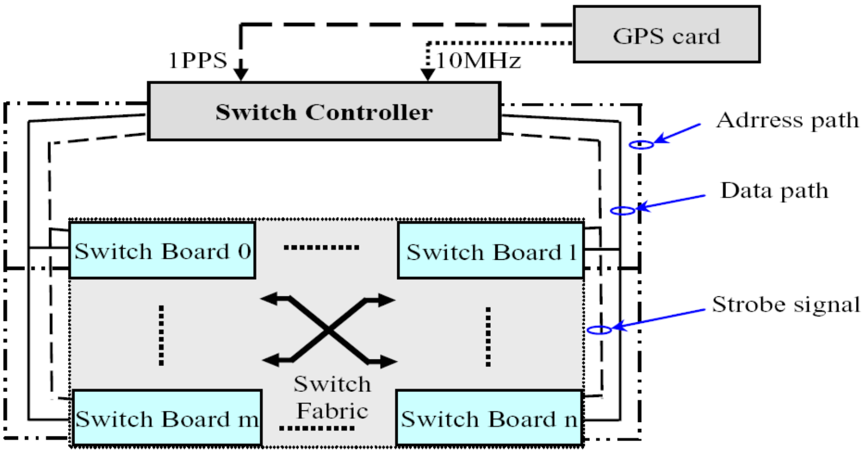


Figure 6.1: Functional block diagram of a TDS switch.

Control data (including output register addresses and their input channel selections for every time-frame) are stored in the memory of the controller. The data and address path are used to transfer controlled register address and input channel selection, which are then written in registers of main-switching boards. The writing process must end before the falling edge of the strobe signals, which corresponds to the start of the next time-frame. At the falling edge of strobe signal, a new switching configuration is latched on all switching boards. The new switch configuration is ready in less than 10 ns.

6.2.1 GPS receiver

The GPS receiver, which is EPSILON Board OEM II [58], provides accurate and stable time and frequency signals for synchronization. It provides 1PPS (pulse per second) and 10 MHz sine wave, and time-of-day output. Furthermore, the 10 MHz frequency reference is always locked to the 1PPS, which is the standard Universal Time Coordinated (UTC) second. This implies that within 1PPS there are exactly 10,000,000 cycles of the 10 MHz output from the GPS card.

6.2.2 Mindspeed switch board

Mindspeed switch board [44] - the primary component of switching architecture is a low-power complementary metal oxide semiconductor (CMOS), high-speed 144x144 cross point switch with integrated clock data recovery (CDR), input equalization, and built-in system test and broadcasting features. Each CDR is preceded by a programmable input equalizer (IE). The IE removes inter symbol interference (ISI) jitter usually caused by printed circuit board (PCB) skin effect losses. It offers programmable switch configuration to switch off unused portion thus reduces power consumption. Each CDR can be independently bypassed and turned off if not in use.

6.2.3 Switch controller

Opal Kelly XEM3001 module [31] is used for implementation of switch controller. The XEM3001 consists of an electrically erasable programmable read only memo (EEPROM), a universal serial bus (USB) 2.0 microcontroller, a phase-locked loop (PLL), a 400,000-gate Xilinx Spartan-3 field programmable gate array (FPGA) submodule and 1 MHz to 150 MHz multi-output clock generator. The block diagram of implemented FPGA-based controller is shown in Fig. 6.2. Very high speed integrated circuit hardware description language (VHDL) is used for the implementation of the FPGA sub module controller.

In Fig. 6.2, main FPGA submodule blocks are control logic (CTRL), memory table, GPS Interface and Mindspeed switch controller. The implementations of all functional blocks in VHDL are synthesized in a single bit file that is stored on a PC. This file can be easily uploaded to the FPGA through a USB connection. This implementation provides flexibility for updating the controller with new versions of VHDL program.

The bit file is downloaded to configure the FPGA every time the con-

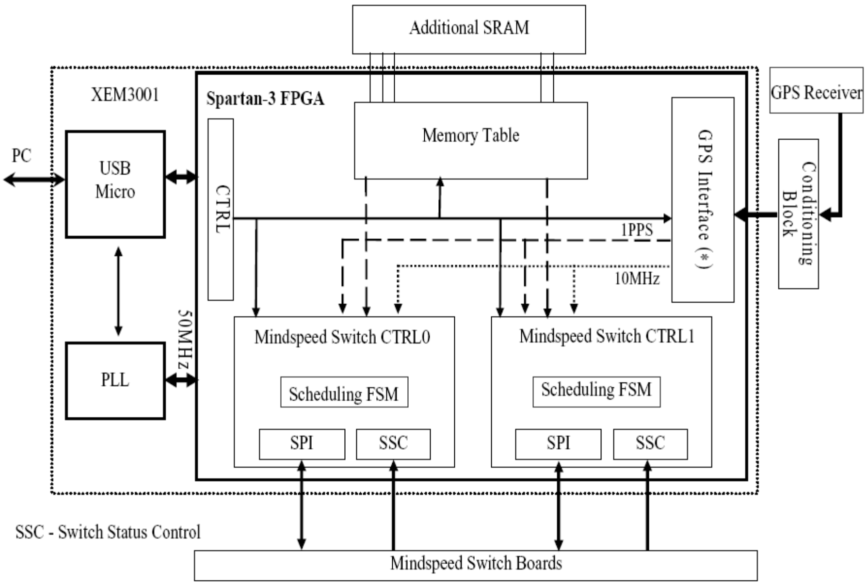


Figure 6.2: Functional block diagram of a TDS switch.

troller is started. In the prototype presented in this chapter, we have used one controller for controlling two Mindspeed switch boards. There are two scheduling finite state machines (FSMs), each for a board respectively. A master clock for all blocks of Spartan-3 FPGA submodule is taken from an external PLL chip. The GPS interface communicates with the GPS receiver mentioned in Section 6.2.1 via a serial link. The GPS interface also allows remote configuration and status reporting. Two signals, 10 MHz sine-wave and 1PPS, are used as clock sources for the scheduling FSMs. GPS status control sub-block are used to monitor correct locking of the receiver with signal from GPS.

The Conditioning Block composed of discrete electronics required to condition clock signal coming from GPS receiver to make it compatible with CMOS pins of the Spartan-3 FPGA submodule. USB control block (CTRL) is the interface between PC and the internal sub-blocks configuration registers implemented in other sub-blocks on Spartan-3 FPGA. Memory table and Mindspeed switch control sub-blocks are described in following subsections.

Memory table

Memory table is in a matrix format and stores switching configurations of all channels and for all time-frames. The size and structure of the table depends on timing parameters and number of switching channels. Timing parameters are time-frame duration, number of time-frames per time-cycle, number of time-cycles per super-cycle, which is equal to one UTC second.

Memory table can be configured from the graphical user interface (GUI) on PC through the USB communication link. For complete switch configurations (i.e., 144 channels per switching board, with multiple boards), 216-kbits of block RAM [8] embedded in the FPGA will not be enough then an external SRAM should be added. For future development with network protocols, memory table will be updated according to signaling protocol between distant TDS switches. The signaling issue is outside the scope of this chapter.

Mindspeed switching board controller

The block Mindspeed Switch CTRL0/CTRL1 represents a FSM that controls and connects all the sub-blocks implemented on FPGA module. At every clock cycle it reads the input status register connected to a USB wire. This block consists of scheduling FSM, serial peripheral interface (SPI) and Switch Status Control sub-blocks. These blocks are detailed in the following descriptions.

Scheduling FSM

One scheduling is required for each Mindspeed Switch Board and number of FSM modules can be implemented for number of switching boards. There are three important counters: time-frame counter, time-cycle counter, and

UTC second counter. There are two reference input signals to controller through GPS receiver. One is 1PPS and the other is 10 MHz. Every UTC second (1PPS) is divided into number of time-cycle and every time-cycle is divided into number of time-frames. The base unit of time is given by 1/10 MHz=100 ns and it is a reference for counters.

All counters are 16-bit implementation. The time-frame duration is the integer multiply of time resolution. The number of time-frame, time-cycle and time-frame duration can be modified by changing the corresponding parameters stored in registers, which are accessible via USB from PC. The strict criterion is that the product of time-frame duration and number of time-frames and number of time-cycles must be equal to one UTC second.

At the beginning of every time-frame the FSM downloads a new switching configuration cyclically to the Mindspeed switch board and activates hardware strobe via SPI. All the data are stored into a memory table which is read sequentially. When a new time-cycle starts, the pointer in the memory table is reset to the first memory location. This is the cyclic operation of the controller. 1PPS pulse resynchronizes all the FSMs. If the download terminates before the 1 Hz clock goes high the FSM stops in a waiting status until the 1PPS clock reset and restart whole process. It means any malfunction will last maximum one UTC second.

Fig. 6.3 shows simulated hardware strobe generated by time-frame counters using 10 MHz GPS clock. The right side of Fig. 6.3 shows the effect of re-synchronization with the UTC 1PPS. Various switch functions are accessed through 8-bit registers (memory), which are resolved by the 10-bit address bus. The contents of the registers are transferred via an 8-bit data bus during a read and write operations.

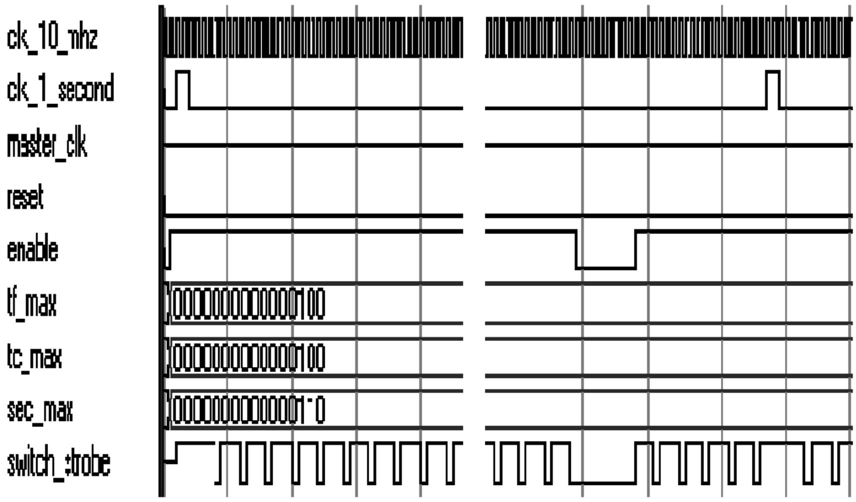


Figure 6.3: Timing diagram of a FSM.

Serial parallel interface

SPI is used to transfer data between PC and Mindspeed boards in both the directions through USB. This block implements a FSM that meets the timing and shape specifications given by Mindspeed M21151 data sheet [44]. The maximum input clock is 50 MHz and SPI operate at 25 MHz (i.e., a half of input clock).

Data writing using SPI takes 20 SPI clock cycles (1 start bit, 1 RD/WR bit, 10 address bits, 8 data bits), which is 0.8 μs for downloading one channel configuration. For a configuration with 100 μs time-frame length, a single FPGA implemented SPI peripheral is able to load up to 125 channel configurations. When more number of switch configurations loading is required, it is possible to programme SPI clock up to the maximum speed (25 MHz), which is supported by Mindspeed board.

More than one SPI peripherals can be implemented in the same FPGA for increasing the speed and for scalability. For a large system with multiple switch boards, a 18-bit parallel interface has been implemented and tested. The high downloading speed of 100 MHz (i.e., 10,000 switch configurations can be downloaded in 100 μs time-frame duration). This incentive comes at the price of complex interconnections of switch fabric.

Switch status control

Switch status control (SSC) sub-block checks the loss of signal (LOS) status of the board. A LOS circuit is included on each input and detects whether valid data is present or not. LOS acts as an alarm and can be used to inhibit the signal into the switch core when the data to the input terminal is lost. If the input signal is clamped high or low, or if the difference between the input data rate and the programmed data rate is greater than approximately ± 100 $Mbit/s$, the LOS alarm will be activated.

Software interface

The interaction between a PC and an FPGA board consists of two layers. First layer software is based on C++ classes of Opal Kelly library [31]. This type of interface uses more than one 8-bit pipes in order to transfer data. The second layer is a simple graphical user interface (GUI), developed using VC6++ [18]. The GUI is a set of dialog boxes that react to events like switch start/stop, input/output channel connection configuration. Switch configurations can be easily edited and updated by users.

6.3 Prototype Implementation and Testing

The complete prototype setup photograph diagram is shown in Fig. 6.4. The prototype major components are streaming media sources (audio, video and text); a network interface for packets scheduling; 25 km single mode optical fiber; and multiple Mindspeed switch fabrics (each with capacity of 144 channels of 3.2 $Gbit/s$ with total switching capacity of 420 $Gbit/s$).

The source side switch fabric is a two-stage network implemented by connecting two Mindspeed switches. Two switches are controlled and configured by a single FPGA switch controller. On receivers side of the 25 km single mode optical fiber, there is another switch fabric consists of one Mindspeed switch associated with another FPGA switch controller. Two receivers for receiving and playing the two movies with sound and subtitles. Standard single mode and multimode transceivers have been used for optical interconnections between various electrical-to-optical and vice versa conversion.

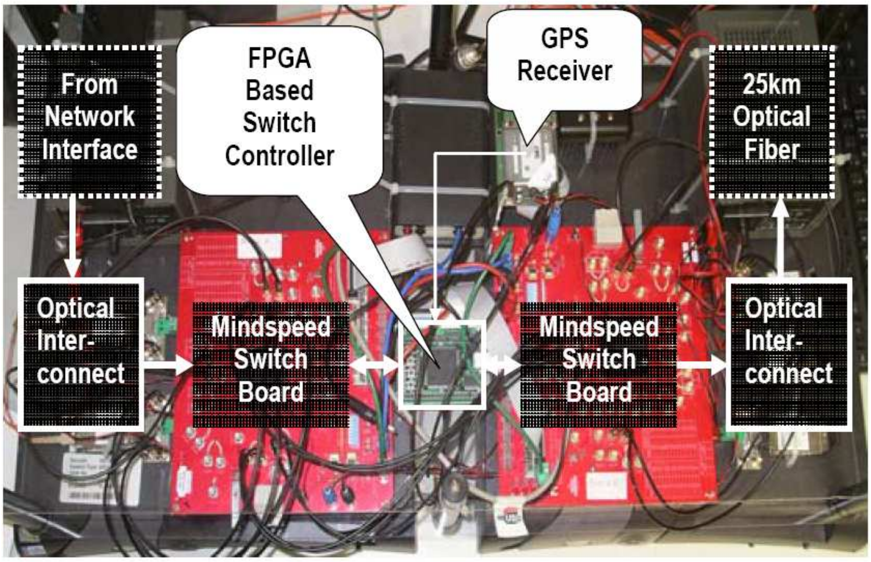


Figure 6.4: A photograph of the prototype.

The switch prototype experimental setup is shown in Fig. 6.5. Two media streams destined for two different receivers: one DVD movie with soundtrack and subtitles; the other animation movie with soundtrack. The streams are transmitted from one source PC (shown IP Stream) using VLC media player [2].

Asynchronous packets are sent to a network interface (the detail of network interface is out of scope of this paper), which schedules incoming packets so that they are forwarded in synchronization with 1PPS signal from GPS through the GPS antenna (not shown in Fig. 6.5).

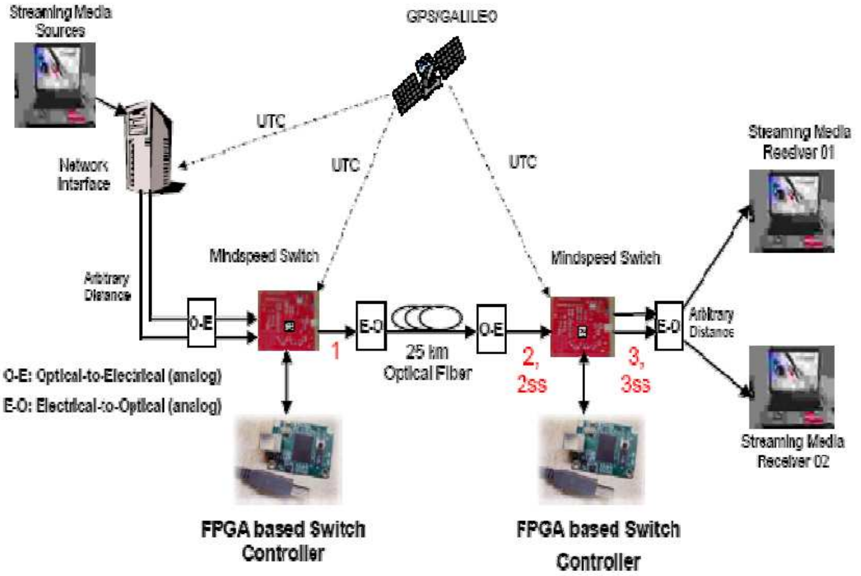

Streaming Media Sources
GPS/GALILEO
UTC
UTC
UTC
Network Interface
Streamng Media Receiver 01
Mindspeed Switch
Mindspeed Switch
Arbitrary Distance
O-E
E-O
O-E
E-O
Arbitrary Distance
1
25 km Optical Fiber
2, 2ss
3, 3ss
O-E: Optical-to-Electrical (analog)
E-O: Electrical-to-Optical (analog)
Streaming Media Receiver 02
FPGA based Switch Controller
FPGA based Switch Controller

Figure 6.5: The prototype setup.

The media streaming packets from two sources are forwarded via an optical link by the network interface through transceivers to source side switch fabric during different predefined time-frames. The packets are split into two streams by the first switching stage.

These separate streams from the first stage are forwarded to the second stage through electrical connections. At the second stage both streams are again mixed by the cross point switches. Then the mixed stream is transmitted to receivers side through transceiver and 25 km single mode optical fiber link. On the receiver side mixed stream received through transceiver is separated into two streams by switching. Separated streams are forwarded to two receivers. Switching of all three cross point switches and network interface are synchronized with 1PPS received from GPS.

6.4 Experimental results

The eye pattern test is a quick method for visually examining the quality of serial signals, e.g., the amount of timing jitter and amplitude variation in a serial data streams. A synchronous clock and/or the clock recovered from the data, triggers the oscilloscope. In one captured screen, all possible signal transitions of the signal are displayed: positive-going, negative going, leading, and trailing. This single display provides information about the eye opening, noise, jitter, rise and fall times, and amplitude.

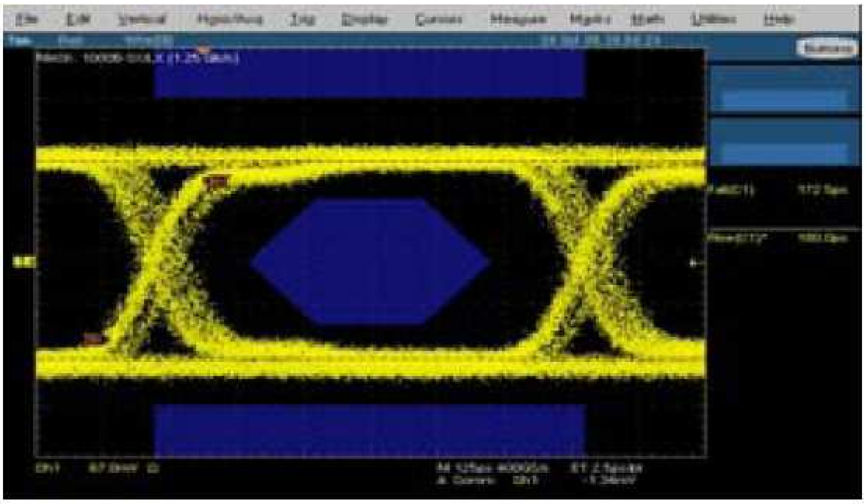


Figure 6.6: The eye pattern with mask: 1000BSX/LX (1.25 *Gbit/s*) at location 1.

The two-dimensional shape can easily be compared to a standard mask. Fig. 6.6-6.10 shows actual eye diagrams as captured on a real-time Tektronics [14] TDS 6604, 6 GHz digital storage oscilloscope. The oscilloscope has built in standard Gigabit Ethernet 1,000 base SX-LX mask test. The measurements are carried at various locations (1, 2, and 3) indicated in Fig6.5. At point 2 and 3, measurements were repeated using 25 km single mode fiber (2ss, 3ss). Note that the boundaries of all these eye diagrams, which pass the compliance test, are within the ranges expressed by masks (in dark blue color). Moreover there is sufficient margin between the signal

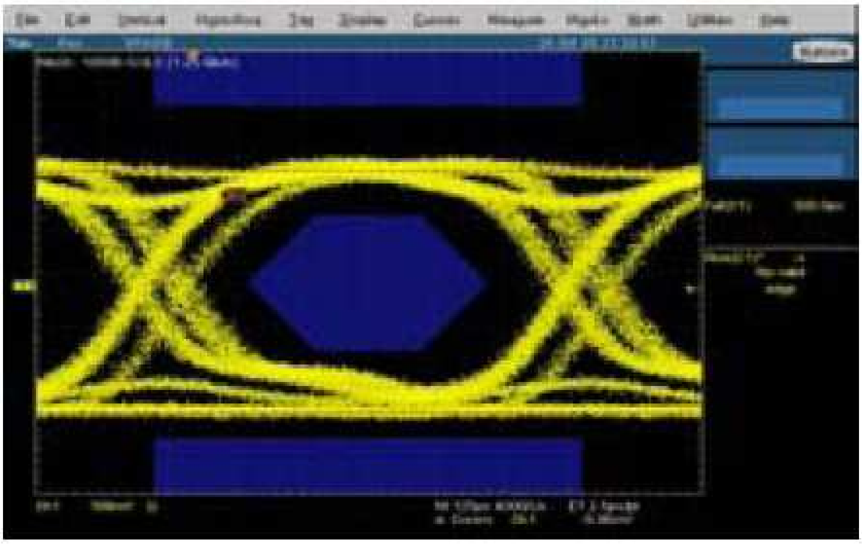


Figure 6.7: The eye pattern with mask 1000BSX/LX (1.25 *Gbit/s*) at location 2 with multi-mode fiber.

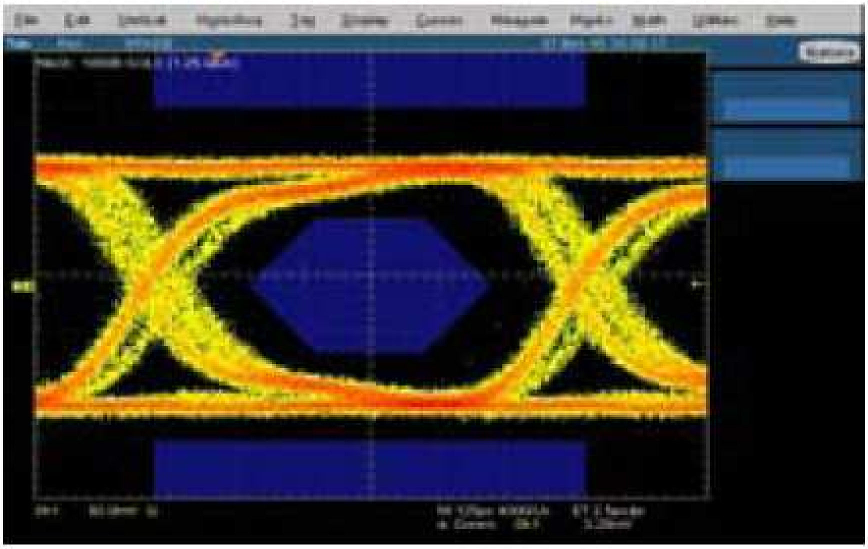


Figure 6.8: The eye pattern with mask 1000BSX/LX (1.25 *Gbit/s*) at location 2 with single mode fiber.

and the mask.

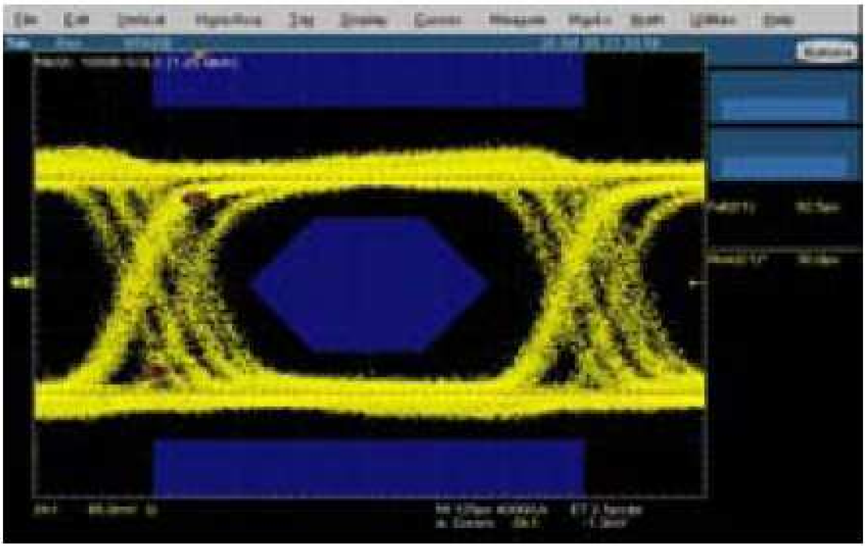


Figure 6.9: The eye pattern with mask 1000BSX/LX (1.25 *Gbit/s*) at location 3 with multi-mode fiber.

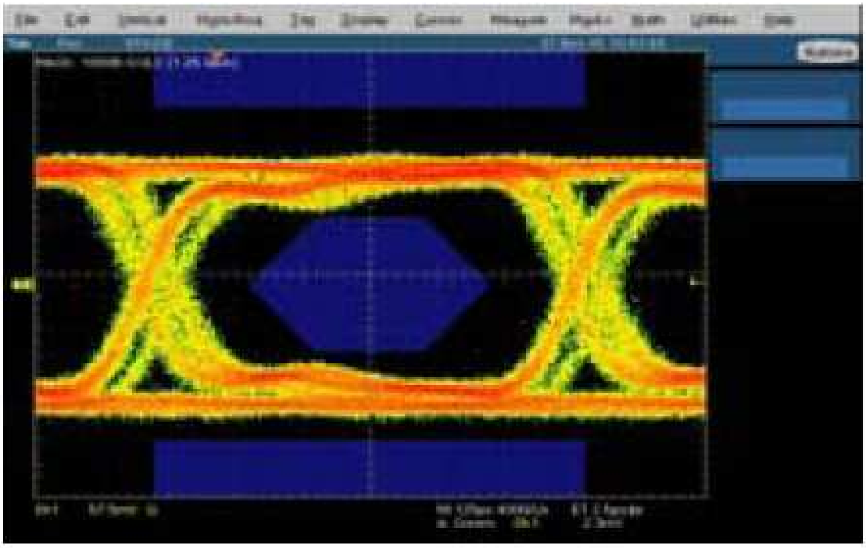


Figure 6.10: The eye pattern with mask 1000BSX/LX (1.25 *Gbit/s*) at location 3 with single mode fiber.

6.5 Discussions

This chapter presented the implementation and testing of a low cost ultra scalable TDS switch prototype. The FPGA based controller for dynamically configuring the ultra scalable IP-packet switching is the main logic component of this prototype. The beauty is that the simplicity of this realization did not compromise two most desired performance properties for the future Internet: (1) switching scalability up to more than 10 $Tbit/s$ in a single chassis and (2) predictable QoS performance for streaming media and large (content) file transfers. There is some delay between the source and the receiver streaming media play. This delay is contributed by propagation delay and mainly due to buffering in the media player because the media player is designed for streaming media over asynchronous IP packet switching network. Designing and using the TDS friendly media player will reduce the delay significantly.

There are a few open issues that require further investigation in order to implement even larger TDS switch. One of the issues is the scalability of the FPGA switch controller for controlling large number of channels and switching components. Another issue is corresponding to the FPGA memory requirements for storing configurations for all channels and time frames. Memory is scalable because there is a provision for expanding the internal Spartan-3 RAM using external SRAM BLOCK for adding external SRAM chips.

Another important aspect is time taken by controller to configure very large matrix of switching elements for supporting larger number of optical channels. There are at least two possible solutions. One is the implementation of multiple SPIs on the same FPGA for configuring multiple switching elements simultaneously. The use of multiple SPIs for connecting many Mindspeed switching chips increases the scalability of the system. Another way is the development of dedicated parallel connectors for fast configuration of large number of Mindspeed switches. But this speed may come at the price of complex interconnection.

Other open issue is time-frame alignment when the delay between two switching channels is not integer multiple number of time frames. To overcome it time-frame alignment module is necessary [4]. Bit synchronization is another issue that requires further investigation in order to determine the most appropriate solution.

Chapter 7

Conclusion

TDS/FλS introduces a truly novel approach to networking fields. In TDS/FλS, timing issues are fully addressed to obtain an accurately synchronous network worldwide with lowest costs. Once all network nodes are strictly synchronized, the well-known pipeline forwarding principle is possible, thus helping eliminate one of the most expensive operations at routers/switches - the packet header processing. This in turn helps remedy a scalability issue of current router/switch architectures. Moreover, QoS emerges as a bonus in TDS/FλS thanks to pipeline forwarding schemes. This thesis contributes some important research results to favor this promising networking technology - the TDS/FλS.

7.1 Towards the reality of sub-wavelength switching

Various theoretical and experimental attempts have been contributed to realize all-optical sub-lambda switching. Some being abundantly mentioned are slotted/unslotted optical burst switching, time-wavelength interleave networking, time-slot interchange with WDM, etc. Still none of mentioned technologies has been realized. These technologies require either a sophisticated asynchronous control plain or lot of buffering to resolve contentions in order to obtain some moderate blocking performance. Several others require a strict network synchronization but fail to obtain it properly.

Tunable laser - an advanced optical device is emerging and shifting fast towards high speed applications. Within few more years, stable and wide-range tunable lasers operating at ns speed are expected to be commercialized. At that time, the integration of this advance device into TDS will introduces to FλS-the technology that enables a true sub-wavelength switching capability. This helps leverage the power of DWDM technology at a higher level towards the networking area. Consequently, we introduce some novel designs described in chapter 3. We consider this contribution as one among many possibilities towards the realization of sub-wavelength switching in the optical networking domain.

7.2 High throughput network with few buffering

The combination of time, space, and wavelength dimensions in FλS makes it possible to reach high throughput while maintaining low blocking performance. Some initial efforts and results of the blocking performance presented in chapter 4 and 5 have confirmed this property.

Conventionally, analyzing blocking probability has been done thoroughly at a call level for many different networks. We tackle the problem in a different way. The analysis shown in chapters 4 and 5 mostly based on combinatorics and probability. Given a certain link load level, the analysis of time-blocking probability involves in counting various possible combinations of busy, blocked, available, and schedulable time-frames under various strict constraints. In fact, nonzero scheduling delays allow more flexibility in scheduling time-frames and thus help reduce the blocking probability. However, they also create a higher level of complexity that makes the analysis untraceable for large systems (in terms of the number of time-frames per time-cycle). An approximation with lower bound and upper bound has been derived in chapter 5.

Various numerical results have shown that allowing one or two time-frame scheduling delays significantly reduce the blocking probability in TDS/FλS. Another major remark we should mention is that much further improvements can be obtained if we increase the number of time-frames per time-cycle and in the meantime take the advantage of DWDM technology at link layer by allowing multiple channels per link/hop. All in all, we believe that a very high throughput networks based on FλS technology with very few buffering per node is practical.

The study has not been completed in the sense that only strictly non-space blocking switch is considered in all cases. In practice, space blocking fabrics such as multi stage Banyan structures are frequently met, thus the combination of space and time blocking are more interesting and deserved further studies. Besides, another dimension to extend the research is the blocking analysis for cases when wavelength swapping is possible. As discussed and proposed in chapter 3, full wavelength swapping can be implemented using tunable lasers or wavelength converters. Another possibility to obtain full wavelength swapping is the use of electronics switches as we implemented in the prototype described in chapter 6.

Bibliography

[1] *ITU-T Recommendation G.872.*

[2] Vlc player. `http://www.temex.com`.

[3] J.K. Aikio. *Extremly short external cavity (ESEC) laser devices: Wavelength tuning and related optical characteristics.* PhD thesis, VTT Technical Research Center of Finland, 2004.

[4] A. Aloisio, F. Cevenini, and V. Izzo. An approach to dwdm for real-time applications. *IEEE Transactions on Nuclear Science*, 51(3):526–531, 2004.

[5] J.D. Angelopoulos et al. Slotted optical switching with pipelined two-way reservations. *IEEE/OSA Journal of Lightway Technology*, 24(10):3616–3624, 2006.

[6] M. Baldi and Y. Ofek. Fractional lambda switching. *IEEE Proc. of ICC*, 5:2692–2696, 2002.

[7] M. Baldi and Y. Ofek. Fractional lambda switching - principles of operation and performance issues. *SIMULATION: Transactions of The Society for Modeling and Simulation International*, 8(10):527–544, 2004.

[8] A. Bianciotto et al. Fast and efficient fault-recovery strategies in the wonder metro architecture. *Proc. of 10th European Conference on Networks & Optical Communications*, 2005.

[9] A. Bianco et al. Measurement-based reconfiguration in optical ring metro networks. *IEEE/OSA Journal of Lightway Technology*, 23(10):3156–3166, 2005.

[10] A. Bianco, G. Galante, E. Eionardi, and M. Mellia. Analysis of call blocking probability in tdm/wdm networks with transparency constraint. *IEEE Communications Letters*, 4(3):104–406, 2000.

[11] A. Bilenca et al. Broad-band wavelength conversion based on cross-gain modulation and four-wave mixing in inasinp quantum-dash semiconductor optical amplifiers operating at 1550 nm. *IEEE Photonics Technology Letters*, 15(4):563–564, 2003.

[12] S. Bregni, D. Carzaniga, and R. Gaudino. Slot synchronization strategies in optical slotted rings: the wonder approach. *IEEE Proc. of ICC*, 2006.

[13] A. Busic and J.M. Fourneau. A matrix pattern compliant strong stochastic bound. *Proc. of the Symposium on Applications and the Internet Workshops (SAINT-W05)*, 2005.

[14] J. Buus and E.J. Murphy. Tunable lasers in optical networks. *IEEE/OSA Journal of Lightway Technology*, 16(5):5–11, 2006.

[15] A. Carena et al. Ringo: An experimental wdm optical packet network for metro applications. *IEEE Journal on Selected Area in Communications*, 22:1561–1571, 2004.

[16] R. Chen et al. Msm-based integrated cmos wavelength-tunable optical receiver. *IEEE Photonics Technology Letters*, 17(6):1271–1273, 2005.

[17] C. Clos. A study of nonblocking switching networks. *Bell Syst. Tech. J.*, 32(2):406–424, 1953.

[18] Microsoft Corporation. Visual c++ developer center. `http://msdn2.microsoft.com/en-us/visualc/default.aspx`.

[19] N.E. Dahdah et al. All-optical wavelength conversion by eam with shifted bandpass filter for high bit-rate networks. *IEEE Photonics Technology Letters*, 18(1):61–63, 2006.

[20] K. Dolzer, C. Gauger, J. Spaeth, and S. Bodamer. Evaluation of reservation mechanisms for optical burst switching. *International Journal of Electronics and Communications*, 55(1):18–26, 2001.

[21] P. Ebrahimi, R. Chen, A.E. Willner, and D.A.B. Miller. Filtering and high-speed switching characteristics of a c-band tunable wavelength-selective msm detector. *Proc. of 31st European Conference on Optical Communication*, 3:499–500, 2005.

[22] J.M.H. Elmirghani and H.T. Mouftah. Technologies and architectures for scalable dynamic dense wdm networks. *IEEE Communications Magazine*, 38(2):58–66, 2000.

[23] F. Farahmand, Q. Zhang, and J.P. Jue. Dynamic traffic grooming in optical burst-switched networks. *IEEE/OSA Journal of Lightwave Technology*, 23:3167–3177, 2005.

[24] C.M. Gallepand and E. Conforti. Reduction of semiconductor optical amplifier switching times by preimpulse step-injected current technique. *IEEE Photonics Technology Letters*, 14(7):902–904, 2002.

[25] P. Gambini et al. Transparent optical packet switching: network architecture and demonstrators in the keops project. *IEEE Journal on Selected Areas in Communications*, 16(7):1245–1257, 1998.

[26] C.M. Gauger, M. Kohn, and J. Scharf. Comparison of contention resolution strategies in obs network scenarios. *Proc. of 6th International Conference on Transparent Optical Networks*, 1:18–21, 2004.

[27] D. Grieco, A. Pattavina, and Y. Ofek. Fractional lambda switching for flexible bandwidth provisioning in wdm networks: principles and performance. *Photonic Network Communications*, 9(3):281–296, 2005.

[28] N.F. Huang, G.H. Liaw, and C.P. Wang. A novel all-optical transport network with time-shared wavelength channels. *IEEE Journal on Selected Area in Communications*, 18(10):1863–1875, 2000.

[29] D.K. Hunter and D.G. Smith. New architectures for optical tdm switching. *IEEE/OSA Journal of Lightwave Technology*, 11(3):495–511, 1993.

[30] L. Huo et al. A study on the wavelength conversion and all-optical 3r regeneration using cross-absorption modulation in a bulk electroabsorption modulator. *IEEE/OSA Journal of Lightway Technology*, 24(8):3035–3044, 2006.

[31] Opal Kelly Incorporated. Xem3001 - xilinx spartan-3 integration module. `http://www.opalkelly.com/products/xem3001/`.

[32] I.P. Kaminow et al. A wideband all-optical wdm network. *IEEE Journal on Selected Area in Communications*, 14(5):780–799, 1996.

[33] F.P. Kelly. Blocking probabilities in large circuit-switched networks. *Advances in Applied Probability*, 18(2):473–505, 1986.

[34] S. Kim, B. Mukherjee, and M. Kang. Integrated congestion-control mechanism in optical burst switching networks. *IEEE Proc. of GLOBECOM*, 4, 2005.

[35] L. Kleinrock. *Queueing Systems, Volume 1, Theory.* John Wiley & Sons, 1975.

[36] D. Klonidis et al. Fast and widely tunable optical packet switching scheme based on tunable laser and dual-pump four-wave mixing. *IEEE Photonics Technology Letters*, 14(8):1412–1414, 2004.

[37] S. Kodama et al. 2.3 picoseconds optical gate monolithically integrating photodiode and electroabsorption modulator. *IEEE Electronics Letters*, 37(19):1185–1186, 2001.

[38] R. Laroy. *New concepts of wavelength tunable laser diodes for future telecom networks.* PhD thesis, Universiteit Gent, 2005-2006.

[39] S. Lee and L. Kim. Drop policy to enhance tcp performance in obs networks. *IEEE Communications Letters*, 10(4):299–231, 2006.

[40] J. Li, C. Qiao, J. Xu, and D. Xu. Maximizing throughput for optical burst switching networks. *IEEE Proc. of INFOCOM*, 3:1853–1863, 2004.

[41] Z.G. Lu, S.A. Boothroyd, and J. Chrostowski. Tunable wavelength conversion in a semiconductor-fiber ring laser. *IEEE Photonics Technology Letters*, 11(7):806–808, 1999.

[42] C.F.R. Mateus et al. Widely tunable torsional optical filter. *IEEE Photonics Technology Letters*, 14(6):819–821, 2002.

[43] J.D. Merlier et al. Wavelength channel accuracy of an external cavity wavelength tunable laser with intracavity wavelength reference etalon. *IEEE/OSA Journal of Lightway Technology*, 24(8):3202–3209, 2006.

[44] Inc. Mindspeed Technologies. 144x144 3.2 gbps crosspoint switch with programmable input equalization and output pre-emphasis. `http://www.mindspeed.com/web/home.html`.

[45] T. Miyazaki et al. Proteus-lite project: dedicated to developing a telecommunication-oriented fpga and its applications. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 8(4):401–414, 2000.

[46] A. Muller and D. Stoyan. *Comparison Methods for Stochastic Models and Risks.* Wiley & Sons, 2002. ISBN: 978-0-471-49446-1.

[47] Hung Q. Ngo. Wdm switching networks, rearrangeable and nonblocking [w,f]-connectors. *SIAM Journal on Computing*, 35(3):766–785, 2005.

[48] V.T. Nguyen, R. LoCigno, and Y. Ofek. Design and analysis of tunable laser-based fractional lambda switching (fls). *IEEE Proc. of INFOCOM*, 2006.

[49] Available Online. Intel optical transceivers. `http://www.intel.com/design/network/products/optical/lc_transceivers.htm`.

[50] R. Parthiban et al. Does optical burst switching have a role in the core network? *Technical Digest of OFC/NFOEC*, 3, 2005.

[51] C. Qiao and M. Yoo. Optical burst switching ('obs') a new paradigm for an optical internet. *Journal of High Speed Networks*, 8(1):69–84, 1999.

[52] J. Ramamirtham and J. Turner. Time sliced optical burst switching. *IEEE Proc. of INFOCOM*, pages 2030–2038, 2003.

[53] J. Ramamirtham, J. Turner, and J. Friedman. Design of wavelength converting switches for optical burst switching. *IEEE Journal on Selected Area in Communications*, 21(7):1122–1132, 2003.

[54] R. Ramaswami and K.N. Sivarajan. *Optical networks: a practical perspective*. Morgan Kaufmann Publishers, 2 edition, 2001. chapter 8, 9, 10, 11.

[55] M.C. Ring. My arbitrary precision math (mapm) library. `http://www.tc.umn.edu/~ringx004/mapm-main.html`. Available online.

[56] K. Ross et al. Scheduling bursts in time-domain wavelength interleaved networks. *IEEE Journal on Selected Areas of Communications*, 21(9):1441–1451, 2003.

[57] I. Rubin and J.H. Lee. Performance analysis of interconnected metropolitan area circuit-switched telecommunications networks. *IEEE Transactions on Communications*, 36(2):171–185, 1988.

[58] Temex SA. Time & frequency synchronisation gps clocks, epsilon board oem ii. `http://www.temex.com`.

[59] D. Sadot. High speed tunable fiber loop lasers for dense wdm systems. *Optical Engineering*, 37(6):1770–1774, 1998.

[60] D. Sadot and B. Gurion. Tunable optical filters for dense wdm networks. *IEEE Communications Magazine*, 16(12):50–55, 1998.

[61] R. Sato et al. Tuning technique to optimize input power of a cross-phase modulation wavelength converter. *IEEE/OSA Journal of Lightway Technology*, 22(8):1883–1892, 2004.

[62] Y. Shun, S.J.B. Yoo, B. Mukherjee, and S. Dixit. All-optical packet switching for metropolitan area networks: opportunities and challenges. *International Journal of Electronics and Communications*, 39(3):142–148, 2001.

[63] J.E. Simsarian and L. Zhang. Wavelength locking a fast-switching tunable laser. *IEEE Photonics Technology Letters*, 16(7):1745–1747, 2004.

[64] R. Srinivasan and A. K. Somani. A generalized framework for analyzing timespace switched optical networks. *IEEE Journal on Selected Area in Communications*, 20(1):202–215, 2002.

[65] S. Subramaniam, E.J. Harder, and H.A. Choi. Scheduling multirate sessions in time division multiplexed wavelength-routing networks. *IEEE Journal on Selected Area in Communications*, 18(10):2105–2110, 2000.

[66] D. Suvakovic and I. Hadzie. An fpga application with high speed serial transceiver running at sub nominal rate. *In Proc. of Int' Conf. on Field Programmable Logic and Applications*, pages 229–234, 2005.

[67] J. Teng and G.N. Rouskas. Wavelength selection in obs networks using traffic engineering and priority-based concepts. *IEEE Journal on Selected Areas in Communications*, 23(8):1658–1669, 2005.

[68] A. Tucker. *Applied Combinatorics.* John Wiley & Sons, 1980.

[69] J.S. Turner. Terabit burst switching. *Journal of High Speed Networks*, 8(1):3–16, 1996.

[70] W. Wang et al. Tunable photodetector based on gaas/inp wafer bonding. *IEEE Electron Device Letters*, 27(10):827–829, 2006.

[71] I. Widjaja, I. Saniee I, R. Giles, and D. Mitra. Light core and intelligent edge for a flexible, thin-layered, and cost-effective optical transport network. *IEEE Communications Magazine*, 41:S30–S36, 2003.

[72] I. Widjaja and I. Saniee. Simplified layering and flexible bandwidth with twin. *Proc. of Workshop on Future Directions in Network Architecture*, pages 13–20, 2004.

[73] D. Wolfson, T. Fjelde, and A. Kloch. Technologies for all-optical wavelength conversion in dwdm networks. *Proc. of 4th IEEE CLEO/Pacific Rim*, pages II574–II575, 2001. invited talk.

[74] Y. Xiong, M. Vandenhoute, and H.C. Cankaya. Control architecture in optical burst switched wdm networks. *IEEE Journal on Selected Areas of Communications*, 18(10):1838–1851, 2000.

[75] L. Xu, H.G. Perros, and G. Rouskas. Techniques for optical packet switching and optical burst switching. *IEEE Communications Magazine*, 39:136–142, 2001.

[76] F. Xue et al. Design and experimental demonstration of a variable-length optical packet routing system with unified contention resolution. *IEEE/OSA Journal of Lightway Technology*, 22(11):2570–2581, 2004.

[77] J.M. Yates, J.P.R. Lacey, and D. Everitt. Blocking in multiwavelength tdm networks. *Telecommunication Systems*, 12(1):1–19, 1999.

[78] C.H. Yeh, C.C. Lee, Y. Hsu, and S. Chi. Fast wavelength-tunable laser technique based on a fabryprot laser pair with optical interinjection. *IEEE Photonics Technology Letters*, 16(3):891–893, 2004.

[79] S.J.B. Yoo et al. Rapidly switching all-optical packet routing system with optical-label swapping incorporating tunable wavelength conversion and a uniform-loss cyclic frequency awgr. *IEEE Photonics Technology*, 14(8), 2002.

[80] T. Yoshimatsu, S. Kodama, K. Yoshino, and H. Ito. 100-gb/s error-free wavelength conversion with a monolithic optical gate integrating a photodiode and electroabsorption modulator. *IEEE Photonics Technology Letters*, 17(11):2367–2369, 2005.

[81] A.H. Zaim, H.G. Perros, and G.N. Rouskas. Computing call-blocking probabilities in leo satellite constellations. *IEEE Transactions on Vehicular Technology*, 52(3):622–636, 2003.

Appendix A

Proof of equation (4.5.10)

Following is the proof of eq. (4.5.10).

First, let us find the number of ways $C(m, n, p)$ to put m identical balls into n distinct boxes such that no box has more than p balls, $p > 1$. (While this may look like a classic combinatorial problem, we failed to find appropriate references also outside the field of blocking probabilities.)

The above problem is equivalent to finding the number of integer solutions to the equation:

$$e_1 + e_2 + e_3 + ... + e_n = m \qquad\qquad 0 \leq e_i \leq p$$

The generating function for the above equation is:

$$h(x) = (1 + x + x^2 + x^3 + x^4 ... + x^p)^n$$

The problem turns to find the coefficient of x^m in the polynomial $h(x)$.

From a well-known polynomial identity

$$\frac{1 - x^{p+1}}{1 - x} = 1 + x + x^2 + x^3 + ... + x^p$$

we can represent

$$h(x) = f(x)g(x)$$

where

$$f(x) = \frac{1}{(1 - x)^n}$$

and

$$g(x) = (1 - x^{p+1})^n$$

Following are polynomial expansions in chapter 3 of [68]:

$$f(x) = \frac{1}{(1-x)^n} = \sum_{i=0}^{\infty} \alpha_i$$

where

$$\alpha_i = \binom{i+n-1}{i} \tag{A.0.1}$$

and

$$g(x) = (1 - x^{p+1})^n = \sum_{i=0}^{n} \beta_{i(p+1)} x^{i(p+1)}$$

where

$$\beta_{i(p+1)} = \begin{cases} (-1)^i \binom{n}{i} & \text{if } i = 0, 1, .., n \\ 0 & \text{otherwise} \end{cases} \tag{A.0.2}$$

Therefore, $h(x)$ can be rewritten as

$$h(x) = \sum_{q=0}^{\infty} \Lambda_q x^q$$

where the coefficients Λ_q are given by:

$$\Lambda_q = \sum_{i=0}^{q} \alpha_{q-i} \beta_i \tag{A.0.3}$$

We aim at finding the coefficient of x^m in $h(x)$, thus we only need to consider the terms $\alpha_{m-i}\beta_i$ in which the β_i's, coefficients of $g(x)$, are nonzero. Substitute $q = m$ into (A.0.3), we have:

$$\begin{aligned} C(m,n,p) = \;& \Lambda_m = \textstyle\sum_{i=0}^{n} \binom{m-i(p+1)+n-1}{m-i(p+1)} (-1)^i \binom{n}{i} \\ = \;& \textstyle\sum_{i=0}^{n} (-1)^i \binom{n}{i} \binom{m-i(p+1)+n-1}{n-1} \end{aligned} \tag{A.0.4}$$

In our problem, we need to find the number of dispositions of the b_v symbols '0' into the v distinct runs such that each run has at least one symbol and no run has more than $(z-1)$ symbols. We first put in each run one symbol, then distribute $b_v - v$ remaining symbols so that no more than $z-2$ symbols will be distributed into each run. Thus, substituting $m = b_v - v$, $n = v$, and $p = z-2$ into (A.0.4), we obtain:

$$C_{b_v} = \sum_{i=0}^{v} (-1)^i \binom{v}{i} \binom{b_v - i(z-1) - 1}{v-1}$$

for $v > 0$.

Besides, notice that:

- if $v = 0$, we set $C_{b_v} = 1$ since C_{b_v} is a factor of a product.
- if $b_v = v$ then obviously $C_{b_v} = 1$.

Thus, we obtain the number C_{b_v} as in 4.5.10.

Appendix B

Proof of multiple counting while deriving (4.5.6)

Following is the proof of multiple counting while deriving eq. (4.5.6).

Let Σ denote the set of all combinations generated in the product $C_{uv}C_aC_{b_u}C_{b_v}$.

Consider one pattern χ_1 drawn at random from Σ, and label all runs of χ_1 in an increasing order within their proper set as following:

$$\chi_1 = u_1a_1v_1a_2\cdots u_i\cdots a_j\cdots v_k\cdots u_u\cdots v_va_{u+v}$$

in which $u_i \in \mathbb{U}$, $v_k \in \mathbb{V}$, and $a_j \in \mathbb{A}$; $i = 1, 2, \cdot\cdot, \ u$; $k = 1, 2, \cdot\cdot, \ v$; and $j = 1, 2, \cdot\cdot, \ u + v$.

By construction, also all the following patterns are items of the set Σ and they are distinct:

$$\begin{aligned} \chi_2 &= v_1a_2\cdots u_i\cdots a_j\cdots v_k\cdots v_va_{u+v}u_1a_1 \\ &\cdots \\ \chi_{u+v} &= v_va_{u+v}u_1a_1v_1a_2\cdots u_i\cdots a_j\cdots v_k\cdots \end{aligned}$$

It is clear that $\chi_2, \cdots, \chi_{u+v}$ are also obtained as s-position shifts of χ_1, with proper $s < K$.

More formally, letting $\overleftarrow{\chi_i}^s$ denote a left shifting on χ_i with shifting length s, and $\mid *_i \mid$ length of a generic run $*_i$, we have:

$$\begin{aligned}
\overleftarrow{\chi_2}^{|u_1|+|a_1|} &\equiv \chi_1 \\
\dots &\equiv \chi_1 \\
\overleftarrow{\chi_{u+v}}^{|u_1|+|a_1|+|v_1|+|a_2|+\cdot+|u_i|+\cdot+|a_j|+\cdot+|v_k|+\cdot} &\equiv \chi_1
\end{aligned}$$

Since χ_1 was selected randomly, we conclude that any χ_i appears exactly $(u+v)$ times in the total number $KC_{uv}C_aC_{b_u}C_{b_v}$.

Appendix C

Acronyms

AOTF acoustooptical tunable filter

AWG arrayed waveguide grating

CDR integrated clock data recovery

CMOS complementary metal oxide semiconductor

CTR common time reference

DBR distributed Bragg reflector

DeMUX de-multiplexing

DFB distributed feedback

DSDBR digital supermode distributed Bragg reflector

DTOF digitally tunable optical filter

EAM electroabsorption modulator

ECL external-cavity laser

EEPROM electrically erasable programmable read only memo

FBG fiber Bragg grating

FDL fiber delay line

FLP fractional lambda pipe

FLS fractional lambda switching

FP Fabry-Perot

FPGA field programmable gate array

FPGAs field programmable gate arrays

FWM four-wave mixing

GPS global positioning system

IF immediate forwarding

ISI inter symbol interference

ITU international telecommunication union

MEMS micro-electromechanical structure

MGY modulated grating Y-structure

MUX multiplexing

MZ Mach-Zehnder

NIF non-immediate forwarding

OADM optical add-drop multiplexer

OBS optical burst switching

OPS optical packet switching

ORAM optical random access memory

PCB printed circuit board

PIC photonic integrated circuit

PLL phase-locked loop

QoS quality of service

RAM random access memory

RIN relative intensity noise

SFR semiconductor-fiber ring

SGDBR sampled grating DBR SOA Semiconductor optical amplifier

SMSR side-mode suppression ratio

SOA semiconductor optical amplifier

TDM time division multiplexing

TDS time-driven switching

TEC thermoelectrical cooler

TFF thin film filter

TSI time slot interchange

TTF torsional tunable filter

TWIN time-domain wavelength interleaved network

USB universal serial bus

UTC Universal Time Coordinated

VHDL VHSIC hardware description language

VOA variable optical attenuation

WR static wavelength router

XAM cross-absorption modulation

XGM cross-gain modulation

XPM cross-phase modulation