HOW FAR CAN WE GO WITH OBS NETWORKS

Mohammad Nurujjaman

A thesis

in

The Department

of

Computer Science & Software Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Computer Science

Concordia University

Montréal, Québec, Canada

January 2010

Your file Votre référence
ISBN: 978-0-494-67131-3
Our file Notre référence
ISBN: 978-0-494-67131-3

Abstract

How far can we go with OBS networks

Mohammad Nurujjaman

Optical Burst Switching (OBS) was proposed ten years ago as an alternative switching paradigm in order to overcome some of the drawbacks of Optical Circuit Switching (OCS). While OBS is no more necessarily perceived as a competitor to OCS, but more of a more adapted switching for networks with bursty and highly dynamic traffic, there is still a debate around OBS, i.e., how far an OBS network can go in terms of throughput with no or limited burst losses. This thesis attempts to answer this question by investigating how to devise an upper bound on the throughput of an OBS network, assuming no recourse to electrical buffering is made at any intermediate node. We investigate both the burst scheduling and routing issues, with a larger focus on routing in three directions: (i) exploration of weighted k-shortest paths, (ii) revisiting load balancing, (iii) examining tree decomposition. Simulations have been conducted to compare and evaluate each of the new ideas with adapted (with respect to throughput upper bounding) previously proposed routing algorithms on different network and traffic instances. A comparison of the best upper bound with lower bounds obtained under various assumptions is presented.

Acknowledgments

I am heartily thankful to my supervisor, Dr. Brigitte Jaumard, Concordia University Research Chair (Tier 1) on the Optimization of Communication Networks. This thesis would not have been possible without her encouragement and personal guidance. Her vast knowledge in optimization techniques, capability of explaining things very easily and supporting attitude from the initial to the final level enabled me to understand the subject of the thesis.

I would like to show my gratitude to Dr. Gérard Hébuterne, Telecom & Management-SudParis, France, who often guided me with his innovative ideas. I also would like to thank my colleague Thomas Coutelen, Concordia University, who has made available his support in many ways during my research.

My special gratitude to my family and friends whose continuous encouragement and mental support made this thesis possible.

Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of the thesis.

Contents

List of Figures

List of Tables

Acronyms

AARA Adaptive Alternate Routing Algorithm

ADM Add-Drop Multiplexer

ASE Amplifier Spontaneous Emission

ATh-BA Adaptive Threshold Based Burst Assembly

ATM Asynchronous Transfer Mode

CLDR Contention-based Limited Deflection Routing

E-JIT Enhanced Just in Time

FDL Fiber Delay Line

FF First-Fit

GRWA Grooming, Routing and Wavelength Assignment

IEEE Institute of Electrical and Electronics Engineers

ILP Integer Linear programming

JET Just Enough Time

JIT Just-in-Time

LAUC-VF Latest Available Unscheduled Channel With Void Filling

LLRBA Link Loss Rate-based Burst Assembly

MILP Mixed Integer Linear programming

OBS Optical Burst Switching

OCS Optical Circuit Switching

OEO Optical-Electrical-Optical

OPS Optical Packet Switching

PLBRA Path Loss Rate-based Burst Assembly

RWA Routing and Wavelength Assignment

VFO Virtual Fixed Offset

Chapter 1

Introduction

1.1 Motivation

Demand of bandwidth in communication networks is gradually increasing and expected to continue to increase during the next decades. Bandwidth usage of the Internet is doubling in every year [35]. Due to this continuous growth in Internet, Dense Wavelength Division Multiplexing (DWDM) technique has been highly focused on realizing the next-generation IP backbone networks. It is expected to fulfill the high bandwidth requirements of Internet and appears to be a cost effective approach to cope with the ever growing Internet traffic.

To overcome the limitations of switching processing times of electronic routers, the idea of all-optical networks came up. Three switching technologies have been proposed and thoroughly investigated for IP traffic over WDM networks: Optical Circuit Switching (OCS), Optical Packet Switching (OPS) and Optical Burst Switching (OBS). The last two, OPS and OBS, are still in a prototype stage and only OCS has been implemented in real life networks till today. Optical Circuit Switching (OCS) is based on wavelength routing where a lightpath needs to be established using a specific wavelength in every link for each connection [5]. Once a connection is setup, the data remains in the optical domain throughout the path if it is a single hop connection and ensures guaranteed and reliable transmission. However, in current networks, not all requests are single-hop routed, though most often the connections are limited to two or three hops on average. Due to the required setup time and the probability of inefficient usage of bandwidth in case of bursty traffic, OCS is well adapted to stable and large amount of traffic. The second technology is Optical Packet Switching (OPS), which was designed to provide an efficient use of bandwidth and overcome the limitations of OCS such as bandwidth utilization. In an OPS network, the packet header needs to be processed either all-optically or electronically after an optical-electronic (O/E) conversion [5]. During the header processing time, the data payload must be stored in Fiber Delay Lines (FDLs). The requirements of FDLs appear as a shortcoming to the OPS approach and limits the use of OPS since the FDL technology is not yet matured and very expensive. Synchronization is another issue that needs to be managed in an OPS network [19]. Optical Burst Switching (OBS), another switching technique, appeared as a novel approach for the next-generation all-optical Internet.

Optical Burst Switching was designed to combine the advantages of OCS and OPS while overcoming their limitations. In an OBS network, several data packets are aggregated to form large bursts at ingress nodes. Data payload and its control information are separated. A control packet contains the header information and is transmitted on a dedicated control channel well ahead of the time the burst is sent. This time interval between data burst and control packet is called Offset Time. It allows the core routers enough time to process the control packet and reserve the resources for the upcoming data burst while eliminating the necessity of FDLs at each intermediate switch. OBS was designed to use a one way reservation protocol, i.e., data bursts are sent without waiting for any confirmation of resource reservation. It leads to a critical issue of OBS networks called contention. When two or more data bursts arrive at a same output port at the same time, contention occurs and only one burst is allowed to transmit while the other contending bursts are dropped. Contention is considered as a vital shortcoming of OBS and causes significant performance degradation. This is why it has received a considerable amount of attention in literature. Several classical techniques have been proposed for contention resolution in OBS networks such as deflection routing, wavelength conversion, burst segmentation, etc. Another proactive approach for contention resolution is to avoid congestion in the network by well distributing the traffic loads across the network. Besides contention, some other critical issues such as burst assembly, burst switching, QoS support need to be resolved before implementing OBS in real life networks.

1.2 Thesis Contribution

The objective of this thesis was to find out an answer to the question: Whether to further develop OBS. To answer this question, we have investigated several directions that have the potential to maximize the throughput of OBS networks with some valid relaxations. We have designed two algorithms: SSP and WSP, to select the best possible path among k-shortest paths for a given source-destination pair under specified criteria. SSP sequentially examines all available paths to select the path that provides the smallest destination time for a given burst while WSP introduces a weight to each path under various considerations and selects the one with minimum possible weight. A heuristic was designed to balance the loads over the network to avoid bottleneck scenarios in the network. Another two algorithms: ST and CT were designed to organize the set of routes over a set of trees such that each tree collects all the routes for all possible sources toward a given destination. ST builds a tree by taking into account the offered load and the number of incoming flows on a given link. CT, in addition, takes into account the existence of other trees in the network as well as the offered load and the number of incoming flows. All the algorithms have been implemented using object oriented technique in C++. Simulations have been performed to examine the throughputs provided by the algorithms under three traffic patterns and three network topologies. A comparative analysis of the performances of our newly designed algorithms with some of the most efficient ones from the literature is presented. We also present a comparison of the best upper bound with lower bounds obtained under various assumptions.

1.3 Thesis Organization

The thesis is organized as follows. An overview of evolution of all optical networks is given in Chapter 2, with a brief presentation of classical concepts of the Optical Burst Switching (OBS) technology. In Chapter 3, we review various resolutions that have already been proposed in the literature to avoid the limitations of OBS and a brief comparison of performances of those resolution techniques. Chapter 4 describes the algorithms (SSP & WSP) that selects one path among the set of k-shortest paths for each source-destination pair. Chapter 5 presents the algorithm designed to balance the loads over the network. The key concepts of two tree decomposing algorithms (ST & CT) are described in Chapter 6. We present the experimental results of our newly designed algorithms and an analysis of the comparative results in Chapter 7. Chapter 7 also includes a comparison of the best upper bound with the lower bounds obtained under various assumptions. Conclusions of the thesis and future work are discussed in Chapter 8.

Chapter 2

Optical Burst Switching (OBS)

In this chapter, we present the basic concepts of an OBS network. Section 2.1 provides the definitions of two alternate major switching technologies: Optical Circuit Switching (OCS) and Optical Packet Switching (OPS). The basic characteristics of OBS are developed in Section 2.2. Section 2.3 compares the three main switching technologies: OCS, OPS and OBS. The major challenges of OBS are discussed in Section 2.4. Some classical and innovative techniques to resolve OBS challenges are briefly described in Sections 2.5 and 2.6. Section 2.7 presents the major focus of the thesis.

2.1 Optical Switching

There are three available switching techniques in optical WDM networks: Optical Circuit Switching (OCS), Optical Packet Switching (OPS) and Optical Burst Switching (OBS).

Optical Circuit Switching (OCS) relies on connection-oriented data transmission. Switching is performed at the granularity level of an optical circuit or lightpath. Lightpaths need to be established before any data transmission. OCS provides guaranteed transmission, but may cause inefficient bandwidth utilization for bursty data traffic.

In OPS, switching is performed at the granularity level of packets. OPS is designed to overcome the limitations of OCS with data traffic. Since OCS is not optimized for bursty data traffic that is dominating in access networks, a considerable amount of attention has been paid to OPS by research community. OPS transmits data in the form of packets and does not need a dedicated connection. It requires packet header recognition, header processing and optical buffers at the intermediate nodes. However, the ideal form of OPS is beyond reach for the time being because of the unavailability of the required particular technologies by OPS.

2.2 Definition of an OBS Network

Optical Burst Switching (OBS), a new switching scheme, has been proposed to provide packet-like bandwidth utilization services while overcoming the limitations of both Optical Circuit Switching and Optical Packet Switching. It has been first introduced by Qiao and Yoo in [38]. OBS is a hybrid switching technique that combines the advantageous concepts of Optical Circuit Switching and Optical Packet Switching.

In an OBS network, a data burst is composed of several IP packets and formed by aggregating the IP packets at the ingress node. The bursts are transmitted through the network all optically until they reach their egress node where they will be disassembled. For each data burst, a control packet containing the header information including the burst length is transmitted to configure the switches along the path of the burst. Since the control packet is separated from burst data and significantly smaller in size, it is transmitted on a dedicated control channel. The control packet goes through optical-electrical-optical (O/E/O) conversion in each OBS intermediate node and processed electronically to configure the switches. To allow sufficient time for the control packet to be processed, the data burst is transmitted after a certain amount of time the control packet is sent. This time is called Offset Time. It is introduced to transmit the burst all optically and avoids the necessity of Fiber Delay Lines (FDLs) at the intermediate nodes in an OBS network. Figure 1 shows a burst and its control packet transmission in an OBS network with offset time.

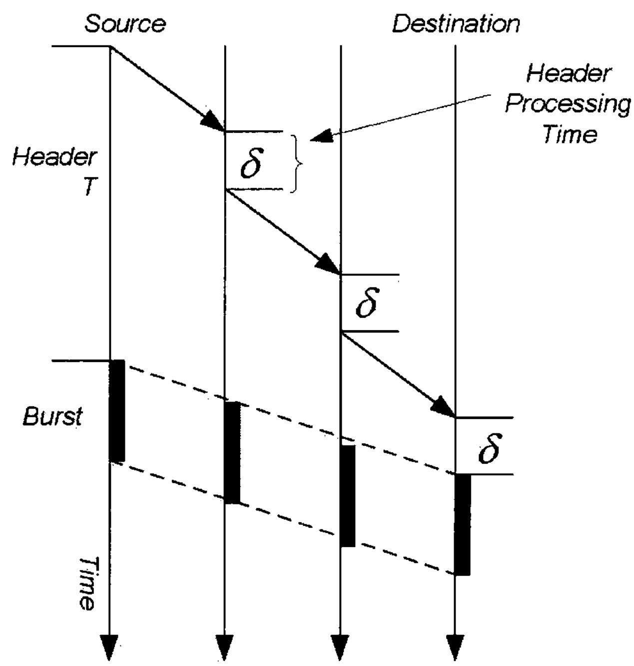

Figure 1: Offset time [taken from [20]]

Resource reservations like switch setup at the intermediate nodes in an OBS network for transmission of a burst is performed by its control packet. Since a control packet carries the length information of the burst, resources can be reserved for a specified period of time rather than indefinite time. Thus OBS has the opportunity to allocate resources in a more efficient manner than OCS. Separation of data from its header is another advantage of OBS.

It overcomes the deficiency of OPS where each packet needs either an optical storage or an O/E/O conversion (time costly) at each intermediate node. By introducing an offset time between the transmission of the control packet and the data burst, OBS also overcomes the necessity of FDLs at each intermediate node, which is another limitation of OPS. So OBS potentially avoids the shortcomings of optical circuit switching and optical packet switching while having their advantages.

An OBS network consists of optical burst switching nodes that are interconnected via fiber links. Each fiber link is capable to support multiple wavelength channels using wavelength division multiplexing. Nodes are categorized as either edge nodes or core nodes in an OBS network as shown in Figure 2. Assembling packets into bursts and scheduling bursts for transmission on an outgoing wavelength are the primary responsibilities of edge nodes. The core nodes are responsible for switching bursts from input ports to output ports based on the burst header packets, and for handling burst contentions.

Edge nodes can be either ingress or egress nodes. An ingress edge node is responsible for burst assembly, routing, wavelength assignment and scheduling of bursts at the edge. The assembled bursts are transmitted all optically over OBS core routers without any storage at intermediate nodes within the core. Upon receiving a burst, the egress edge node disassembles the burst into packets and forwards the packets to the higher network layer.

2.3 Comparison

Figure 3 summarizes the differences between the OCS, OPS and OBS switching paradigms. It shows that, OPS and OBS are designed to improved bandwidth utilization while decreasing the setup latencies over OCS. However, OCS does not require fast switches like OPS

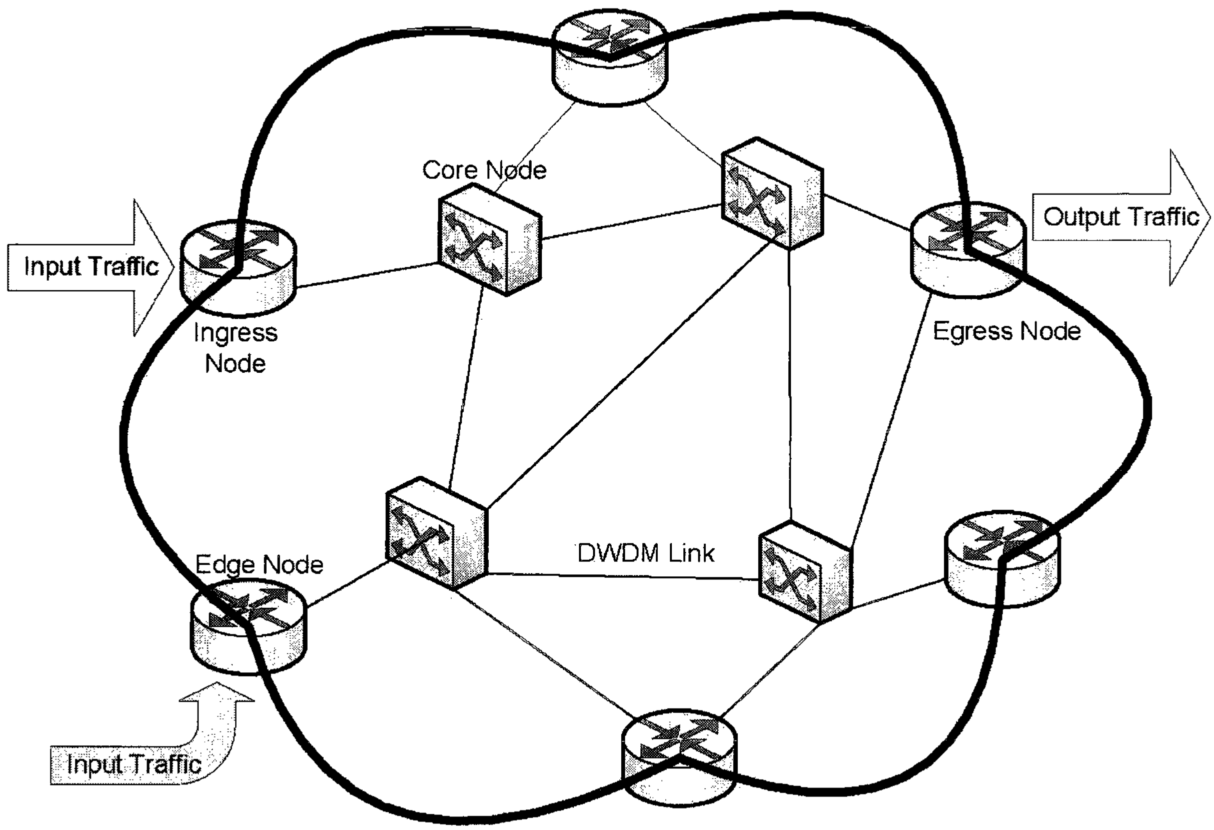

Figure 2: OBS network architecture [taken from [20]]

and OBS and also does not suffer from synchronization issues.

2.4 OBS Challenges

The idea of OBS combines the major advantages of OCS and OPS while overcoming their deficiencies. However, OBS has still some open issues that need to be resolved before a real life implementation.

Optical Switching Paradigm	Bandwidth Utilization	Setup Latency	Switching SpeedReq.	Proc. / Sync. Overhead	Traffic Adaptively
Optical Circuit Switching	Low	High	Slow	Low	Low
Optical Packet Switching	High	Low	Fast	High	High
Optical Burst Switching	High	Low	Medium	Low	High

Figure 3: Comparison of switching technologies [taken from [20]]

Contention: Since an OBS network provides a connectionless transmission and uses a one way reservation protocol, bursts may contend at the intermediate nodes of the core network without mechanisms to resend bursts in case of burst losses. Contention will occur at a node if two or more bursts from different sources intend to use the same outgoing link at the same time. It degrades the overall performance of an OBS network to a great extent. While contention is also a critical issue in OPS networks, it is resolved by using FDLs. But FDLs are very expensive up to now and not a very practical solution to implement.

Length of Offset time: The length of the offset time may also cause a significant impact on the performance of an OBS network. An offset time that is too small may cause the burst arrival at an intermediate node before it is configured by the control packet and results in the loss of the burst. Whereas an offset time that is too long may lead to insufficient bandwidth reservation. Also, there is a confusion with the benefits of fixed and variable length offset time. Fixed length offset time provides a better QoS performance for the traffic with higher-priority classes. However, some studies show that fixed offset time leads to increased burst contention possibilities [48]. The authors of [3] proposed a variable length offset time scheme by using a statistical traffic shaping model. It keeps reducing the burst dropping probability.

Burst assembly: Burst assembly is another key area in OBS networks that needs to be further investigated. Various studies have shown that the assembly strategies in the egress nodes have a significant impact on traffic characteristics [57] in an OBS network. In addition, traffic characteristics have a large impact on network performances in terms of blocking probability and throughput [41]. Burst assembly algorithms are mainly classified as timer and length driven. In a timer based scheme, packets arrived within a fixed amount of time interval, are aggregated to form a burst while the length based scheme uses a threshold with a maximum size to form a burst.

2.5 Classical Techniques to Resolve Contention

OBS networks use a one-way reservation protocol and the ingress nodes send out the bursts into the network without any confirmation of resource reservation. Thus, contention may occur at any intermediate node and can be resolved in three different manners: Deflection, dropping and preemption. Deflection is a method where the burst is sent to a different output port rather than the one it destined to. If a burst cannot be deflected, a common approach is to drop the burst. Another approach is to preempt an existing burst by the contending burst with higher priority. Performance and traffic statistics need to be collected to perform preemption. In this section, we briefly describe the various domains of deflection.

Deflection can be performed in three different domains: wavelength, space and time domain. Wavelength conversion technique is used in wavelength domain, deflection routing in space and fiber delay line in time domain.

2.5.1 Wavelength Domain: Wavelength Conversion

Wavelength conversion is a process where the wavelength of an input signal is to be converted to another wavelength on the outgoing channel. Now a days in a WDM network, several wavelengths run on a fiber link and, in near future, it is expected to be as many as 160 - 320 wavelengths per fiber [20]. Wavelength conversion technique can be used to take the advantage of having multiple wavelengths in a fiber. Let us assume that two bursts from different sources are destined to use the same output port at the same time. In a usual situation, i.e., without wavelength conversion, one of those two bursts need to be dropped. By the use of wavelength conversion, both bursts can be transmitted. Wavelength of one of the contending bursts can be converted to any other available output wavelength, thus eliminating collision.

Some studies show that, in an OBS network, wavelength conversion can reduce contention by utilizing additional capacity in the form of multiple wavelengths [45, 18]. But the technology of optical wavelength conversion is not matured yet and still in the laboratory. The devices are very expensive and also the range of conversion is limited [20]. Several categories of wavelength conversion exist: Full conversion, limited conversion and fixed conversion.

Full conversion allows any incoming wavelength to be converted to any outgoing wavelength and overcomes the wavelength continuity constraint in an OBS network. Limited conversion restricts the number of conversions at any intermediate node. It only allows a signal to be converted from one wavelength to a limited subset of other wavelengths. Fixed conversion is a restricted form of limited conversion. In fixed conversion, each incoming channel is connected to one predetermined outgoing channel and incoming wavelengths are restricted to be converted to only the predetermined outgoing channel.

2.5.2 Space Domain: Deflection Routing

In deflection routing, a contending burst is to be routed to an output port other than the port it was intended to resolve contention. Unlikely the store-and-forwarding routing which is widely used in classical routers with plenty of electrical memory, deflection routing forwards a contending burst immediately through another port without being stored. Deflection routing is usually triggered by an OBS node considering the status information of its own locally available resources. Such a decision does not include the overall status of the whole network and may end up with a sub optimal network performance. Typically, a burst travels longer routes in deflection routing compared to usual routes, which results in increased delay and degradation of signal quality [20]. Deflection routing may also be combined with other contention resolution schemes to improve network performance.

Deflection routing has pros and cons as well. On the one hand, it may suffer from potential looping and out-of-sequence delivery of packets, on the other hand, it may benefit by decreasing burst drop rate. In [45], deflection routing has been studied on a TCP network. It shows that deflection routing has a negative impact on TCP performance but has a benefit over burst dropping. Indeed, performance degradation due to deflection routing is smaller than the degradation due to burst losses. Deflection routing is alternatively called hot-potato routing when compared with the store-and-forward routing in [20]. Alternative port selection in deflection routing need to be dealt with different manners if the nodal degree is greater than two.

As the concept of deflection routing for OBS is still under experiments, several issues need to be resolved to make it practical. A major issue is to maintain a proper offset time between header and the data of a deflected burst. Bursts that are deflected to avoid contention may suffer from an insufficient offset time because a deflected burst may travel a longer path than its regular travel to destination. In such a scenario, there may be a point at which the initial offset time may appear as insufficient for the header to be processed and the switch to be configured before the data arrives to the switch. To resolve the issue, either the deflected burst needs to be dropped or some mechanism need to be implemented to limit the maximum number of hops that a deflected burst may travel. Also, additional mechanisms need to be implemented to overcome potential looping.

2.5.3 Time Domain: Fiber Delay Line (FDL)

The most conventional way of contention resolution in electronic packet switching networks is to store the contending packets in nodal electrical buffers and transmit them later. Packets are stored in the random access memory buffers at the intermediate switches. Unfortunately, optical memories are not yet available. However, optical signals can be delayed for a fixed amount of time by using fiber delay lines (FDLs). Buffering bursts is also possible in electronic domain but, in that prospect, bursts need to be converted from optical to electronic domain. This approach results in the loss of network transparency and the network no longer remains all-optical. It also increases the cost of network because all of the intermediate switches must have O/E/O conversion capabilities and it also requires electronic memories that are compatible with the speed of the optical domain.

2.6 Some Innovative Techniques to Resolve Contention

Now a days several innovative approaches have been investigated to resolve contention, in other words, to avoid contention to be occurred in the network pro-actively.

2.6.1 Load Balancing

Load balancing is a promising approach to avoid congestion in a network. In this technique, load is distributed over the network as evenly as possible to avoid a bottleneck situation that causes contention in the network. Load balancing is performed by selecting a less loaded path for a given source/destination pair rather than the shortest path. However, this approach may increase the average end-to-end delay in the network due to the selection of longer paths. Several load balancing approaches, investigated by research community, are explained in Section 3.3.

2.6.2 Burst Segmentation

Burst segmentation is another new approach to resolve contention. Burst segmentation was first introduced in [49] as a contention resolution technique. In burst segmentation, only the segment of a contending burst that overlaps with another burst is dropped rather than the whole burst. Studies related to burst segmentations and performance comparisons are explained in Section 3.3.

2.7 Thesis Project

This section demonstrates the goal of the thesis. Section 2.7.1 explains the objective of the research. We divided the research problem into two subproblems: Routing and Scheduling.

Sections 2.7.2 describes the routing subproblem and 2.7.3 explains the scheduling subproblem. Note that a lot more emphasis has been given to routing throughout the thesis.

2.7.1 Throughput Upper Bound

Besides the challenges of OBS networks discussed in Section 2.4, another major unanswered question is whether to develop OBS further. Since OBS is still a prototype, it is very important to get an idea of the maximum achievable throughput of an OBS network in order to evaluate the full OBS potential. There has been some attempts in the literature to compare the performance of OBS and OCS for given traffic classes. However, there was no clear domination of one over another. Some studies claim that OCS performs better whereas others claim in favor of OBS. We believe, both OBS and OCS may coexist in future optical networks for different classes of traffic. Instead of comparing two technologies, in this thesis, we stress out our investigation to find a tight upper bound on the throughput of OBS networks. Throughout the investigations, we consider a set of assumptions to relax some constraints of classical OBS networks:

All-optical. We assume that the network will be all-optical and no electrical memory is available in the core network since it was the basic motivation of optical burst switching. All-optical networks should a priori keep the transparency of the network.

Overall information available. In classical OBS, the bursts are launched immediately after the offset time without any confirmation of reservation and without any information on potential contention in core networks due to resource unavailability. Contention would not occur at such a high rate if the resources could be reserved from source to destination before the burst leaves the edge node. This is why, we assume that the overall information of available time-slots of all links are available in edge nodes at the time of scheduling.

No offset. Since the offset time itself may cause some loss of bursts in the network, in this thesis, we did not consider any offset time for the bursts. The rational of this assumption is quite similar to the previous assumption. Also, the offset time is not necessary anymore once all the information are available at the edge nodes and reservations are made before the bursts are launched.

Fixed size burst. The length of the bursts is assumed to be fixed in this thesis. Indeed, fixed size has been highly recommended for the burst length in the literature. Several studies showed that the network performance increases with fixed length bursts compared to variable length bursts.

Delay at the source. We allow the bursts to be delayed at the sources up to a predetermined maximum amount of time, if the desired resource is unavailable since the electrical memory is available at the edge nodes. Delaying the bursts at the edge nodes also helps us to avoid some contentions that might occur if the bursts are launched as soon as they are ready.

Under the above assumptions we believe that we achieve an empirical upper bound on throughput of OBS networks taking into account the classical all-optical techniques used so far in OBS networks. In particular, we exclude the recent proposals with, e.g., OBS translucent architectures [9].

2.7.2 Routing

The Routing subproblem takes care of the route that a burst should follow to reach destination. It should take into account the potential causes of contention and provides an efficient set of routes for all source-destination pairs in the network so that the burst loss is minimum. The techniques for contention resolution discussed in Section 2.5 are reactive and attempt to resolve the contention of bursts rather than to avoid them. A proactive approach attempts to prevent contention from occurring in the network rather than resolving the contention if it occurs. Contention could be reduced by minimizing the congestion. Shortest path routing is widely used in Optical Circuit Switching (OCS) and Optical Packet Switching (OPS) networks. The major shortcoming of shortest path routing is that it does not consider the traffic load offered to the network and results in congestion on some links while some other links in the network might remain underutilized. Such situations cause quite significant burst losses in an OBS network. The main objective of the routing subproblem is to determine a set of routes that balance the traffic across the network to reduce congestion and improve overall performance. In this thesis, we have designed some heuristics to find a best possible set of routes in the network.

2.7.3 Scheduling

Scheduling has a large impact on the performance of OBS networks. Since, except if we use FDLs, there are no optical buffers in the core nodes, scheduling only occurs at the edge nodes when the bursts are still in the electrical domain. Therefore at the edge nodes, the scheduler decides on the burst order and on the output channel the burst should be transmitted through. Scheduling bursts in an OBS network appears challenging due to two components: Burst arrival, heterogeneity of offset time and contention due to the lack of optical buffering. However, in the context of this thesis, we only look at the burst scheduling in an ideal framework (with respect to throughput upper bounding) when there is no offset time. Finding an optimal scheduling, i.e., which maximizes the throughput, is not an easy task even for a given set of routes and even without offset times.

Optimal burst scheduling on a tree network

We propose to explore a first idea where we restrict the network to be a tree network. The idea is then that, in such a case, we can reduce the scheduling on a star network without loss of generality. Then, as a star network is a simpler structure than a tree one, it is easier to solve the burst scheduling problem. Let us next go through an example to illustrate the idea.

An example

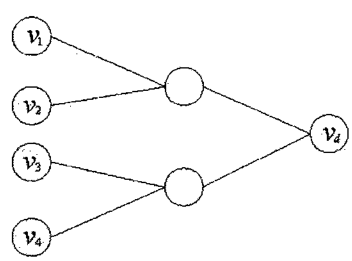

Figure 4: Tree with four nodes

Consider a tree with four leaf nodes v_1, v_2, v_3 and v_4 which are generating traffic destined to root node v_d. Figures 4 and 5 show the tree and its corresponding star network, respectively. Figure 6 shows the traffic instance that we will consider in this example. Bursts

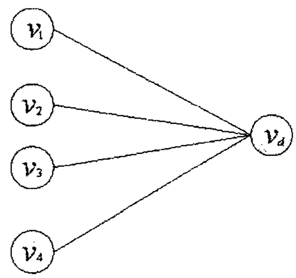

Figure 5: Star with four nodes

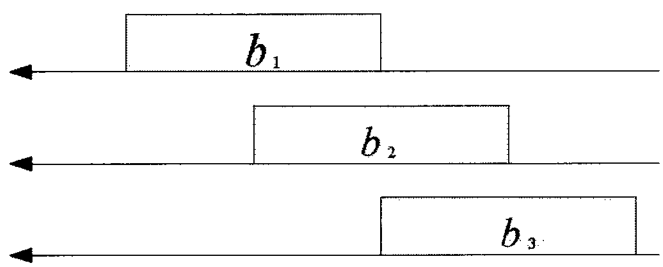

Figure 6: Traffic instance

b_1, b_2 and b_3 are generated by source nodes v_1, v_2 and v_3 respectively and successively. We assume these 3 bursts are ready to be launched in the order as shown in Figure 6. Let us assume that the propagation delays are identical for all links.

Now let us schedule the bursts in the tree network assuming we launch the bursts as soon as they are ready (i.e., first ready first scheduled). From Figure 6, it is clear that bursts b_1 and b_2 are overlapping each other and a contention follows. Since b_2 arrives before b_1, b_1 will be dropped, and b_2 survives. At destination node v_d, b_2 and b_3 contend again. This time, b_3 arrives before b_2. So b_3 survives and b_2 is dropped. In this overall scenario, two bursts are dropped and only b_3 is transmitted successfully.

On the other hand, consider the star architecture and let us schedule the bursts in order to maximize the throughput. It is easy to find that maximum throughput is obtained with dropping b_2, while b_1 and b_3 can safely reach their destination. Therefore in star network, only one burst needs to be dropped with this particular traffic instance whereas only one burst survived in the tree network.

Optimal burst scheduling on a star network

Let us first explain how we can reduce the scheduling on a star network, starting from a tree network, without loss of generality. In the optimal scheduling, we assume that the route is given for any pair of source and destination and for every burst (we do not necessarily use the same route for all bursts with the same source and destination nodes). Let us consider a burst b which is ready to be launched at time t_b^{READY}. Also assume that the travel time from source to destination of the given route is t for the burst b. The travel time is a fixed time but differs from one burst to the next and includes both the propagation time and the switch traversal time. To perform the conversion from tree network to star network, we need to adjust the ready time t_b^{READY} of burst b by taking into account the travel time t of the given route as $t_b^{\text{READY}} + t$.

We next propose a mathematical model to compute the optimal schedule on a star network. The underlying idea is as follows. We built a conflict graph where each node of the graph is associated with a burst. Two nodes are linked if their corresponding bursts b and b' overlap, taking into account their potential delay at their respective source nodes (i.e., δ_b and $\delta_{b'}$), see Figure 7 for an illustration. Then, in order to maximize the throughput, the objective is to find the independent set (subset of nodes that are not pairwise linked in

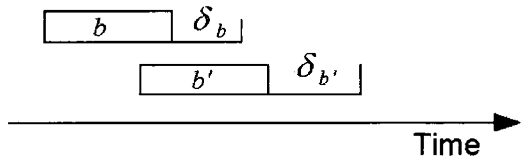

Figure 7: Optimal scheduling

any way) in that graph such that the sum of the lengths of the bursts associated with the nodes of that set is maximum. The following mathematical model translates the search of such an independent set as follows.

Parameters

B Set of bursts indexed by b

$\delta_b^{\max}$ Maximum amount of time by which b can be delayed while still able to meet the end-to-end delay (assuming routing is known)

t_b^1, t_b^2 Beginning/end of the burst

Variables

x_b =1 if burst b is successful routed up to its destination, 0 otherwise.

δ_b amount by which burst b can be delayed at the source (routing is known) while still able to meet the end-to-end delay, i.e., δ_b + travel time < end-to-end delay $\Rightarrow \delta_b < \delta_b^{\max}$

$y_{bb'}$ = 0 if b and b' overlap and b has been launched before b', 1 otherwise.

$$\max \quad \sum_{b \in B} x_b \qquad \text{number of bursts}$$

$$\max \quad \sum_{b \in B} (t_b^2 - t_b^1) x_b \qquad \text{throughput}$$

subject to:

$$x_b + x_{b'} \leq 1 + y_{bb'} \qquad b, b' \in B \tag{1}$$

$$x_b + x_{b'} \leq 1 + y_{b'b} \qquad b, b' \in B \tag{2}$$

$$-My_{bb'} \leq t_b^2 + \delta_b - t_{b'}^1 - \delta_{b'} \leq M(1 - y_{bb'}) \qquad b, b' \in B \tag{3}$$

$$-My_{b'b} \leq t_{b'}^2 + \delta_{b'} - t_b^1 - \delta_b \leq M(1 - y_{b'b}) \qquad b, b' \in B \tag{4}$$

$$x_b \in \{0, 1\} \qquad b \in B \tag{5}$$

$$0 \leq \delta_b \leq \delta_b^{\max} \qquad b \in B \tag{6}$$

where M is an arbitrarily large constant.

The number of decision variables of the above model is equal to the number of bursts. In the simulations of our heuristic algorithms, even for a small amount of time, we have taken into account 16,000 bursts on average. In practice, the time is continuous and the amount of bursts will be very large even if we slice the continuous time in pieces of fixed size (i.e., small windows). But doing so, we may have difficulties to solve the above exact scheduling mathematical model in reasonable time. For those reasons, we did not implement the above Integer Linear Programming (ILP) model in our simulations due to the scalability issue. Since we did not implement the optimization model, it is very difficult to compare how far the simulation results of our heuristic algorithms are from the optimal one. Consequently, the empirical throughput upper bounds found in the simulations of this thesis are valid as long as the burst scheduling we use (i.e., first in first scheduled) has similar performance.

There is another critical issue in scheduling the bursts in a tree network. In practice, multiple trees coexist in a mesh network and usually they overlap each other. Overlapping trees can be defined as a set of trees that share one or more common links to transmit bursts. Scheduling the bursts in one tree does not take into account the bursts from other overlapping trees which may cause more burst losses than anticipated. Thus, optimal scheduling in one tree does not necessarily represent the optimal scheduling in the overall network.

Though the optimal scheduling was explained in the context of star structure, we recently found out that it is also valid under more general assumptions, i.e., as soon as the routes of the network are given even if the network is not a tree.

Chapter 3

Literature Review

Optical switching has been recognized as a promising technology to achieve an all optical network to take the full advantage of optics. Optical Circuit Switching guarantees the signal transmission thus is more reliable than Optical Packet Switching. However, in terms of bandwidth utilization, OPS outperforms OCS though it experiences some technological limitations. Optical Burst Switching tries to balance and combine the benefits of OCS and OPS while overcoming their demerits. Due to some unique behavior, OBS still rises up several unresolved issues and draw sufficient attention of researchers. Now a days, these switching technologies are being immensely studied to examine their performance and yet to be matured. This chapter provides a detailed overview of literature on optical burst switching, thus demonstrates the motivation for OBS as an alternative to existing optical switching paradigms.

3.1 WDM Networks

Like any other newly invented technology, optical networks are also evolving very rapidly. First generation of optical networks are simple point-to-point networks. Optical fibers are used as a faster transportation medium over copper cables. At that point, Wavelength Division Multiplexing (WDM) technology was invented to have more than one wavelength per fiber and, as a result, significantly increases the capacity. At each intermediate switch, all the signals in a wavelength are terminated and converted to electrical domain. These signals are remodulated to the optical carrier later and transmitted to the desired output ports of the switch [40].

WDM networks of the second generation are capable of establishing connection oriented end-to-end lightpaths [34] in the optical layer. Optical/Wavelength add/drop multiplexers (OADM/WADM) have been introduced to establish a lightpath. Traffic can be added or dropped at the nodes where the WADMs are located. WADMs allow selected wavelengths on a fiber to be dropped. The remaining wavelengths on the fiber can pass through without any modification. The lightpaths are operated and managed on a virtual topology over the existing physical topology. Both first and second generation WDM networks are deployed in various operational networks.

Next generation WDM networks will be all optical networks that avoid the need of conversion from optical to electronic domain at the intermediate switches by providing switching and routing services at the optical layer. All-optical WDM networks can be categorized as the broadcast and select architecture, the wavelength routing architecture and the optical packet switching architecture [40]. Mukherjee in [33] provides an overview of development of such networks. Broadcast and select architectures are sometimes known as single-hop networks [32] and mostly refer to access network area. Wavelength routing and optical packet switching networks are intended for core networks.

Wavelength routing networks are not exact equivalent to electronic circuit switching networks [2]. The major deviation is that the connections in WRN are distinguishable at the intermediate switches, which is not applicable to the connections in electronic circuit switching. WRNs are able to perform some routing tasks in optical domains and reduce the processing overhead for the traffic that desire to pass through some intermediate switches [40]. It also can establish a lightpath to transmit data from source to destination.

The concept of a lightpath was first introduced by Chlamtac *et al.* [6]. A lightpath needs to be set up from source to destination by signaling all intermediate switches along the path for every connection in the network. Setting up a lightpath implies that the route for the connection is selected and a wavelength is assigned to carry the data. Now a days, establishing a lightpath becomes more complicated than before since, the capacity of the wavelength has been significantly increased and in most of the cases, one wavelength for one connection does not satisfy the level of efficiency in terms of bandwidth utilization. Thus, traffic grooming need to be considered during lightpath set up. Grooming allows multiple low rate traffic demands to be transmitted in one wavelength. Therefore, setting up a lightpath combines three critical issues: traffic grooming, routing and wavelength assignment. In one word, the whole problem is well addressed in literature as GRWA (Grooming, routing and wavelength assignment) problem. More details about GRWA problem can be found in [59].

There are still a vast amount of outstanding issues that need to be resolved to achieve all optical networks. Mukherjee [33] also discussed issues such as fault management, protection, traffic grooming etc.. An overview of all optical networks from management viewpoint is given by Okamoto [36]. Also some design and performance issues are addressed by Karasan and Ayanoglu in [24].

3.2 Protocols and Burst Definition

3.2.1 Signaling and Reservation Protocols

Optical Burst Switching (OBS), a new switching paradigm, has been designed to combine the coarse-grained circuit switching and the fine-grained packet switching while overcoming their limitations [38]. A signaling scheme must be implemented to reserve resources for bursts at any intermediate node. The signaling in an OBS network is usually implemented by sending a control packet through out-of-band control channel. Several variations of signaling protocols for OBS have been proposed. In this section we briefly investigate some of the most prominent protocols.

Tell-n-Go (TAG)

OBS was first introduced by Qiao and Yoo in [38] and uses one way reservation protocol like *tell-n-go* (TAG) which was developed by Hudek and Muder [17] to use in Asynchronous Transfer Mode (ATM) networks. Though it was not a successful approach in ATM due to the required electronic complexity, Qiao *et al.* [38] have proposed this protocol in the context of OBS networks as its complexity can be avoided by the use of optical switching technologies. TAG protocol transmits a control packet through the out-of-band control channel before any data transmission. Immediately after the control packet transmission, TAG transmits the burst without waiting for confirmation of reservation or acknowledgment from the network. TAG is required to have fiber delay lines at each intermediate switch to store the data burst during the time period when the switch processes the control packet and establish a connection [17].

Just-Enough-Time (JET)

Yoo and Qiao [53] have proposed another reservation protocol named Just-Enough-Time (JET). They have introduced the concept of *offset time* and *delayed reservation* in JET. Control packets in JET are transmitted ahead of a certain amount of time from burst transmission to allow the intermediate switches sufficient time to process the control packets. This time period is called offset time that helps JET to avoid the necessity of FDLs at each intermediate switch. JET also reserves the resources at the switches from the time the burst arrives at the switch rather than just after the processing of the control packet like TAG, which is referred to as delayed reservation. JET releases the resources as soon as the burst leaves. This is made possible because the control packet carries the length information of the burst. It thus improves the efficiency of utilization. Qiao and Yoo in [38] provided a detailed motivation behind JET and showed some experimental results.

Though Yoo and Qiao [54] found that JET has a better performance than TAG in terms of bandwidth utilization, they also pointed out that JET has also very high burst dropping rate. They have extended their work and introduced a scheme with different classes of traffic according to the priority [54, 56]. In [54], the authors showed that the dropping probability significantly decreases for higher priority bursts at the expense of lower priority classes in a two class system, however, the overall dropping probability remains unchanged. They extended their works in [56] with multiple classes and applied M/M/k/k and M/M/k/D models to provide an upper and a lower bounds on dropping probability for each class. A detailed overview of Yoo and Qiao's work can be found in [39].

Just-in-Time (JIT)

Just-in-Time (JIT) is another alternative scheme for OBS proposed by Wei *et al.* [50]. JIT allocates the resources from the time the control packet is received at the intermediate switch until an explicit release packet is received. It is a simpler approach but cannot achieve the same bandwidth utilization as TAG and JET. JIT was evaluated in [51]. Rodrigues *et al.* [43] have devised an extended version of JIT which they called Enhanced Just in Time (E-JIT). E-JIT attempts to improve and optimize JIT by improving the channel utilization and channel scheduling while keeping the advantage of JIT's simplicity in terms of implementation. E-JIT reserves the resources immediately after the arrival of the control packet: (i) if the channel is available immediately or (ii) the channel will be available before the processing of control packet at the switch while JIT only reserves the resources if it is available immediately. They showed E-JIT outperforms JIT. A detailed performance assessment of E-JIT is given in [42] where the authors also observed that network performance is independent of switch configuration time, however, loss probability increases with the increase of processing time of control packets.

Tell-n-Wait (TAW)

Tell-n-Wait (TAW) is a two way signaling scheme where an acknowledgment message is sent after successful reservation. The setup control packet travels through all the pre-selected path of the burst to collect the availability information of the intermediate nodes [20]. At the destination, the scheduler determines the reservation period at each intermediate link and sends a confirm message in reverse direction to reserve the resources. If the confirm message reaches the source, the burst is sent to the network. If the required resource is already occupied, a release message is sent to the destination to release the resources that are already reserved.

All the protocols discussed above are one-way signaling schemes except TAW. JET and JIT are very similar except that JIT employs immediate reservation and explicit release whereas JET uses delayed reservation and implicit release. TAG is outdated to some extent, compared to JET and JIT, and relies on the use of FDLs at each intermediate node. Though TAW offers very low loss rate, the end-to-end delay is large compared to JET due to the round trip setup time.

Virtual Fixed Offset (VFO)

Virtual Fixed Offset (VFO), a new channel reservation protocol, has been proposed by Qiao *et al.* [29, 30]. Qiao *et al.* analyzed the worst case performance of a large set of best-effort on-line scheduling algorithms in an OBS network. They have identified some factors that are mainly responsible for the performance of any on-line scheduling algorithms such as the length of offset time, burst length ratio, scheduling algorithm itself and available data channels. Based on their formulation, they have also proved that all best-effort on-line scheduling algorithms produce the same optimal solution assuming that all bursts are of same length and offset times are equal. They proposed a new channel reservation protocol which they called Virtual Fixed Offset (VFO) time to improve the performance of OBS networks in worst-case and provided some guidelines for offset time setting. VFO mimics the behavior of same offset time for all bursts though it uses variable offset time for different bursts. It also schedules bursts in the order of burst arrival time instead of the arrival time of control packets. The authors proposed to use Fiber Delay Line (FDL) to delay bursts in each node. Their simulation results showed that VFO improved the performance as much as 35% with comparing to another well-known reservation protocol Just Enough Time (JET).

3.2.2 Burst Definition

A burst can be defined as composed of IP packets. It is formed through a process called *Burst Assembly*. Burst assembly is a process in OBS networks where the ingress router assembles a number of packets into a burst to transmit. Previous studies show that burst assembly may have an important role in network performance. Several classes of burst assembly algorithms have been proposed in literature.

Time and Data Driven

Ge *et al.* [13] proposed the first burst assembly algorithm that reduces the self similarity of traffic and shows that the traffic can be shaped by an efficient burst assembly. The algorithm that Ge *et al.* proposed was a timer based algorithm and maintains a logical queue for any given destination D_i. A timer T_i is associated with each queue and starts to count as soon as it receives the first packet and continue to count until it reaches the threshold W_i. Then the burst is created with the received data packets in queue Q_i and the timer T_i reset to 0. The timer T_i starts to count again only when another packet arrives for destination i. The burst is padded with null data if it is too small even after the threshold time. Their experimental results provide an upper bound on delay due to burst assembly and the minimum size of a burst. Xiong *et al.* in [52] proposed another burst assembly that assigns limit on both the time and the maximum size of a burst. A detailed analysis and comparison of these two schemes can be found in [23]. They observed that the latter scheme leads to higher loss rate. However, the delay of burst assembly is smaller compared to the former time based scheme.

Congestion Driven

Kantarci and Oktug [21] proposed a congestion level based assembly algorithm that they called *Adaptive Threshold Based Burst Assembly* (ATh-BA). In the new scheme, they have reused the concept of timer based and size based burst assembly. However, they proposed an adaptive threshold value for both time and size. The threshold values have been proposed to change, considering the congestion level which is represented by the loss rate of the links. Congestion level and the threshold values are inversely proportional. As the congestion becomes lighter, the threshold values are increased to have longer bursts and improve channel utilization. They have analyzed their scheme under various traffic patterns and observed a dramatic decrease in drop rate while the end-to-end delay remains in feasible range.

Kantarci and Oktug have enhanced their work in [22] where they proposed two different approaches to evaluate the congestion level in the network as well as the threshold values. Link loss rate-based burst assembly (LLRBA) focuses on the loss rates on the outgoing links leading to the corresponding destination while Path loss rate-based burst assembly (PLBRA) pays attention to the loss rate along paths. They have evaluated their new schemes on both uniform and heterogeneous environments. LLRBA significantly decreases the loss rate at medium load while PLRBA outperforms LLRBA with higher load under uniform environment. The reverse performance is observed in heterogeneous environment. However, in terms of end-to-end delay, PLRBA achieves better performance in both environments.

Others

In [14], Hayashitani *et al.* proposed another burst assembly technique that supports a fair QoS among the number of hops. They have assessed their scheme with a three class systems and showed that their new scheme improves the fairness by about 140% compared to the conventional system. In a conventional system, the low QoS packets are placed in the head part and high QoS packets in the tail part while assembling a burst. On the contrary, Hayashitani *et al.* changed the ratio of number of packets for each class according to the number of hops.

3.2.3 Burst Scheduling

Scheduling a burst might be one of the most critical issues in OBS networks. Unfortunately, very few works in literature are devoted to burst scheduling. Primarily the OBS scheduling algorithms are focused on filling the voids that are generated by traditional resource reservation protocols.

The issue of wavelength assignment or wavelength scheduling was first addressed by Xiong *et al.* in [52]. They have described a possible architecture for OBS and also have outlined several algorithms in their paper. A simple first-fit (FF) algorithm was first described that uses a round-robin search and assigns the first available wavelength found. They also described a Horizon [47] like algorithm named *latest available unscheduled channel* (LAUC) that assigns wavelength with latest horizon time. LAUC algorithm was also extended in their paper with a new algorithm that utilizes the tiny gaps between other bursts if possible. This new algorithm outperforms both FF and LAUC and named as *latest available unscheduled channel with void filling* (LAUC-VF).

Another burst scheduling technique called *Horizon Scheduling* was proposed by Turner in [47]. Horizon scheduling attempts to minimize the gaps or fragmentation of channels occurred by JET. This technique was proposed to maintain only a single scheduling horizon for each channel and the scheduling horizon was defined as the latest time at which the channel is scheduled to be in use. Its failure to maintain information about the gaps between bursts results a lower bandwidth utilization than JET however it is simpler than JET and easier to implement.

3.3 Throughput Maximization of OBS

In this section, we explain the major contribution which have been done in literature to improve the performance of OBS networks. Since burst loss is the key issue in OBS networks, most of the works focused on it and proposed several directions to decrease the losses. Some conventional attempts have been made to resolve contention by wavelength conversion and deflection routing. Some works focused on pro actively distributing the load across the network to avoid congestion, thus reducing burst losses. A new reservation protocol was proposed in [29, 30] to improve the performance of OBS networks. More recently, some authors have proposed some well thought O/E/O conversion to enhance OBS performances. We will discuss their results in Section 3.3.3.

Teng and Rouskas have identified several aspects that differentiate the problem of burst loss from other network flow problems in [46]. They analyzed that the burst loss depends on the actual load of the links that are unknown. This is why the minimization of burst loss appears as a non linear problem even after using approximation. In [46], they also formulated a linear problem with certain relaxation in order to determine a set of optimal routes that minimizes the burst drop probability over the network. Due to the computational complexity to solve their formulation in large networks, they also have developed another heuristic especially for large networks. They observed that their heuristic and LP formulation outperforms the ILP formulation because they were not able to achieve the optimal solution even after using CPLEX. They claimed that their experimental results for ILP formulation is sub-optimal and still perform better than shortest path routing.

3.3.1 Contention Resolution

Contention is considered as a major problem in OBS network considering that optical buffers are not available at core routers. Contention causes most of the burst losses in OBS networks and already paid enough attention in literature for resolution. Many approaches were made using conventional methods of contention resolution such as deflection routing. Also, some works have been done using burst segmentation.

Wavelength Conversion and Deflection Routing

Kim *et al.* have proposed the first intra-class contention resolution scheme in [25]. They have implemented an enhanced alternate routing approach which combines wavelength conversion and enhanced deflection routing scheme. They proposed a hop-by-hop routing function at every core router and also consider that the core routers will maintain a routing table with two alternating paths for each destination. If contention occurs, one burst will be routed through a shortest path and the other one will take an alternate route. Since the alternating route may undergoes longer hops than the shortest paths, the data burst may arrives before the control packet at some core routers. The authors have introduced an additional offset time called *Routing Offset Time* to avoid such instance in the network. Though the authors showed that their approach significantly decreases the blocking probability comparing to conventional OBS, they have conducted their experiments in 4×4 Manhattan networks with four ingress nodes only; which does not sufficiently proves the performance of their scheme at highly loaded real life network.

A variant of deflection routing approach was proposed by Lee *et al.* [27] for contention resolution which they called *Contention-based Limited Deflection Routing* (CLDR). CLDR dynamically decides if the burst should be deflected or retransmitted from the source, based on certain criteria. It also intelligently selects the deflected routes based on some performance measure and proposed to maintain a *Deflection Routing Information Base* (DISB) at the edge nodes. Their performance measurement includes minimizing both distance and blocking probability due to contention. They have evaluated their analysis through simulation and showed that CLDR outperforms shortest path deflection. Another work that focuses on alternate routing has been performed by Li *et al.* [28]. They proposed an *Adaptive Alternate Routing Algorithm* (AARA) that selects routes by distributing the load between two pre-determined link-disjoint alternative paths for a given pair of source and destination nodes. Balancing the load helps to avoid congestion in the network and significantly reduces the burst loss compared to static alternate routing.

Burst Segmentation

Burst segmentation was first proposed as a contention resolution approach by Vokkarane *et al.* in [49]. In this newly proposed scheme, only a portion of the contending bursts that overlaps with other bursts will be dropped instead of the whole burst. They have implemented a modified tail-dropping policy to determine the segment of the contending burst to drop. The tail of a burst is dropped only if the number of segments in that tail is less than the number of segments in the whole contending burst; otherwise the entire contending burst will be dropped. They have investigated the burst segmentation scheme with deflection and evaluated the performance in a 14-node NSFNET topology. The experimental results demonstrate that the segmentation with deflection policy performs better than any other standard dropping policy and even offer best performance at high loads. With burst segmentation, some new challenges arise such as switching time, how to detect segment boundary, trailer creation etc. and require additional processing. It may also lead to some side effects such as burst fragmentation and faces difficulties to implement with longer bursts.

Several detailed comparative studies of contention resolution policies can be found in [55, 12, 11, 58]. Gauger *et al.* [12, 11] compared three different contention resolution schemes: Wavelength conversion, deflection routing and fiber delay lines; in terms of loss probability and end-to-end delay. They have observed that the performance of contention resolution schemes extremely depends on offered load and the dimensioning of given nodes and links. Another study has been done comparing the deflection routing, FDL, dropping and re-transmission policies in [55]. They compared these schemes in terms of wavelength utilization and end-to-end delay. A framework that estimates path blocking probabilities considering the reduced load approximation was developed by Zalesky *et al.* [58]. Their framework can also simulate any combination of resolution policies within limited wavelength conversion, burst deflection and burst segmentation.

3.3.2 Balance Loads Using Drop Rate Estimation

Load balancing appears as an efficient approach to avoid congestion in the networks. To balance loads, alternate routing was preferred by a number of studies to shortest path routing. Several studies have proposed new formulations to estimate blocking probabilities of a given link. The newly proposed formulations were utilized to obtain alternate routes for bursts with lower dropping probability.

Rosberg *et al.* [44] proposed a unified model to evaluate blocking probabilities in an OBS network. Their approximation formulation takes into account the reduced offered load due to blocking in earlier stages. They have evaluated the accuracy of their approximation by simulation in NSFNET and noticed that the introduced error is very small. They also observed that the burst segmentation policy has the lowest blocking probability compared to other policies.

In [37], Phùng *et al.* described a unique phenomenon of OBS network named *streamline effect*. Based on their analysis, they proposed a formulation to compute loss probability that is more accurate than Erlang B formula. Streamline effect demonstrates that the bursts within one input stream do not contend among each other and, only contend with those from other input streams. They observed two significant folds of this streamline effect. Firstly, they analyzed that the burst loss probability is lower than that obtained from M/M/k/k queuing model. Another substantial observation was the non uniform loss probability of bursts among the input streams. It inversely relates to the burst rate of the input streams. The total loss probability of an OBS network will be reduced if a large flow of traffic is formed with a minimum number of merging flows. Streamline effect has major implications for QoS and load balancing algorithms. An accurate loss probability estimation helps to obtain a better set of routes that provide higher success rate for bursts transmission. Phùng *et al.* have also proposed a load balancing scheme that uses the idea of streamline effect and showed that the scheme outperforms shortest path routing.

Chen *et al.* [4] have extended the study with streamline effect and developed a MILP formulation to compute primary and backup paths for each flow while minimizing the burst loss. Due to computational complexity, they were not able to implement their formulation in a real time mesh network topology. However, they have developed a heuristic that utilizes the formulation to compute the loss probability. The heuristic randomly selects a number of routes and re-route them considering the loss probability given by the newly proposed formula. Authors in [4] claimed that their heuristic outperforms other known algorithms in terms of burst loss.

3.3.3 Recent Improvements

A translucent node architecture has been proposed for OBS networks by Coutelen *et al.* in [9]. The newly proposed architecture allows re-aggregation of bursts at intermediate nodes in core networks. This architecture provides electrical buffers at intermediate nodes and utilize aggregation grooming. The simulation results show that the aggregation reduces the loss probability in OBS networks. However, it increases the end-to-end delay by around 30% and the cost of the node by around 20%. To overcome these drawbacks, the authors proposed to use aggregation grooming which according to their experiments, significantly reduces the loss rate. The authors, in [9], also proposed an accurate traffic model LCH^+ for OBS networks after a careful analysis of traffic properties. LCH^+ is an enhancement of *Lost Call Held* (LCH) model. It combines the finite number of sources and the independent arrival property while discards the burst segmentation.

In [8], Coutelen *et al.* have been enhanced their previous work in [7] and proposed an improved RWA-OBS formulation. RWA-OBS problem performs routing and wavelength assignment for OBS networks. RWA-OBS differs from classical RWA for OCS and WR-OBS by allowing merging of ingress flows and transit flows if the transit flows are OT-isolated and the ingress nodes run ALAP. In this enhancement, the authors introduce the flexibility to use different OT extension factor (EOT) while the previous work only assumes a constant EOT for each connection. The authors intelligently managed the scalability of the enhanced model by employing column generation technique combined with a Tabu Search heuristic.

Although these recent studies have a very high interest in order to increase the meaningfulness of OBS networks, they are not taken into account in the conditions under which our experiments provide an upper bound on the throughput of an OBS network.

Chapter 4

Shortest Path Selection

In this Chapter, we propose the first approach that we have designed to maximize the throughput of an OBS network. In this approach, we experiment two algorithms to select a path from a set of k-shortest paths to transmit bursts for any s-d pair in the network. First algorithm defined as Sequential Selection of Shortest Paths (SSP), selects the path that assures a burst the smallest time to reach the destination. Second algorithm referred as Weighted Selection of Shortest Paths (WSP), introduces a weight on each path based on certain criteria and selects the path with smallest weight. Sections 4.1 and 4.2 describe the assumptions and notations respectively that we have considered throughout the thesis. Section 4.3 explains the resource reservation process. Algorithms are illustrated in Sections 4.4 and 4.5. Section 4.6 presents the comparative results of SSP and WSP algorithms.

4.1 Assumptions

We have considered several assumptions to maximize the throughput of an OBS network.

Wavelength Continuity Constraint: No wavelength conversion capability is available at core nodes. Thus a light-path must use the same wavelength on all the links along its path from source to destination.

Bufferless Core Node: We assume that no mechanism is used to delay the bursts at core nodes.

Link: Any pair of connected nodes are linked by two directional links. Each link is supported by a directional optical fiber.

Traffic: Traffic is static and the traffic matrix is known in advance.

4.2 Notations

This section depicts the notation we have used to represent network topology and traffic.

4.2.1 Network Topology

We consider an OBS network represented by a directed graph $G = (V, L)$ where V is associated with the set of nodes and L corresponds to the set of (directed) links between the nodes. The set of nodes is divided into the set V_{ACCESS} of access nodes and the set V_{CORE} of core nodes. On each fiber link, the transport capacity is made of W wavelengths. Let $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_W\}$ be the set of wavelengths. Connection requests are received between pair of access nodes.

For each pair of source and destination $(v_s, v_d) \in V_{\text{ACCESS}} \times V_{\text{ACCESS}}$, let

$\mathcal{P}_{sd}$ be the set of available routes. If $|\mathcal{P}_{sd}| = 1$, we will assume that $p_{sd} \in \mathcal{P}_{sd}$ corresponds to one of the shortest routes from v_s to v_d.

$t_{sd}^{\max}$ be the maximum end-to-end delay, after which the burst will be too late. Therefore, one has to disregard from the source nodes, the bursts which cannot be launched early enough in order to reach their destination before the maximum allowable end-to-end delay has expired. A burst is stored in the electrical buffers that are available at the sources before it is transferred to the optical domain.

p_{sd} be a path from v_s to v_d.

$\delta_{p_{sd}}^{\text{PROPAG}}$ be the propagation delay on path p_{sd}. It is the cumulative sum of propagation delays on each link $\ell \in p_{sd}$.

Link

For each link $\ell \in L$, let

LENGTH(ℓ) be the length of link ℓ.

s_ℓ be the source node,

d_ℓ be the destination node,

$\delta_\ell^{\text{PROPAG}}$ be the propagation delay, usually expressed in ms. It is equal to $\frac{\text{LENGTH}(\ell)}{v_{\text{LIGHT}}}$ where $v_{\text{LIGHT}} = 200,000$ km/seconds in the typical glass which is used for optical fibers.

T_ℓ^λ be the list of time slots for wavelength $\lambda \in \Lambda$ on link ℓ.

BD-USAGE(ℓ) be the used bandwidth on link ℓ.

Node

For each node $v \in V$, let

$\text{BUFFER}(v)$ be the size of the electrical buffer at source node $v \in V_{\text{ACCESS}}$,

$\delta_v^{\text{SWITCHING}}$ be the switching delay at node $v \in V$. In OBS networks, this delay is negligible at the intermediate nodes as, at the burst arrival, switching configuration has been done following the early arrival of the burst header, consequently data remain in the optical domain. The switching time is however not negligible at the source node, especially for the conversion from the electronic domain to the optical domain. As this is a fixed delay, we will assume that it is taken into account in the limit on the end-to-end delays.

4.2.2 Traffic

We consider a static traffic model where, for each source node $v \in V_{\text{ACCESS}}$, we are given a list B_v of bursts to launch at some time from $v \in V_{\text{ACCESS}}$. For each $b \in B_v$, let

$\text{LENGTH}(b)$ be its length. It can be expressed either in terms of the number of packets (and number of bytes/packet) or in ms or μs.

$\text{DEST}(b)$ be its destination,

t_b^{READY} be the earliest time at which b can be launched from v,

t_b^{LAUNCH} be the time at which b is launched from v,

$\delta_b^{\text{LAUNCH}} = t_b^{\text{LAUNCH}} - t_b^{\text{READY}}$ be the launch delay after t_b^{READY}.

Let $B = \bigcup_{v \in V_{\text{ACCESS}}} B_v$ be the overall set of bursts to be launched over the set of access nodes. The procedure of traffic generation is explained in Section 4.6.2.

4.3 Resource Reservation

This section describes the process of resource reservation that we followed in the simulations. Let us consider a pair or source and destination nodes v_s and v_d, and let us assume that the shortest path from v_s to v_d has three fiber links ℓ_1, ℓ_2 and ℓ_3, see Figure 8. Let t_b^{LAUNCH} the time at which a given burst b is launched:

$$t_b^{\text{LAUNCH}} \geq t_b^{\text{READY}} \qquad \delta_b^{\text{LAUNCH}} = t_b^{\text{LAUNCH}} - t_b^{\text{READY}} \qquad t_b^{\text{DEST}} - t_b^{\text{READY}} \leq t^{\max}$$

where t_b^{DEST} is as defined below in equation (7).

Time of arrival of b at the entrance of node v_a:

$$t_{v_a}^{\text{ENTRANCE}} = \quad t_b^{\text{READY}} + \delta_{\ell_1}^{\text{PROPAG}},$$

where LENGTH(b) is expressed in bits and where $\delta_{\ell_1}^{\text{PROPAG}}$ = TRAVEL TIME OF THE FIRST BIT OF b THROUGH ℓ_1, see Figure 8 for an illustration.

Time of arrival of b at the exit of node v_a:

$$t_{v_a}^{\text{EXIT}} = t_{v_a}^{\text{ENTRANCE}} + t_{v_a}^{\text{SWITCHING}}.$$

and so on.

Time of arrival of b at the entrance of node v_d:

$$t_b^{\text{DEST}} = t_{v_d}^{\text{ENTRANCE}} = t_{v_b}^{\text{EXIT}} + \delta_{\ell_3}^{\text{PROPAG}} + \frac{\text{LENGTH}(b)}{\text{CAPACITY}(\ell_3)}, \quad (7)$$

where LENGTH(b) is expressed in bits and where $\delta_{\ell_3}^{\text{PROPAG}}$ = TRAVEL TIME OF THE FIRST BIT THROUGH ℓ_3, see again Figure 8 for an illustration.

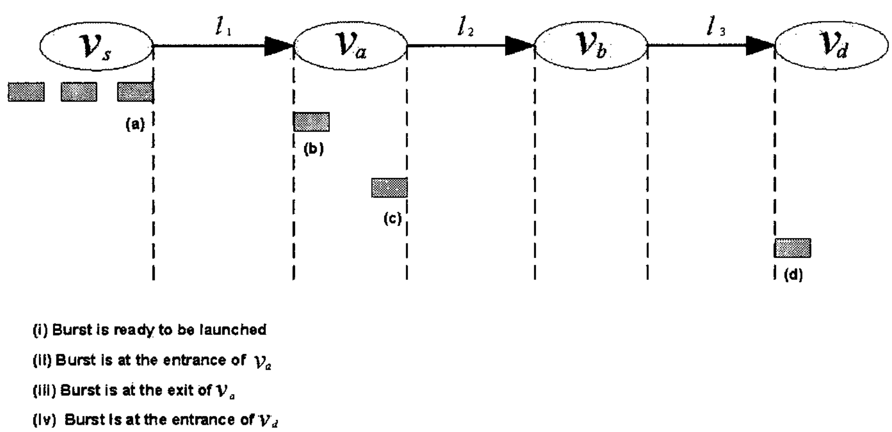

Figure 8: How to compute the end-to-end delay

4.4 Sequential Selection of Shortest Paths (SSP)

This section explains the first algorithm that we have designed to select a path among the set of k-shortest paths for a given (v_s, v_d) pair. The set of k-shortest paths for each (v_s, v_d) pair is obtained by Eppstein's [10] algorithm. One path is selected from the set of k-shortest paths to transmit the burst based on the time the burst reaches the destination.

We have implemented Eppstein's algorithm that obtain k shortest paths for all (v_s, v_d) pairs in a given network. More precisely, we have used an adapted version of Eppstein's algorithm to obtain a set of three loop-less paths for all (v_s, v_d) pairs in the network. Eppstein's algorithm was selected due to its several advantageous characteristics. It provides a better memory management by using efficient data structures than other similar algorithms available in [1, 26]. Most important element is the time complexity it requires to obtain k-shortest paths. In a $n \times m$ directed graph, it can provide the set of k-shortest paths for a given (v_s, v_d) pair by $O(m + n \times \log n + k)$ time.

Once the set of k-alternating paths for each (v_s, v_d) pair is acquired, our algorithm dynamically selects one path from every set for a given (v_s, v_d) pair to transmit bursts. The algorithm sequentially looks for available time slots for a given burst in every alternating paths. Based on the availability of resources, the algorithm estimates the time when the given burst can reach its destination if it follows a given path. The algorithm compares the estimated arrival times and choose the path with the smallest one. This selection process was aimed to reduce the overall end-to-end delay in the network. The selected path is used to transmit burst for the given (v_s, v_d) pair. The detailed algorithm of SSP is given in Step 1 and 2. Step 1 defines the initializations that are necessary for the next part of the algorithm. Step 2 explains the resource reservation process that we have explained above.

Algorithm SSP

Step 1 Initialization

{Initialize the data structures}
for all $\ell \in L$ **do**
  for all $\lambda \in \Lambda$ **do**
    $T_\ell^\lambda \leftarrow \emptyset$
  end for
end for
for all $v \in V_{\text{ACCESS}}$ **do**
  BUFFER(v) $\leftarrow \emptyset$
end for
{Initialize the parameters}
for all $(v_s, v_d) \in V_{\text{ACCESS}} \times V_{\text{ACCESS}}$ **do**
  compute $\mathcal{P}_{sd}$ = set of k-shortest paths from $v_s to v_d$
end for.

4.5 Weighted Selection of Shortest Paths (WSP)

We have investigated another scheme to select a path among the set of k-shortest paths for burst transmission. The major idea was to distribute the load over the available k-alternating paths. Load distribution is a well known strategy to reduce congestion in the network and reduce the dropping rate in the network. In this approach, we try to select the path that is least loaded among the set of k-alternating routes.

We obtain the set of k-alternating routes for each (v_s, v_d) pair in a same manner as explained in Section 4.4. Unlike in the previous strategy, we do not select the path according to the shortest arrival time of bursts at the destination. Instead, we start our search of available time slots from the path that is least loaded and continue our search in a round robin fashion through the set of available paths. As soon as we obtain a free time slot to launch the burst, we stop our search and use that path to transmit the burst. We associate a new attribute named *weight* to each path so that we can keep track of highly loaded paths.

Step 2 Current Step

while $B \neq \emptyset$ **do**
  Let $b \in B$ be the earliest non routed burst;
  Let v_s be its source node and v_d be its destination node;
  $B \leftarrow B \setminus \{b\}$;
  Update T_ℓ^λ for all $\ell \in L$ and for all $\lambda \in \Lambda$;
  Update weights for all $p_{sd} \in \mathcal{P}_{sd}$
  next_burst $\leftarrow$ .FALSE.
  while next_burst $\neq$ .TRUE. **do**
    $\hat{t}_b^{\text{DEST}} \leftarrow \infty$;
    for all $(p_{sd}, \lambda) \in \mathcal{P}_{sd} \times \Lambda$ in their increasing weight order **do**
      success = .TRUE.
      while success. **do**
        Consider the next link on (p_{sd}, λ).
        If there exists no void interval $I_\ell = [\alpha_\ell, \beta_\ell]$ of length LENGTH(b) such that:

$$\alpha_\ell = \alpha_{\ell'} + t_{\ell'}^{\text{PROPAG}} \qquad \ell \neq \text{first link on } p_{sd}$$

with ℓ' being the preceding link of ℓ on p_{sd}

        then
        success $\leftarrow$ .FALSE.
      end while
      if success **then**
        Compute δ_b^{LAUNCH} ;
        Compute the expected time arrival, t_b^{DEST}, at the destination node.
        if the buffering capacity limit is not exceeded **and** $t_b^{\text{DEST}} - t_b^{\text{READY}} \leq t^{\max}$ **then**
          $\hat{t}_b^{\text{DEST}} \leftarrow t_b^{\text{DEST}}$;
          $\hat{\lambda} \leftarrow \lambda$;
        end if
      end if
    end for
    if $\hat{t}_b^{\text{DEST}} \neq \infty$ **then**
      Update $T_\ell^{\hat{\lambda}}$ for all $\ell \in p_{sd}$
      next_burst $\leftarrow$ .TRUE.
      Update weights for all $p_{sd} \in \mathcal{P}_{sd}$
    end if
    if next_burst = .FALSE. **then**
      Disregard (drop) the burst;
      next_burst $\leftarrow$ .TRUE.
    end if
  end while
end while

We compute and update the weights for all paths after reserving the resources for every burst. We consider two different criteria to estimate the weight value for each path. One criterion is the current load or used bandwidth of all links in a path. We also intuitively consider the length of the path as another important criterion in path selection.

Criteria 1:

$$\mathrm{W}_{sd}^{1} = \frac{\sum\limits_{\ell \in p_{sd}} \text{LENGTH}(\ell) \times \text{BD-USAGE}(\ell)}{\sum\limits_{\ell \in p_{sd}} \text{LENGTH}(\ell)} \tag{8}$$

Criteria 2:

$$\mathrm{W}_{sd}^{2} = \max_{\ell \in p_{sd}} \{\text{LENGTH}(\ell) \times \text{BD-USAGE}(\ell)\} \tag{9}$$

These two criteria referred in expressions (8) and (9) were used to compute the total weight of path p_{sd}. We also have used another factor denoted by α to increase the flexibility of having different levels of precedence on these criteria. In our experiments, we have assigned same precedence for both criteria by setting the value of α as 0.5. We finally propose expression (10) to estimate the weight of a path.

Weight of path p_{sd}:

$$\mathrm{W}_{p_{sd}} = \alpha \times \mathrm{W}_{sd}^{1} + (1-\alpha) \times \mathrm{W}_{sd}^{2} \tag{10}$$

where we use $\alpha \in [0,1]$ as a factor to customize the precedence of one factor over another.

4.6 SSP vs. WSP

4.6.1 Network Instances

Simulations have been conducted on three different network topologies. Different traffic patterns have been used for each of these three networks. Each of the link has two wavelengths and each wavelength has a capacity of 10 Gbps.

NSF

We consider NSF (National Science Foundation) network with 14 nodes and 21 bi-directional links as shown in Figure 9. The NSF network was a major contribution to the Internet backbone of early 1990s and designed to create an open network to allow the opportunity to access supercomputers for academic researchers [15].

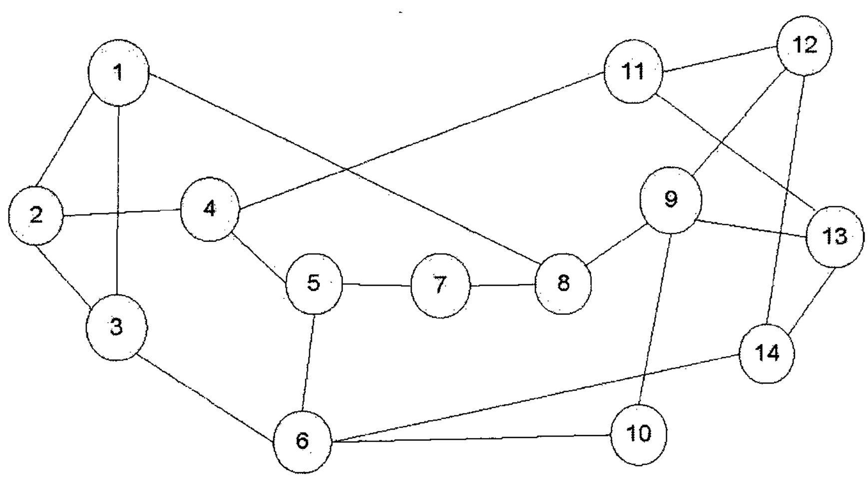

Figure 9: NSF network

Step 2 Current Step

```
while B ≠ ∅ do
  Let b ∈ B be the earliest non routed burst;
  Let v_s be its source node and v_d be its destination node;
  B ← B \ {b};
  Update T_ℓ^λ for all ℓ ∈ L and for all λ ∈ Λ;
  next_burst ← .FALSE.
  while next_burst ≠ .TRUE. do
    t̂_b^DEST ← ∞ ;
    for all p_sd ∈ 𝒫_sd in their increasing length order do
      λ_min ← 1;
      minLoad ← ∞ ;
      for all λ ∈ Λ do
        load_λ = 0;
        for all ℓ ∈ p_sd do
          load_λ + = load_λ^ℓ
        end for
        if load_λ < minLoad then
          minLoad ← load_λ ;
          λ_min ← λ ;
        end if
      end for
      λ = λ_min;
      count = 0;
      while success ≠ .TRUE OR count ≤ W do
        for all ℓ ∈ p_sd do
          Find if there exists a void interval I_ℓ = [α_ℓ, β_ℓ] of length LENGTH(b) such that:

                α_ℓ = α_ℓ' + t_ℓ'^PROPAG        ℓ ≠ first link on p_sd
                                        with ℓ' being the preceding link of ℓ on p_sd

          if there exists such a set of intervals then
            Compute δ_b^LAUNCH ;
            Compute the expected time arrival, t_b^DEST, at the destination node.
            if the buffering capacity limit is not exceeded and t_b^DEST − t_b^READY ≤ t^max then
              t̂_b^DEST ← t_b^DEST ;
              λ̂ ← λ ;
              success ← .TRUE;
              BREAK;
            end if
          end if
        end for
        count = count + 1;
        λ = λ + 1;
      end while
    end for
    if t̂_b^DEST ≠ ∞ then
      Update T_ℓ^λ̂ for all ℓ ∈ p_sd
      next_burst ← .TRUE.
    end if
    if next_burst = .FALSE. then
      Disregard (drop) the burst;
      next_burst ← .TRUE.
    end if
  end while
end while
```

EON

The second network is the European network (EON2004) defined by IST (Information Society Technologies) project as LION & COST action 266 [31]. The EON2004 topology with 28 nodes and 41 bi-directional links is illustrated in Figure 10.

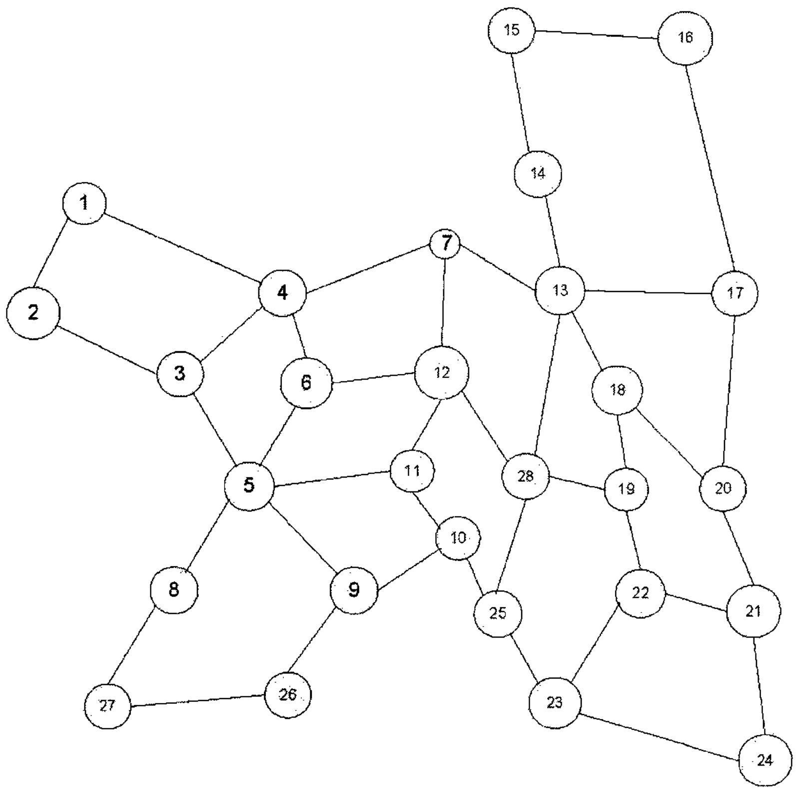

Figure 10: European network

NY

Another network is the New York network (NY) defined in [16]. The NY topology with 16 nodes and 98 bi-directional links is illustrated in Figure 11. Similar traffic scenarios as NSF and EON2004 have been applied to NY to examine the network performance.

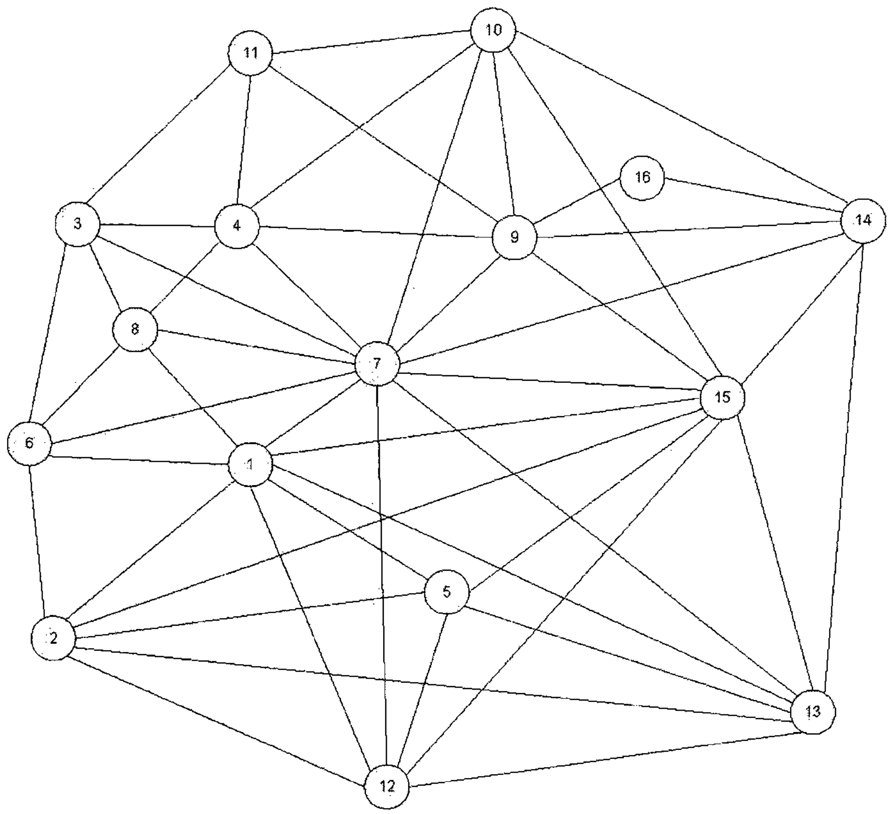

Figure 11: NY network

4.6.2 Traffic Instances

We have considered different traffic instances with various patterns for each network instance.

In one instance, we generate traffic for every source-destination *(s-d)* pair in the network and the amount of traffic is uniformly distributed over the *s-d* pairs. The amount of traffic varies for a given network instance and a given heuristic approach. Because we consider the amount of traffic load for each s-d pair such that the overall drop rate in the network remains around 1%.

We also considered non-uniform traffic for all network instances. Two instances of non-uniform traffic patterns have been examined. In the first instance, traffic load varies about 10% between minimally loaded *s-d* pair to maximally loaded s-d pair in one instance. In another instance, load varies about 50% between min and max loaded *s-d* pairs. Amount of traffic load for each *s-d* pair is randomly selected.

In the simulations, the incoming bursts follow Poisson arrival pattern and the length of the bursts are uniformly distributed. A burst contain 200 number of packets and each packet has a size of 1250 bytes.

4.6.3 Simulation Environment

The parameter $t_{sd}^{\max}$ appears as an important component in our experiments. It defines the maximum time we allow a burst to wait at the sources. All of the bursts dropping that are experienced in the simulations are due to $t_{sd}^{\max}$. We have used a normalized value of $t_{sd}^{\max}$ defined in equation 12 in the simulations. Since, all of the algorithms use analogous value of $t_{sd}^{\max}$, varying the value of $t_{sd}^{\max}$ will not change the order of the algorithms in performance comparison tables. We consider both the average end-to-end propagation time, say t^{AVERAGE}, over all s-d pairs in the network and the largest propagation delay among the shortest paths in the network. t^{AVERAGE} can be defined as:

$$t^{\text{AVERAGE}} = \frac{\sum_{p_{sd} \in \mathcal{P}_{sd}} \delta_{p_{sd}}^{\text{PROPAG}}}{|\mathcal{P}_{sd}|} \tag{11}$$

$t^{\max}$ can be defined as follows:

$$t^{\max} = 50 \text{ ms } + \max \left\{ t^{\text{AVERAGE}}; \max_{(v_s,v_d) \in V_{\text{ACCESS}} \times V_{\text{ACCESS}}} \text{end-to-end propagation delay} \right.$$
$$\left. \text{on the shortest path from } v_s \text{ to } v_d \right\}. \quad (12)$$

4.6.4 Comparison

Tables 3 and 4 show the comparative results for SSP and WSP with both NSF and EON networks. Table 1 and 2 present the average E2E delay and the average travel time experienced by the bursts when using the path selected by SSP and WSP in NSF and EON networks respectively.

Table 1: Comparison of SSP & WSP in NSF

NSF Network			
Algorithms	**Average E2E delay** (μs)	**Average travel time** (μs)	**Average waiting time** (μs)
SSP	3070.0	2472.6	597.4
WSP	4312.4	3046.9	1265.5

Table 2: Comparison of SSP & WSP in EON

EON Network			
Algorithms	**Average E2E delay** (μs)	**Average travel time** (μs)	**Average waiting time** (μs)
SSP	4047.3	3337.6	709.7
WSP	5168.8	3744.4	1424.4

Simulation results show that WSP increases the throughput 6% in EON2004 and 5% in NSF over SSP however, WSP experiences larger average E2E delay over SSP. We presume two reasons for larger delays experienced by WSP as follows. (i) WSP may select longer

Table 3: Comparison of throughputs for SSP & WSP in EON

EON with Uniform traffic		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
SSP	141.4	1.0
WSP	160.6	0.9

Table 4: Comparison of throughputs for SSP & WSP in NSF

NSF with Uniform traffic		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
SSP	107.3	1.0
WSP	113.1	0.9

paths than SSP to transmit bursts. (ii) WSP may experience larger amount of conflicts in the selected path since the possibility of conflicting bursts increase with the increase of the length of the path. The reasons can be explained by following example.

Let us consider an instance in our simulations in NSF network where bursts need to be transmitted from source node $v_s = 3$ to destination node $v_d = 1$. For this specific instance, SSP selects the path $3 \rightarrow 1$ to transmit the bursts for a certain period of time. On the other hand, WSP selects the path $3 \rightarrow 6 \rightarrow 5 \rightarrow 4 \rightarrow 2 \rightarrow 1$ to transmit the bursts which is much longer than the path selected by SSP.

Chapter 5

Load Balancing

Balancing load is an innovative concept that distributes the load over the network as evenly as possible. This approach helps to avoid congestion in the network and reduces the probability of dropping a burst as well. Now-a-days, load balancing has been given huge attention in research community to employ it in OBS networks. Several approaches have been proposed to distribute loads over the network. Studies related to load balancing in OBS networks are described in Section 3.3. In this chapter, we propose a novel heuristic to balance loads in an OBS network. Section 5.1 describes the idea of load balancing. We present the algorithm of our heuristic in Section 5.2. Section 5.3 illustrates a heuristic from literature and the heuristic that we designed by taking into account the *streamline effect*. Comparison of the simulation results are presented in Section 5.4.

5.1 An Example of Load Balancing

Load balancing is a novel approach in OBS networks that attempts to reduce the probability of dropping bursts in the network. The major focus of load balancing is to avoid congestion in the network since congestion causes a good amount of burst dropping. Load balancing tries to spread out the load all over the network to avoid congestion.

A typical way to find a path for a given (v_s, v_d) pair in a network is to follow the shortest path. Once all (v_s, v_d) pairs in the network attempt to use shortest paths to transmit data, congestion may occur in the network. One or more links in the network might be overloaded while some links remain under utilized. The overloaded links are referred as bottleneck links and cause most of the burst dropping in an OBS network. Balancing the load over the network can avoid such troubles to occur in the network. A load balancing algorithm typically re-routes some connections through the under utilized links and thus, reduces the load of bottleneck links. Such an approach usually selects a least loaded path for a given (v_s, v_d) pair instead of the shortest path.

Let us consider the network as shown in Figure 12. Assume that all links feature same amount of cost and the capacity of each link is 10 Gbps. Nodes v_a, v_b, v_c and v_d are generating bursts at a rate of 4 Gbps to transmit to destination v_g. If shortest paths are utilized to transmit the bursts, then v_a, v_b, v_c and v_d will transmit bursts through paths $v_a \rightarrow v_h \rightarrow v_g$, $v_b \rightarrow v_h \rightarrow v_g$, $v_c \rightarrow v_h \rightarrow v_g$ and $v_d \rightarrow v_h \rightarrow v_g$ respectively as explained in Figure 13 (a).

Since each connection is generating traffic at a rate of 4 Gbps, the total load offered to link $v_h \rightarrow v_g$ will be higher than its capacity. Thus, some bursts will be dropped due to the congestion occurred in this instance, however, the capacity of some other links in the network are not utilized.

The burst losses in this particular scenario can be avoided by utilizing the capacity of unused links. If we re-route the traffic from sources v_c and v_d through a longer path than the

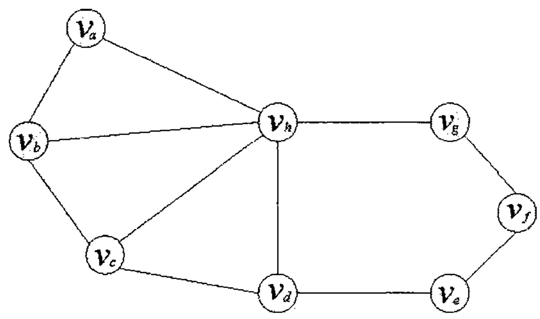

Figure 12: Network instance

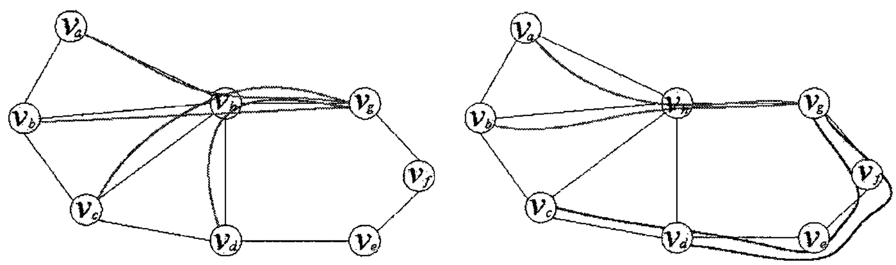

Figure 13: Typical and re-routed connections

shortest path such that the traffic from v_c and v_d travel through $v_c \rightarrow v_d \rightarrow v_e \rightarrow v_f \rightarrow v_g$ and $v_d \rightarrow v_e \rightarrow v_f \rightarrow v_g$ respectively, to reach the destination as explained in Figure 13 (b), then there will be no loss of bursts in the network.

5.2 Load Balancing Heuristic (LBH)

We have implemented a heuristic to find the best possible routes in the network for each (v_s, v_d) pair. In this approach, our objective was to minimize the maximum load of a link and the overall variance of loads in the network. We use the shortest path algorithm to obtain a set of shortest paths for all (v_s, v_d) pairs in the network. The algorithm uses the set of shortest paths as an initial solution. The physical length of the links is used by the routing algorithm as cost of the links. It ensures the shortest arrival time at the destination if the bursts are launched as soon as they are ready.

Once the initial solution is obtained, the heuristic algorithm evaluates the number of connections passing through each link and the amount of load provided by each connection on that link so that the total amount of traffic passing through each link can be computed. We replace the cost of the links by the amount of load it carries. In every iteration, we find the link with uppermost load and refer as the bottleneck link hereafter and denote it by ℓ_b. It is defined as follows: $\text{LOAD}(\ell_b) = \max_{\ell \in L} \text{LOAD}(\ell)$. Then we find the connection c_b such that $\text{LOAD}_{c_b}(\ell_b) = \max_{c \in C} \text{LOAD}_c(\ell_b)$, i.e., the connection that contributes the maximum load to the link ℓ_b. C denotes the set of connections exist in the network. We apply the routing algorithm to find an alternate route for connection c_b that does not use link l_b, so that the load of the link l_b will be decreased. Any rerouting of the connections in the network may change the number of connections passing through a link and the amount of load as well. Therefore we evaluate the cost for every link in the network again.

We evaluate the quality of a given solution by measuring the network weight for the given solution. The weight of the network is measured by the variance of load of the links in the network. The variance is computed as $\sum_{\ell} (\text{LOAD}(\ell) - \text{MEAN}(\text{LOAD}))^2$. A solution is referred as a better solution if it reduces the weight of the network.

The iteration continues the search for possible solutions so that the load is well distributed over the network. We stop our search after a certain amount of iterations with no more improvement. We also maintain a Tabu list to store the previous solutions in memory to avoid repeated selection by the heuristic.

Algorithm LBH (Load balancing Heuristic)

Step 1 Initialization

{Initialize the parameters}
for all $(v_s, v_d) \in V_{\text{ACCESS}} \times V_{\text{ACCESS}}$ **do**
  Compute $\mathcal{P}_{sd}$ = shortest path from v_s to v_d;
end for.
{Initialize the data structures}
for all $\ell \in L$ **do**
  Compute offered load $\text{LOAD}(\ell)$ on ℓ;
  Replace link cost by $\text{LOAD}(\ell)$;
end for
Initialize Tabu list τ;

Step 2 Current Step

Find link ℓ_b such that $\text{LOAD}(\ell_b) = \max_{\ell \in L} \text{LOAD}(\ell)$;
Find connection c_b such that $\text{LOAD}_{c_b}(\ell_b) = \max_{c \in C} \text{LOAD}_c(\ell_b)$;
Compute network weight W_N using variance of load on the links such that $\sum_{\ell} (\text{LOAD}(\ell) - \text{MEAN}(\text{LOAD}))^2$;
if $c_b \notin \tau$ **then**
  Re-route c_b using updated cost function;
  Compute link load and update link costs;
  Compute network weight W_N^{curr};
  $\tau \leftarrow \tau \cup c_b$;
end if
if $W_N^{curr} < W_N$ **then**
  Update the incumbent value;
  $W_N \equiv W_N^{curr}$;
end if

5.3 Algorithms from Literature and Enhancements

5.3.1 Literature

The *Streamline effect*, a unique phenomenon of OBS, has been described in Section 3.3. Streamline was first identified by Phùng *et al.* in [37]. The authors in [37] proposed a new formulation to estimate blocking probability more accurately taking into account the streamline effect. We have applied this estimation in our simulation to compute the loss probability of each link for given network topology and traffic instance. The estimated amount of loss probability for a given link was assigned as cost of that link to obtain the set of routes with smallest loss rate for each (v_s, v_d) pair. Simulation results are presented in Tables 5 and 6.

Qian *et al.* [4] have proposed a different heuristic focused on streamline effect. The heuristic was initialized by the set of shortest paths for all (v_s, v_d) pairs in the network. Loss probability for each link was computed applying the formulation provided in [37]. A random number of paths in the network are selected to re-route. Re-routing considers the estimated loss probabilities of the links and finds the path with the lowest loss rates for all (v_s, v_d) pairs. Simulation results for Qian's [4] heuristic is presented in Section 5.4.

5.3.2 Enhancements

We have developed another heuristic based on the newly identified characteristic *streamline effect* to extend Phùng's work. We define the heuristic as Streamline With Heuristic (STH). The objective of the heuristic is similar to LBH as described in Section 5.2. STH also uses the equivalent set of shortest paths as initial solution. Each iteration of the heuristic follows the steps illustrated in Step 2 of LBH except the procedure of cost estimation of the links.

Unlike LBH, STH estimates the loss probabilities for all links in the network taking into account Phùng's formulation and assign the estimated loss probability as cost of the link. STH also evaluates a given solution in different manner. It computes the overall weight of the network as the cumulative sum of the estimated loss probabilities of the links in the network.

5.4 Comparison

Table 5 shows the simulation results with EON2004. Experimental results with NSF network are presented in Table 6. We observe that (i) both LBH and Qian's [4] heuristic improve network throughput compared to Streamline and (ii) LBH outperforms the Qian's heuristic and STH. The reasons for bursts loss in these simulations are quite similar to the reasons experienced in the simulations of Chapter 4 (i.e., loss occurred at the sources).

Table 5: Comparison of throughputs for LBH in EON

EON with Uniform traffic		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
Streamline	62.6	1.1
STH	116.8	0.9
Streamline-Qian	139.2	1.1
LBH	140.5	0.9

We compare the performances of two most significant load balancing algorithms with WSP in Chapter 7.

Table 6: Comparison of throughputs for LBH in NSF

NSF with Uniform traffic		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
Streamline	76.9	0.9
STH	114.7	0.9
Streamline-Qian	168.5	1.0
LBH	169.5	0.9

Chapter 6

Tree Decomposition

6.1 Outline

This chapter will describe another approach that we have examined to maximize the throughput of OBS networks. The basic idea was to build a tree network for a given destination from any mesh network. The idea was motivated by the phenomenon *streamline effect* explained in [37] which demonstrates that the bursts traveling in one input stream are streamlined and do not contend with each other until they diverge.

The challenge was to decompose the set of efficient trees from any given mesh network. Let us consider the mesh network represented by $G = (V, L)$ where V is associated with the set of nodes and L corresponds to the set of links between the nodes. For a given destination D, we need to define a set of routes for all possible sources such that, all sources can reach destination D using a tree structure. The set of trees in a given network contains as many trees as the number of destinations.

6.2 An Example

To get an upper bound on network throughput, we relax some certain conditions of classical OBS environment. We assume that the overall information of time-slot availabilities at each link is available at scheduling node. We also assume there is no offset time for the bursts. Under these considerations, a tree structure can be represented by an equivalent star network. Rest of this section explain a simple example supporting this claim.

Consider a network represented by a directed graph $G = (V, L)$ where V is associated with the set of nodes and L corresponds to the set of directed links between the nodes. The set of nodes are divided into the set of source nodes V_s and the set of destination nodes V_d. We need to build a tree structure for all $v_d \in V_d$ so that there is only merging flow in the network. Consider node $v_f \in V_d$ and nodes $\{v_a, v_b, v_c, v_d, v_e\} \in V_s$ in the network shown in Figure 14.

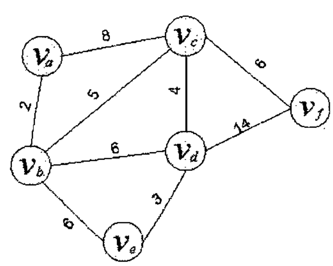

Figure 14: Sample network

A tree can be built for destination v_f from the network shown in Figure 14 assuming that the shortest path from v_a to v_f is $v_a \rightarrow v_b \rightarrow v_c \rightarrow v_f$, v_b to v_f is $v_b \rightarrow v_c \rightarrow v_f$, v_c to v_f is $v_c \rightarrow v_f$, $v_s = v_d$ is $v_d \rightarrow v_c \rightarrow v_f$ and v_e to v_f is $v_e \rightarrow v_d \rightarrow v_c \rightarrow v_f$. Figure 15 shows the corresponding tree for $v_f \in V_d$. The costs associated with links in Figure 14 represents the propagation time of the links.

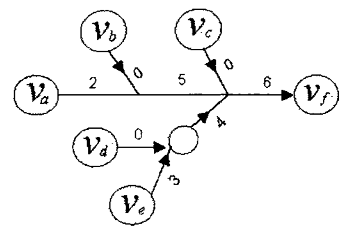

Figure 15: Tree representation

In the tree network shown in Figure 15, nodes v_b, v_c and v_d are shown with an additional link with zero propagation time to indicate that these nodes are also generating flows into the network. As for example, the flow from node v_a to v_f travels through the node v_b and node v_b is also transmitting its own flow. Thus, these two flows are contending each other at node v_b. Note that, the last contending node of all flows in tree network shown in figure 15 is node v_c. So node v_c is the *scheduling node* (v_{sc}) for this tree structure.

The equivalent star network of Figure 15 can be built by considering the propagation time from sources to (v_{sc}) as shown in Figure 16.

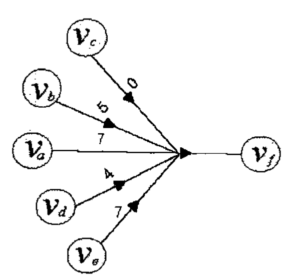

Figure 16: Star representation

6.3 Valid Relaxation

Relaxations, that we made in this thesis are discussed in detail in Section 2.7.1. In a classical OBS network, a control header traverses the intermediate switches over the path to set up the switches for an upcoming burst. The burst is held at the edge node for a offset time period and leave the edge node immediately after that time without any confirmation of switch setup. Contention would not occur in that higher rate if reservation could be confirmed before the burst leaves the edge node. That is the rationality behind our idea of relaxing some constraints of classical OBS networks in such a way that overall information of available time slots of all links are available at all edge nodes at the time of scheduling a burst.

Another relaxation that we have considered in our experiments is the removal of offset time. The rationality of this relaxation is quite similar to above explanation. Also, once all the information are available at the edge node and reservations are made before the burst launch, offset time is not necessary anymore.

6.4 Tree Coverage

This section explains the definition of the tree representation in a mesh network topology. We built a set of tree for all destination node v_d of the network to represent the entire network. Each tree is rooted at a destination node v_d so that all remaining nodes have a unique path to reach v_d and can generate traffic destined to v_d.

In [37], Phùng described the streamline effect in an OBS network and introduce a more accurate model to compute loss probability for each link in the network considering the load applied to it. If we designate this loss probability as link cost and apply a shortest path algorithm, we would have a set of paths for all source-destination pair of a network. But this formulation does not guarantee to form a set of tree from that set of paths. Rest of this section demonstrate two different scenarios with unlikely traffic pattern use this formulation. The first instance always results a set of trees whereas the later one does not.

Consider a sample network topology as shown in Figure 17. Node v_a, v_b, v_c and v_d are generating traffic destined to node v_e. In this example we will consider the traffic is static i.e. all the connection requests in the network and the amount of traffic for each connection are known at the time of computing loss probability. We consider the load for the connections in such a way that loss probabilities of the links will be similar as shown in Figure 17. Loss probabilities of most important links $v_g \rightarrow v_h$, $v_g \rightarrow v_e$ and $v_h \rightarrow v_e$ are pictured in Figure 17.

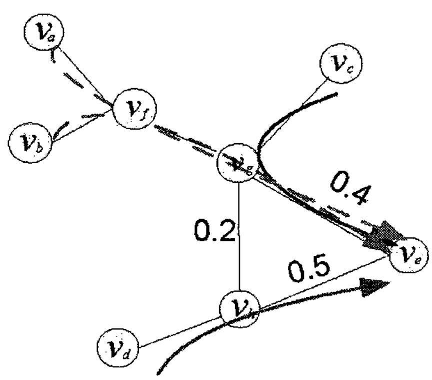

Figure 17: Build a tree structure

Note that, connections $v_a \rightarrow v_e$, $v_b \rightarrow v_e$ and $v_c \rightarrow v_e$ must reach the node v_g to reach the destination v_e. After that point these connections may have two alternative paths to choose. Similarly, connection $v_d \rightarrow v_e$ has to reach v_h first to reach v_e and then it also has two alternates. Therefore links $v_g \rightarrow v_h$, $v_g \rightarrow v_e$ and $v_h \rightarrow v_e$ have the key role in selection of routes for the connections.

We use the shortest path algorithm using loss probabilities as the cost of the links to find the routes for the connection requests. Between alternating routes, this algorithm will always select the path that is least expensive. In this particular example, connections $v_a \rightarrow v_e$, $v_b \rightarrow v_e$ and $v_c \rightarrow v_e$ must reach at node v_g with no alternating route. After that point, they have two alternative paths $v_g \rightarrow v_h \rightarrow v_e$ and $v_g \rightarrow v_e$. Routing algorithm will select the path $v_g \rightarrow v_e$ for all those three connections as it is less expensive than the other one. There is no possibility to select one alternate route for one connection and other route for rest of the connections.

With this observation, we can conclude that these connections will always build a tree rooted at destination node regardless of load variance. This statement is true for every similar scenario in any network topology under the assumptions we have in this example. So, routing with static traffic using Phùng's formula [37] will always form a tree where the root of the tree is the destination node of the connections.

We consider the same network topology and number of connection requests to build another example that shows the set of paths for those connections do not form tree structure. In this example we consider dynamic traffic, i.e., all the connection requests are not ready at the beginning and the sequence of their arrival are not known.

Figure 18(a) shows the snapshot of the network at a particular point of time. At that time, two connections $v_c \rightarrow v_e$ and $v_d \rightarrow v_e$ exist in the network and their paths to destination node v_e are selected. Figure 18(a) also includes the loss probabilities of three key links in the network that plays major role in route selection. These loss probabilities

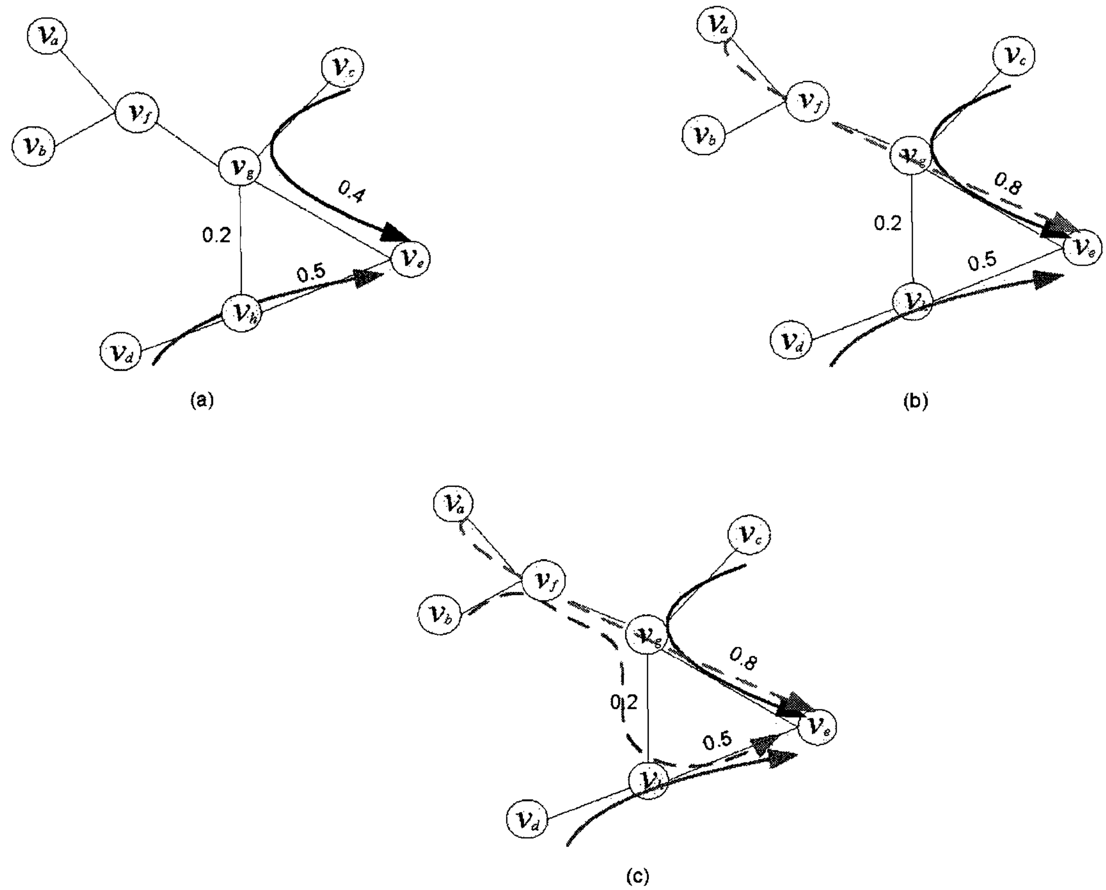

Figure 18: Building a structure which is not a tree

are computed using Phùng's formula [37] considering a certain amount of load for existing connections.

At this time, a new connection request arrives in the network requesting to establish a route from node $v_a \rightarrow v_e$. Now, the routing algorithm will select the route $v_a \rightarrow v_f \rightarrow v_g \rightarrow v_e$ for this new connection considering the loss probability scenario of the network as shown in Figure 18(b). Meanwhile, loss probabilities of the links will be updated according to the load of the new connection established. Figure 18(b) shows the updated loss probabilities of the links.

After that, consider another connection request arrives from node v_b. This connection also requests to establish a route to destination node v_e. Routing algorithm will select the route $v_b \rightarrow v_f \rightarrow v_g \rightarrow v_h \rightarrow v_e$ according to the updated loss probability status of the network as shown in figure 18(c).

By establishing the second connection, existing set of paths in the network no longer form a tree.

6.5 Selfish Tree Generation (ST)

Generating trees from a typical mesh network topology was a critical issue in investigating the performance of an OBS network with our new strategy. To observe the throughput of an OBS network with this strategy, any mesh network should be decomposed to a set of trees. Each member tree of that set will be rooted with a destination node. In this section we explain a strategy that we examined to generate that set of trees by using a meta heuristic. Title of this section implies the general concept behind the heuristic. Each tree was generated considering the best possible options but the impact of existence of other trees in the network.

We have implemented a Tabu Search based meta-heuristic to decompose the trees with the objective to minimize the weight of the network by minimizing the weight of every single tree where the weight of a given tree T is $W_T = \sum_{\ell \in L_T} W_T(\ell) + W \times \max_{\ell \in L_T} W_T(\ell)$. The detail algorithm of this heuristic is given in Section 6.7.

Minimum Spanning Tree (MST) algorithm has been applied to acquire a set of trees as initial solution for the heuristic. Number of trees in this set depend on the number of destinations in the network. Each tree of the initial solution rooted with a destination node.

Let us assume that the transport capacity of a given link ℓ is TC_ℓ and L_T denotes the set of links that are used in a given tree T. Number of incoming links that use the link ℓ to transmit bursts in T is denoted by N_m^T. f_{in}^T denotes the amount of traffic flow in each incoming link that intend to use ℓ in T. Standard deviation of f_{in}^T among the incoming links is referred by σ. Weight of the tree T is denoted by W_T and defined as:

$$\mathrm{W}_T = \sum_{\ell \in \mathrm{L}_T} \mathrm{W}_T(\ell) + W \times \max_{\ell \in \mathrm{L}_T} \mathrm{W}_T(\ell) \tag{13}$$

where

$$\mathrm{W}_T(\ell) = \frac{\sum (f_{in}^T) \times ((N_m^T)^2 - 1) \times (1 + 50 \times \max\{0, \sum (f_{in}^T) - Th\})}{TC_\ell \times (\sigma + \epsilon)}. \tag{14}$$

We examine all potential links in the set $\{L \cap L_T\}$ in a round robin fashion to minimize the weight of the tree. A link is considered as a potential link if it does not make a loop in the tree and also does not disconnect the tree. We select a potential link and recompute the weight of the tree. If this new weight is smaller than the current weight, we accept the move and modify the tree accordingly. That link was kept in the tabu list for a certain period of time. We make as many moves as possible to minimize the weight of the tree, and thus minimize the possibility of dropping a burst.

Equation (14) is composed of three major components: offered load, number of merging and penalty for overloading.

Offered load

We consider the amount of load offered on a given link and the variance of load among the incoming flows. We prefer to build a larger traffic stream taking into account the streamline effect [37]. This consideration is represented in Equation 14 by the terms $\sum(f_{in}^{T})$ and σ. The following example illustrates the effectiveness of the terms in weight computation.

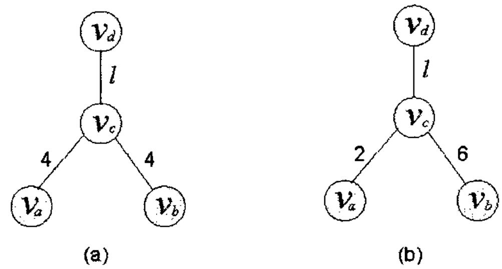

Figure 19: Effect of offered loads and their variances

Let us consider the two tree instances of Figure 19 (a) and (b). The amount of load offered by each incoming link is cited beside the links in the Figure. We need to compute the weight of link ℓ. We assume that the capacity of each link is 10 Gbps in both instances. We build this example in such a way that the amount of offered load and the number of merging operations are equal in both instances. The only difference between Figure 19 (a) and (b) is the variance of load among the flows. In Figure 19 (a), connections $v_a \rightarrow v_d$ and $v_b \rightarrow v_d$ offer equal amount of load however, in Figure 19 (b), one connection offers higher load than another. If we compute weight of link ℓ using Equation 14, weight of link ℓ becomes 24 for the configuration of Figure 19 (a) and 8.48×10^{-6} for Figure 19 (b). Since, the objective is to minimize the weight, Figure 19 (b) appears as a better solution compare to Figure 19 (a) which supports our perception.

Number of merging flows

Number of merging flows in a tree is another important component to compute the weight of a given link. It is represented in equation (14) by the term N_m^T. We build an example as shown in Figure 20 that describe the effect of number of merging operations in a tree.

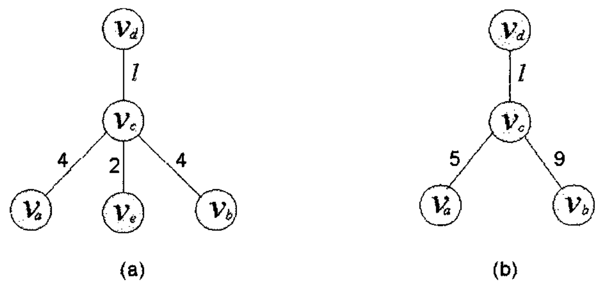

Figure 20: Effect of the number of merging flows

Let us assume that the link capacity is 10 Gbps in both instances illustrated in Figure 20. The example is built in such a way that the amount of offered load and the variances among the incoming flows are equal in both instances. The only difference between Figure 20 (a) and (b) is the number of merging operations in the trees. There are three merging flows in Figure 20 (a), however, the configuration in Figure 20 (b) features two merging flows. Equation (14) results the weight of link ℓ is 5.65×10^{-6} for Figure 20 (a) and 2.12×10^{-6} for Figure 20 (b) which also defend our perception.

Penalty factor

Penalty factor is the third major component in equation (14) and is represented by the term $(1 + 50 \times \max\{0, \sum(f_{in}^T) - Th\})$. We introduce this component in equation (14) to discourage any link to be overloaded in a tree. once a link is requested to carry traffic beyond a given threshold value, an amount of penalty is imposed on the weight of the link. Thus, the weight of the link will be increased and will not favored by the algorithm. Figure 21 illustrates an example that explains the effect of penalty factor in weight computation.

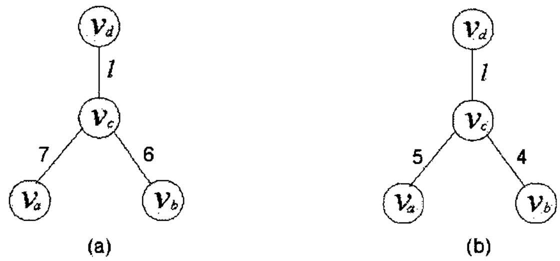

Figure 21: Penalty factor

Let us consider the two instances of the simple tree in Figure 21 (a) and (b). Let us assume the capacity of link ℓ is 10 Gbps in both instances and the threshold value is 0.9, i.e., the penalty amount will be imposed if the offered load on link ℓ is larger than 90% of its capacity. The example is built in such a way that the number of merging flows and the variance of traffic flows between the incoming links are equal in both instances. The offered load on link ℓ is higher than its capacity in Figure 21 (a), however, the load in Figure 21 (b) is equal to the given threshold value. Equation (14) outcomes the weight of link ℓ is 1.23×10^{-4} for Figure 21 (a) and 3.85×10^{-6} for Figure 21 (b). Since, the objective is to minimize the weight of a given tree, the algorithm will select the instance in Figure 21 (b) over Figure 21 (a) which support our perception.

6.6 Collaborative Tree Generation (CT)

This section explains the revised strategy of section 6.5 to decompose a mesh network topology into tree structure. In this strategy, trees were decomposed in a collaborative manner. Each tree was generated and modified using best possible option considering the existence of other trees in the network.

This is also a meta-heuristic likely the one explained in section 6.5 with taking into account an important component that seems to have an major impact on network performance. The objective function here we have considered is to minimize the weight of the network W_N instead of weight of a single tree. The detail algorithm of this heuristic is given in 6.7. Like the "Selfish Tree Generation" explained in section 6.5, here we also have applied MST to find a set of trees as initial solution. We also have considered the number of merging input links and load of the link as two major components to define W_N.

In every iteration, we find the link ℓ_b in the network that carries the maximum load, i.e., $load(\ell_b) = \max_{\ell \in L} load(\ell)$. Then, we select the tree t_b such that $load_{t_b}(\ell_b) = \max_{t \in T} load_t(\ell_b)$ i.e. the tree that uses the link ℓ_b and introduces maximum load on ℓ_b. Once we have a particular tree, we investigate the potentiality of the links from the set of unused links $L^{t_b}_{u'}$ such that $L^{t_b}_{u'} = L \cap L^{t_b}_{u}$ to add to the tree. We build a temporary tree using that potential link and reevaluate the weight of the network W_N^{temp} and compare it with current weight W_N. We make the change of the link permanent if the new weight minimizes the total weight of the network.

Let us assume that the transport capacity of a given link ℓ is TC_ℓ. Number of incoming links that use the link ℓ to transmit bursts is denoted by N_m. f_{in} denotes the amount of traffic flow in each incoming link that intend to use ℓ. Standard deviation of f_{in} among the incoming links is referred by σ. Weight of the network W_N is defined as follows:

$$\mathrm{W}_N = \sum_{\ell \in L} W_N(\ell) + W \times \max_{\ell \in L} W_N(\ell) \tag{15}$$

where

$$\mathrm{W}_N(\ell) = \frac{\sum f_{in} \times (N_m^2 - 1) \times (1 + 50 \times \max\{0, \sum f_{in} - Th\})}{TC_\ell \times (\sigma + \epsilon)}. \tag{16}$$

The motivation behind defining the formulation and introducing the penalty factor are similar as in Section 6.5.

6.7 Detailed Algorithm

We have developed a Tabu Search based meta heuristic that provides a set of optimized trees in the network for all destination nodes in the network. The detailed algorithm of our Tabu Search is given in this section. Our algorithm read a minimum spanning tree as input for any network topology and interpret that as a set of trees for all destinations in the network. This input set of tree is considered as initial solution for the heuristic.

We consider an OBS network represented by a directed graph $G = (V, L)$ where V is associated with the set of nodes and L corresponds to the set of (directed) links between the nodes. The set of nodes is divided into the set V_{ACCESS} of access nodes and the set V_{CORE} of core nodes. Connection requests are received between pair of access nodes. Consider T as the set of trees in the network and W_N as the weight of the network. There are as many trees as the number of destination nodes in the network.

For each pair of source and destination $(v_s, v_d) \in V_{\text{ACCESS}} \times V_{\text{ACCESS}}$, let

$\mathcal{P}_{sd}$ set of routes from v_s to v_d.

For each link $\ell \in L$, let

$load(\ell)$ be the traffic load on ℓ,

s_ℓ be the source node,

d_ℓ be the destination node.

Some other notation that we have used in our algorithm are,

ℓ_b bottleneck link of the network,

$v_s^{l_b}$ be the source node of ℓ_b,

t_b be the tree that provides maximum load on ℓ_b,

L^{t_b} be the set of links used in t_b,

n_c be the current node considering in any single step,

W_N^{curr} be the network weight in a single step,

W_N^{best} be the best network weight among the neighbors,

W_N^{incumb} be the incumbent network weight,

τ be the Tabu list,

L_p be the set of potential links.

Step 1 Initialization

{Initialize the parameters}
for all $(v_s, v_d) \in V_{\text{ACCESS}} \times V_{\text{ACCESS}}$ **do**
  Compute $\mathcal{P}_{sd}$ from minimum spanning tree;
end for.
{Initialize the data structures}
for all $\ell \in L$ **do**
  Compute offered load $load(\ell)$ on ℓ;
end for
Initialize the Tabu list τ;
Initialize the $minHeap$;

Step 2 Current Step

Find link ℓ_b such that $load(\ell_b) = \max_{\ell \in L} load(\ell)$;
Find tree t_b such that $load_{t_b}(\ell_b) = \max_{t \in T} load_t(\ell_b)$;
Find $\bar{L}^{t_b}$ such that $\bar{L}^{t_b} = L \backslash L^{t_b}$;
Find the subtree T_s such that the root of T_s is $v_s^{l_b}$
for all $v \in T_s$ **do**
  Find the set of potential links L_p;
  for all $\ell_p \in L_p$ **do**
    if $\ell_p \in \bar{L}^{t_b}$ **then**
      if $(\ell_p, t_b) \in \tau$ **then**
        if ℓ_p does not disconnect t_b **then**
          Remove ℓ_b from t_b to disconnect the tree;
          Add ℓ_p to t_b to reconnect the tree;
          Compute network weight W_N^{curr};
          Insert ℓ_p and W_N^{curr} in $minHeap$;
        end if
      end if
    end if
  end for
end for
Find the best weight W_N^{best} and corresponding tree t_b^{best} from the $minHeap$ and ACCEPT the move;
$\tau \leftarrow \tau \cup (l_p, t_b)$;
if $W_N^{best} < W_N^{incumb}$ **then**
  $W_N^{incumb} = W_N^{best}$;
  $t^{incumb} = t^{best}$;
end if

Algorithm CT (Collaborative Tree Generation)

Step 2 of this algorithm repeats until the incumbent value does not improve for a certain number of iteration.

6.8 ST vs. CT

In this section, we analyze the quality of the algorithms ST and CT and also compare the simulation results of those algorithms. Figure 22 shows the load distribution over the links in NSF network once ST decompose the trees for all sources. Load distribution over the links after decomposing the trees by CT are shown in Figure 23.

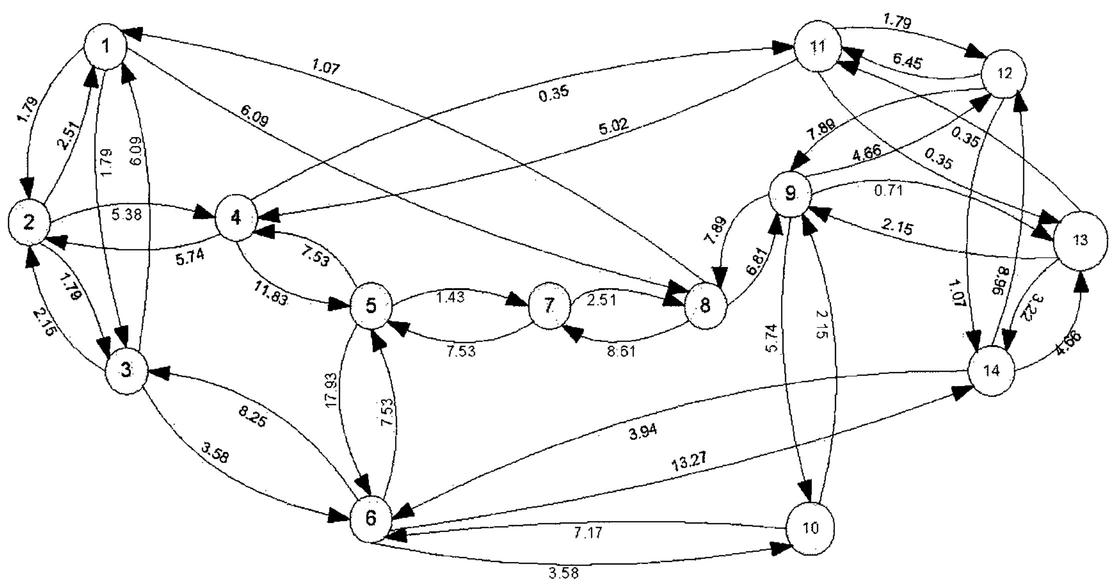

Figure 22: Link loads with ST approach

We also present an instance of tree rooted at destination 1 in Figures 24 and 25 generated by ST and CT respectively. To compare the quality of the trees in Figures 24 and 25, we observe that the number of merging flows are reduced by CT in Figure 25.

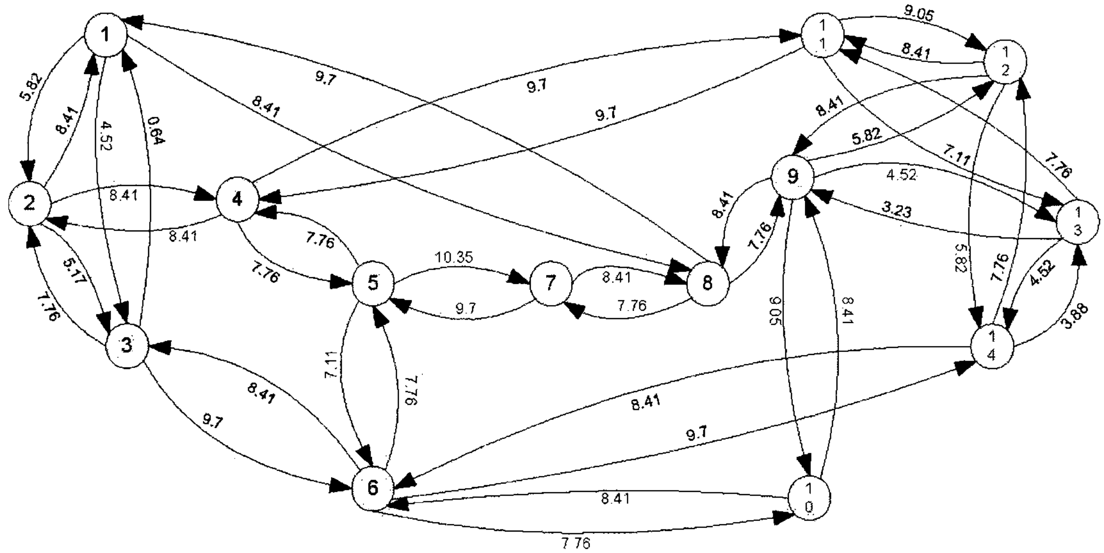

Figure 23: Link loads with CT approach

Table 7: Comparison of ST vs CT

Algorithms	**Load variances** (σ_l)	**Number of merging**
ST	24.8	77
CT	15.1	61

Table 7 provides the values of some evaluating parameters of the trees for both ST and CT. It explains that, CT reduces the overall number of merging flows in the network as well as reduces the variances of link load, i.e., CT distributes the load over the network in a more efficient manner than ST.

Table 8 provides the experimental results of ST and CT for EON2004 topology. Simulation results with NSF topology are presented in Table 9. We observe that CT increases the throughput about 18% in EON2004 and about 44% in NSF network with nearly similar drop rate. Though CT provides a small increase in drop rate in EON2004, we can disregard that based upon the sensitivity analysis of drop rate in Section 7.2.

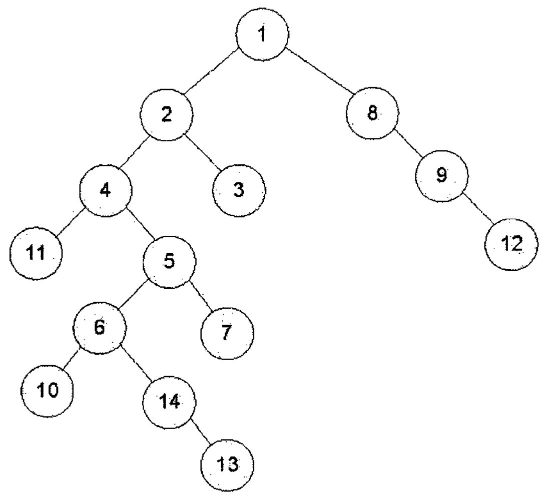

Figure 24: A tree instance generated by ST

Table 8: Comparison of throughputs for ST & CT in EON

EON with Uniform traffic		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
ST	86.9	0.9
CT	102.7	1.1

Table 9: Comparison of throughputs for ST & CT in NSF

NSF with Uniform traffic		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
ST	83.6	0.9
CT	120.8	0.8

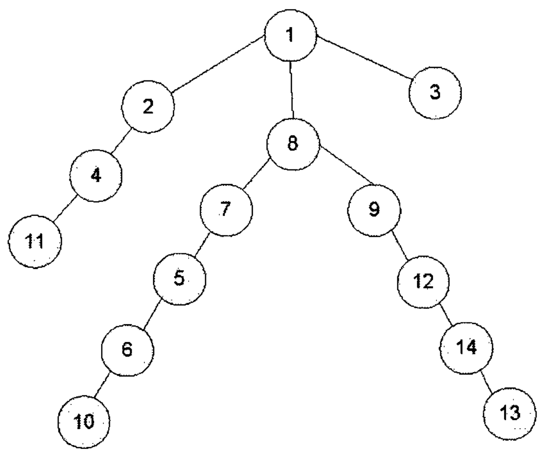

Figure 25: A tree instance generated by CT

Chapter 7

Experimental results

In this chapter we compare the performances of the proposed algorithms in the three previous chapters, with some of the most efficient ones of the literature and analyze the comparative results. We compare the network throughputs achieved by different algorithms under different traffic scenarios. Network topologies and traffic scenarios that are used in our experiments are explained in Section 4.6.

7.1 Solution Characteristics

Different sets of parameters perform important roles in different algorithms. Table 10 provides a comparative study among the values of various parameters with different algorithms. *Number of merging* illustrates the cumulative sum of merging operations performed by each connection in the network. *Average E2E delay* represents the average end to end delay experienced by the transmitted bursts. Variances of load on the links are computed as $\sum_{\ell} (LOAD_{\ell} - LOAD_{\text{mean}})^2$. The last parameter explains the structure formed by the set of paths for a given destination. Since WSP rely on multiple paths for a given s-d pair and selection of path varies dynamically, it is not possible to select a path in advance for a given burst. That is why, we could not compute load variance and number of merging for WSP.

Table 10: Comparison of optimization criteria

Algorithms	**Load variances** (σ_l)	**Number of merging**	**Average E2E Delay**	**Is it always a tree?**
LBH	14.2	88	5546.4	N
CT	15.1	61	5605.3	Y
Streamline-Qian [4]	16.8	96	5538.8	N
WSP	-	-	4312.4	Y

We observe that, variance of loads per link is less with LBH over other algorithms, i.e., LBH distributes the loads over the network most effectively. CT generates the trees in such a way that number of merging operations in the network is minimum comparing with other approaches. WSP experiences least E2E delay over other algorithms. Structures formed by the set of paths for a given destination with LBH and *streamline-Qian* algorithms are not always a tree. However, we observe that about 75% instances shape a tree.

7.2 Results and Analysis

We demonstrate and compare the simulation results on the EON, NSF and NY topologies.

EON

Table 11 shows the simulation results for EON with uniform traffic. Tables 12 and 13 show the results for non-uniform traffic with 10% and 50% variances in offered load in the source nodes respectively.

Table 11: Comparison of throughputs for EON with loss rate sensitivity

EON with Uniform traffic				
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)	**Throughput** (Gbps)	**Drop rate** (%)
LBH	140.5	0.9	142.4	1.1
CT	101.9	0.9	104.5	1.1
Streamline-Qian [4]	139.2	0.9	141.4	1.1
WSP	160.6	0.9	163.3	1.1

Table 12: Comparison of results for EON with non-uniform traffic (10%)

EON with Non-Uniform traffic *load variance 10%*				
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)	**Throughput** (Gbps)	**Drop rate** (%)
LBH	139.2	0.8	142.1	1.1
CT	101.6	0.9	104.3	1.1
Streamline-Qian [4]	137.3	1.0	141.1	1.1
WSP	161.1	0.9	162.6	1.1

We enforce the equal amount of loads to examine the performances of the algorithms and the simulation results are shown in Table 14.

NSF

We execute a similar set of simulations on the NSF topology. Table 15 shows the experimental results with uniform traffic and Tables 16 and 17 present results with non-uniform traffic.

NY

Similar set of simulation results like EON and NSF for NY network are presented in Tables 18 to 20.

Table 13: Comparison of results for EON with non-uniform traffic (50%)

EON with Non-Uniform traffic *load variance 50%*				
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)	**Throughput** (Gbps)	**Drop rate** (%)
LBH	143.0	0.9	145.5	1.1
CT	102.6	0.9	107.4	1.2
Streamline-Qian [4]	140.4	0.9	143.1	1.2
WSP	164.2	0.8	166.1	0.9

Table 14: Comparison of results with equal load for EON

EON with Uniform traffic and equal load		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
LBH	157.2	2.5
CT	133.8	9.6
Streamline-Qian [4]	157.1	2.5
WSP	160.6	0.9

Analysis

We observe that, WSP provides better performance in EON and NY with every traffic instances. However, it does not perform well with NSF. A smaller connectivity for the NSF network may cause this behavior. Since, WSP relies on the alternating routes for transmitting bursts, if the alternate routes are not well separated from each other, WSP may not perform well. Due to the connectivity of NSF network, the degree of separation among alternating routes is not advantageous.

We measure the connectivity and the degree of separation among alternative routes for a given s-d pair to support our claim. We focus on the differences of lengths among the alternating routes to measure the connectivity. Deviation of lengths among the alternating

Table 15: Comparison of results for NSF with uniform traffic

NSF with Uniform traffic		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
LBH	169.5	0.9
CT	120.8	0.8
Streamline-Qian [4]	168.5	1.0
WSP	113.1	0.9

Table 16: Comparison of results for NSF with non-uniform traffic (10%)

NSF with Non-Uniform traffic *load variance 10%*		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
LBH	175.1	0.9
CT	119.8	0.7
Streamline-Qian [4]	169.6	0.9
WSP	113.1	1.1

paths are inversely related to the connectivity of the network. In a largely connected network, the differences of lengths between a given shortest path and the next one of different length, for a given *s-d* pair, are smaller over the differences in a less connected network. To measure the connectivity of NSF, EON and NY topologies, we investigate the selected routes in detail. We have selected three different connections in each network arbitrarily and examined the three alternating routes that are selected by the routing algorithm for every connection. We compute the average length in terms of the number of hops and the average degree of overlapping links of three alternating paths. Table 21 shows the average length of the shortest path, first alternate and second alternate paths of the three given connections. Average degree of overlapping links among the alternating paths are presented

Table 17: Comparison of results for NSF with non-uniform traffic (50%)

NSF with Non-Uniform traffic *load variance 50%*		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
LBH	176.8	1.4
CT	124.5	0.8
Streamline-Qian [4]	169.9	2.0
WSP	117.3	1.6

Table 18: Comparison of results for NY with uniform traffic

NY with Uniform traffic		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
LBH	436.6	1.0
CT	155.7	1.1
Streamline-Qian [4]	337.6	1.1
WSP	439.3	0.9

in Table 22.

The data presented in Table 21 interpret that, the differences of alternating paths in NSF are larger over the differences in EON and NY which illustrates the greater connectivity of NY network over NSF and EON. Data in Table 22 illustrate that the alternating routes are most separated in NY over NSF and EON.

Other than WSP, LBH performs reasonably better than Qian's algorithm and had a clear dominance over the other algorithms under every simulation scenario. However, LBH experiences moderately longer average E2E delay than Qian's algorithm. Though CT performs better than WSP in NSF network, it provides a smaller throughput with EON and NY networks, which was not expected. The computation of weight of the network for a given

Table 19: Comparison of results for NY with non-uniform traffic (10%)

NY with Non-Uniform traffic		
load variance 10%		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
LBH	440.7	0.8
CT	158.6	0.3
Streamline-Qian [4]	343.5	0.9
WSP	443.1	0.9

Table 20: Comparison of results for NY with non-uniform traffic (50%)

NY with Non-Uniform traffic		
load variance 50%		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
LBH	447.4	1.0
CT	162.1	0.1
Streamline-Qian [4]	355.2	1.2
WSP	448.6	1.0

configuration may cause this unexpected behavior. The accurate measurement of weight of the network is very important to select a specific routing configuration. The formulation that we have used to measure the weight of a network has three major components: One is to consider the incoming load, the second one takes into account the number of merging and the third one is a penalty factor that was used to avoid a certain link to be overloaded. It is important to keep the value of these three components in same degree. Otherwise, one of the component could obtain unexpected precedence over other components and might cause unlikely results. We believe that the performance of CT could be improved by appropriate adjustments of the three major components.

Table 21: Comparison of average lengths

	Average number of hops		
Topology	**Shortest path**	**First alternate path**	**Second alternate path**
NSF	2	4.33	5.7
EON	2	3.3	3.3
NY	2	2	2.7

Table 22: Comparison of overlapping degree

Topology	**Average degree of overlapping links**
NSF	3
EON	1.3
NY	1

7.3 Estimation of the Quality of the Upper Bound

Table 23 presents the comparison of upper bound on the throughput with lower bounds. Two values for lower bound are presented in Table 23. One of them shows the lower bound with basic OBS setup and the another one represents the performance of an enhanced architecture of OBS proposed by Coutelen *et al.* [9].

Table 23: Quality estimation of upper bound

EON with Uniform traffic		
Algorithms	**Throughput** (Gbps)	**Drop rate** (%)
WSP	160.6	0.9
Translucent architecture	148.8	1.1
Basic OBS	4.1	1.0

The gap between the upper bound and the lower bound from basic OBS is considerably large, however it is the first ever estimation of lower bound. Though several improvements are possible for the basic lower bound [28, 29, 30, 37], the most recent one includes the proposal of translucent architecture by Coutelen *et al.* in [9]. The translucent architecture provides a tighter lower bound compared to the basic one and very much closer to the upper bound. Obviously, the presented upper bound can also be improved by using an efficient scheduling algorithm.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

Recently OBS has received much attention due to the increasing demand of bandwidth in communication networks. OBS was proposed to overcome the limitations of OCS in the context of inefficient bandwidth utilization. In this thesis, we investigated several approaches to determine the maximum throughput achievable by OBS with some valid relaxations such as removing offset time.

First, we studied a strategy that used k-shortest paths for all s-d pairs in the network to transmit bursts. We investigated two algorithms to select a path among the k-shortest paths: SSP and WSP. SSP employs a sequential selection process while WSP introduces a weight to each path under certain considerations and selects the one with minimum possible weight. Secondly, a heuristic called LBH was developed to investigate the load balancing approach that attempts to distribute the overall load over the network as evenly as possible. Finally, we investigated another innovative approach that we named tree decomposition. We proposed two algorithms to decompose the trees for all possible destinations in the network: ST and CT. ST builds a tree by taking into account the offered load and the number of incoming flows on a given link. CT, in addition, takes into account the existence of other trees in the network as well as the offered load and the number of incoming flows.

We presented the simulation results and compared the results among the proposed algorithms as well as the most efficient algorithms in the literature. We showed that, WSP outperforms other algorithms in EON2004 and NY networks, however, other than WSP, LBH performs reasonably better than other algorithms in all three network instances.

8.2 Future Work

At the beginning of the thesis, we split the maximization problem into two subproblems: routing and scheduling. The algorithms that we have investigated in this thesis are more likely to routing subproblem though we used a reservation process to reserve resources for bursts transmission. We believe, it is worthy to invest more focus on scheduling. An efficient scheduler may gradually increase the throughput of an OBS network. Since, the idea of tree decomposition was proposed with presumption that scheduling will be performed on a single node for each tree, we strongly believe that the performance of CT can be significantly improved by using an efficient scheduler. This area should be more investigated for further improvement of OBS performance.

Performances of OBS networks were not compared with the performances of OCS networks in this thesis. It is not always obvious to compare OBS and OCS, since, these two switching technologies are experts on two different traffic patterns. OCS is proficient with steady and stable traffic while OBS was proposed to feature better deal with bursty traffic. However, additional studies need to be conducted to define a standard manner to compare

OCS and OBS so that one of them can be choose to implement in real time to fulfill any particular needs.

Bibliography

[1] R. K. Ahuja, K. Mehlhorn, J. Orlin, and R. E. Tarjan. Faster algorithms for the shortest path problem. J. ACM, 37(2):213–223, 1990.

[2] A. Birman. Computing approximate blocking probabilities for a class of all-optical networks. In Fourteenth Annual Joint Conference of the IEEE Computer and Communications Societies., pages 651–658, vol.2, April 1995.

[3] H. Chaskar, S. Verma, and R. Ravikanth. A framework to support IP traffic over WDM using optical burst switching. Proc. of IEEE Optical Network Workshop, Jan 2000.

[4] Q. Chen, G. Mohan, and K. C. Chua. Route optimization in optical burst switched networks considering the streamline effect. Computer Networks, 52(10):2033 – 2044, 2008.

[5] Y. Chen, C. Qiao, and X. Yu. Optical burst switching: a new area in optical networking research. IEEE Network, 18(3):16–23, May-June 2004.

[6] I. Chlamtac, A. Ganz, and G. Karmi. Lightpath communications: An approach to high bandwidth optical WAN's. IEEE Transactions on Communications, 40(7):1171–1182, July 1992.

[7] T. Coutelen, G. Hebuterne, and B. Jaumard. An OBS RWA formulation for asynchronous loss-less transfer in OBS networks. In IEEE Workshop on High Performance Switching and Routing (HPSR), 2009.

[8] T. Coutelen, B. Jaumard, and G. Hebuterne. Improving RWA-OBS formulation and solution. In International Workshop on Optical Burst/Packet Switching (WOBS), September 2009.

[9] T. Coutelen, B. Jaumard, and G. Hebuterne. A translucent OBS node architecture to improve traffic emission and loss probability. In International Workshop on Optical Burst/Packet Switching (WOBS), September 2009.

[10] D. Eppstein. Finding the k shortest paths. 35th Annual Symposium on Foundations of Computer Science, pages 154–165, 1994.

[11] C.M. Gauger, M. Kohn, and J. Scharf. Comparison of contention resolution strategies in OBS network scenarios. In Proceedings of 6th International Conference on Transparent Optical Networks., volume 1, pages 18–21, July 2004.

[12] C.M. Gauger, M. Kohn, and J. Scharf. Performance of contention resolution strategies in OBS network scenarios. In 3rd International Conference on the Optical Internet,Yokohama, Japan, July 2004.

[13] A. Ge, F. Callegati, and L.S. Tamil. On optical burst switching and self-similar traffic. IEEE Communications Letters, 4(3):98–100, Mar 2000.

[14] M. Hayashitani, K. Okazaki, and N. Yamanaka. A new burst assembly technique supporting fair qos about the number of hops in OCBS multi-hop networks. In The Joint

International Conference on Optical Internet and Next Generation Network, pages 40–42, July 2006.

[15] http://en.wikipedia.org/wiki/NSFNet.

[16] http://sndlib.zib.de/home.action.

[17] G.C. Hudek and D.J. Muder. Signaling analysis for a multi-switch all-optical network. In IEEE International Conference on Communications, volume 2, pages 1206–1210 vol.2, Jun 1995.

[18] D.K. Hunter, M.C. Chia, and I. Andonovic. Buffering in optical packet switches. Journal of Lightwave Technology, 16(12):2081–2094, Dec 1998.

[19] M. Iizuka, M. Sakuta, Y. Nishino, and I. Sasase. A scheduling algorithm minimizing voids generated by arriving bursts in optical burst switched WDM network. In Global Telecommunications Conference, IEEE, volume 3, pages 2736–2740 vol.3, Nov. 2002.

[20] J. P. Jue and V. M. Vokkarane. Optical Burst Switched Networks. Springer, 2005.

[21] B. Kantarci and S. Oktug. Adaptive threshold based burst assembly in OBS networks. In Canadian Conference on Electrical and Computer Engineering, pages 1419–1422, May 2006.

[22] B. Kantarci and S. Oktug. Loss rate-based burst assembly to resolve contention in optical burst switching networks. IET Communications, 2(1):137–143, January 2008.

[23] B. Kantarci, S. Oktug, and T. Atmaca. Analyzing the effects of burst assembly in optical burst switching under self-similar traffic. In Advanced industrial conference on telecommunications/service assurance with partial and intermittent

resources conference/e-learning on telecommunications workshop, pages 109–114, July 2005.

[24] E. Karasan and E. Ayanoglu. Performance of WDM transport networks. IEEE Journal on Selected Areas in Communications, 16(7):1081–1096, Sep 1998.

[25] S. Kim, N. Kim, and M. Kang. Contention resolution for optical burst switching networks using alternative routing. In IEEE International Conference on Communications, volume 5, pages 2678–2681 vol.5, 2002.

[26] P. Klein, S. Rao, M. Rauch, and S. Subramanian. Faster shortest-path algorithms for planar graphs. In STOC '94: Proceedings of the twenty-sixth annual ACM symposium on Theory of computing, pages 27–37, New York, NY, USA, 1994. ACM.

[27] S. Lee, K. Sriram, H. Kim, and J. Song. Contention-based limited deflection routing in OBS networks. In Global Telecommunications Conference, IEEE, volume 5, pages 2633–2637 vol.5, Dec. 2003.

[28] J. Li, G. Mohan, and K.C. Chua. Dynamic load balancing in IP-over-WDM optical burst switching networks. Computer Networks, 47(3):393 – 408, 2005.

[29] J. Li, C. Qiao, J. Xu, and D. Xu. Maximizing throughput for optical burst switching networks. Twenty-third annual joint conference of the IEEE computer and communications Societies, vol. 3:1853–1863, 2004.

[30] J. Li, C. Qiao, J. Xu, and D. Xu. Maximizing throughput for optical burst switching networks. IEEE/ACM Transactions on Networking: Accepted for future publication, volume PP:1–1, 2007.

[31] LION and COST 266. Reference networks 2004, part of the european information society technologies (IST) fifth framework program.

[32] B. Mukherjee. WDM-based local lightwave networks. i. single-hop systems. IEEE Network, 6(3):12–27, May 1992.

[33] B. Mukherjee. WDM optical communication networks: progress and challenges. IEEE Journal on Selected Areas in Communications, 18(10):1810–1824, Oct 2000.

[34] B. Mukherjee. Optical WDM networks. Springer, 2006.

[35] A. M. Odlyzko. Internet traffic growth: sources and implications. In B. B. Dingel, W. Weiershausen, A. K. Dutta, and K. Sato, editors, Optical Transmission Systems and Equipment for WDM Networking II, volume 5247, pages 1–15. SPIE, 2003.

[36] S. Okamoto. Photonic transport network architecture and OA&M technologies to create large-scale robust networks. IEEE Journal on Selected Areas in Communications, 16(7):995–1007, Sep 1998.

[37] M.H. Phung, K.C. Chua, G. Mohan, M. Motani, and T.C. Wong. The streamline effect in OBS networks and its application in load balancing. 2nd International Conference on Broadband Networks, Volume 1:283–290, Oct. 2005.

[38] C. Qiao and M. Yoo. Optical burst switching (0BS)- a new paradigm for an optical internet. J. High Speed Nets., 8(1):69–84, January 1999.

[39] C. Qiao and M. Yoo. Choices, features and issues in optical burst switching. Optical Networks Magazine, 1:36–44, April 2000.

[40] R. Ramaswami and K. N. Sivarajan. Optical networks: a practical perspective. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[41] M. V. Rodrigo and J. Gotz. An analytical study of optical burst switching aggregation strategies. In First Annual International Conference on Broadband Networks, Oct. 2004.

[42] J.J.P. Rodrigues and M.M. Freire. Performance assessment of enhanced just-in-time protocol in OBS networks taking into account control packet processing and optical switch configuration times. In 2nd International Conference on Advanced Information Networking and Applications - Workshops, pages 434–439, March 2008.

[43] J.J.P.C. Rodrigues, M.M. Freire, N.M. Garcia, and P.M.N.P. Monteiro. Enhanced just-in-time: A new resource reservation protocol for optical burst switching networks. In 12th IEEE Symposium on Computers and Communications, pages 121–126, July 2007.

[44] Z. Rosberg, H. L. Vu, M. Zukerman, and J. White. Blocking probabilities of optical burst switching networks based on reduced load fixed point approximations. In Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies., volume 3, pages 2008–2018, March-3 April 2003.

[45] L. Tancevski, G. Castanon, F. Callega, and L. Tamil. Performance of an optical IP router using non-degenerate buffers. In Global Telecommunications Conference, volume 2, pages 1454–1459 vol.2, 1999.

[46] J. Teng and G.N. Rouskas. Routing path optimization in optical burst switched networks. In Conference on Optical Network Design and Modeling, pages 1–10, 7-9, 2005.

[47] J. Turner. Terabit burst switching. J. High Speed Nets., 8(1):pp. 3–16, January 1999.

[48] S. Verma, H. Chaskar, and R. Ravikanth. Optical burst switching: a viable solution for terabit IP backbone. IEEE Network, 14(6):48–53, Nov/Dec 2000.

[49] V.M. Vokkarane, J.P. Jue, and S. Sitaraman. Burst segmentation: an approach for reducing packet loss in optical burst switched networks. In IEEE International Conference on Communications, volume 5, pages 2673–2677, May 2002.

[50] J. Wei, J. Pastor, R. Ramamurthy, and Y. Tsai. Just-in-time optical burst switching for multiwavelength networks. In IFIP Broadband Communications, (Hong Kong), pages 339–352, November 1999.

[51] J.Y. Wei and Jr. McFarland, R.I. Just-in-time signaling for WDM optical burst switching networks. Journal of Lightwave Technology, 18(12):2019–2037, Dec 2000.

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

[53] M. Yoo and C. Qiao. Just-enough-time (jet): a high speed protocol for bursty traffic in optical networks. In Vertical-Cavity Lasers, Technologies for a Global Information Infrastructure, WDM Components Technology, Advanced Semiconductor Lasers and Applications, Gallium Nitride Materials, Processing, and Devi, pages 26–27, Aug 1997.

[54] M. Yoo and C. Qiao. New optical burst-switching protocol for supporting quality of service. In J. M. Senior and C. Qiao, editors, All-Optical Networking: Architecture, Control, and Management Issues, volume 3531, pages 396–405. SPIE, 1998.

[55] M. Yoo, C. Qiao, and S. Dixit. Comparative study of contention resolution policies in optical burst-switched WDM networks. In J. M. Senior, S. Dixit, and C. Qiao, editors,

Terabit Optical Networking: Architecture, Control, and Management Issues, volume 4213, pages 124–135. SPIE, 2000.

[56] M. Yoo, C. Qiao, and S. Dixit. Qos performance of optical burst switching in IP-over-WDM networks. IEEE Journal on Selected Areas in Communications, 18(10):2062–2071, Oct 2000.

[57] X. Yu, J. Li, X. Cao, Y. Chen, and C. Qiao. Traffic statistics and performance evaluation in optical burst switched networks. J. Lightwave Technol., 22(12):2722, 2004.

[58] A. Zalesky, H.L. Vu, Z. Rosberg, E.W.M. Wong, and M. Zukerman. OBS contention resolution performance. Performance Evaluation, 64(4):357 – 373, 2007.

[59] K. Zhu, H. Zhu, and B. Mukherjee. Traffic Grooming in Optical WDM Mesh Networks. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.