Architectures of electro-optical packet switched networks

Berger, Michael Stubert

Publication date:
2004

Document Version
Publisher's PDF, also known as Version of record

Architectures of Electro-Optical Packet Switched Networks

Ph.D. thesis

Michael S. Berger

Research Center COM
Technical University of Denmark

February 2004

Supervisor
Lars Dittmann

Table of Contents

Abstract

This thesis focuses on network- and node architectures for electrical and optical packet switched networks. Future packet switched networks could evolve towards many small, distributed units or towards fewer large, centralised switch units. This work assumes the latter evolution scenario and examines possible architectures for future high capacity networks with high capacity nodes. It is assumed that optics will play a key role in this scenario, and in this respect, the European IST research project DAVID aimed at proposing viable architectures for optical packet switching, exploiting the best from optics and electronics.

An overview of the DAVID network architecture is given, focusing on the MAN and WAN architecture as well as the MPLS based network hierarchy. A statistical model of the optical slot generation process is presented and utilised to evaluate delay vs. efficiency. Furthermore, a benchmarking study has been carried out to compare power consumption of electrical and optical packet switches.

The basic principles for Traffic Engineering and Quality of Service provisioning are discussed, and a simple scheme for Traffic Engineering in a best effort TCP/IP based network is proposed. Also, Constraint Based Routing is examined, and the effect from taking the link load into account is evaluated.

It is believed that electrical packet switching will satisfy demands in the coming years, and this work covers several aspects hereof. A new load-balancing scheme for multipath packet switches is proposed where packets are collected and transmitted over identically parallel switch planes, eliminating the need for a packet re-ordering mechanism. An analytical result for the worst-case delay is derived, and the average performance is evaluated by a simulation study.

Moreover, a new and more scalable architecture for a buffered crossbar switch is presented. The architecture uses two levels of backpressure (flow control) with different constraints on round trip time. No additional scheduling complexity is introduced, and for the actual example shown, a reduction in memory of 75 % was obtained at the cost of an additional speedup of 10 %.

Lastly, the address lookup and classification problem is addressed, and an IP lookup algorithm with low memory requirement and fast updates is presented. The scheme uses a combination of trie and tree search, which is efficient in memory usage because the structure contains exactly one node for each prefix.

Resumé

Denne afhandling fokuserer på netværks- og knudepunktsarkitekturer for elektriske og optiske pakkekoblede netværk. Fremtidens pakkekoblede netværk kunne udvikle sig mod mange små, distribuerede enheder eller mod færre store, centraliserede enheder. Dette arbejde antager det sidste udviklingsscenario og undersøger mulige arkitekturer for fremtidige høj kapacitetsnetværk med højkapacitetsknudepunkter. Det antages, at optik vil spille en nøglerolle i dette scenario, og i den forbindelse har det europæiske forskningsprojekt DAVID sigtet mod at foreslå gennemførlige arkitekturer for optisk pakkekobling ved at udnytte det bedste fra optik og elektronik.

Der gives et overblik over DAVID arkitekturen, som fokuserer på arkitekturen af MAN og WAN netværket såvel som det MPLS baserede netværkshierarki. En statistisk model af genereringsprocessen for optiske pakker bliver præsenteret og udnyttet til at evaluere forsinkelse versus effektivitet. Derudover er der udført et sammenligningsstudie for effektforbrug af elektriske og optiske pakkeswitche.

De basale principper for Traffic Engineering og Quality of Service diskuteres, og en simpel måde til at foretage traffic engineering i et TCP/IP baseret netwærk bliver foreslået. Desuden undersøges Constraint Based Routing og effekten af at lade forbindelsesbelastningen indgå.

Det antages, at elektrisk pakkekobling vil kunne tilfredsstille behovene i de kommende år, og dette arbejde behandler flere aspekter heraf. En ny load-balanceringsmetode er foreslået, hvor pakker samles og transmitteres over identiske parallelle switch-planer, hvorved man undgår en pakkegenordningsmekanisme. Et analytisk resultat for den værst tænkelige forsinkelse er udledt, og den gennemsnitlige forsinkelse er evalueret i et simulationsstudie.

Derudover er en ny og mere skalerbar arkitektur for en buffereret crossbar switch præsenteret. Arkitekturen benytter to niveauer af backpressure med forskellige begrænsninger på forsinkelsen. Der introduceres ikke ekstra kompleksitet i skeduleringsprocessen, og for det viste eksempel kunne der opnås en reduktion i hukommelsesforbruget på 75 % på bekostning af ekstra speedup på 10 %.

Sidst addreseres adresseopslags- og klassificeringsproblemet, og en IP opslagsalgoritme med lavt hukommelsesforbrug og hurtig opdatering præsenteres. Metoden anvender en kombination af trie og træ søgning, som er effektiv i hukommelsforbrug, fordi strukturen indeholder netop en knude for hver prefix adresse.

Acknowledgements

I would like to thank Lars Dittmann for supervising my PhD, but also for giving me the opportunity to work in a unique research group. Also, I would like to thank all the people in networking group for making the stay at COM a good experience.

In particular, I owe great thanks to Villy B. Iversen, the man behind the statistical model in chapter 3, Brian B. Mortensen for good collaboration on the DAVID demonstrator, and Henrik Christiansen for many fruitful discussions regarding the DAVID network architecture.

Special thanks to all partners in the European IST project DAVID. In particular Joerg Karstad and Harris Linardakis for their never failing optimism during the stressful integration meetings in Stuttgart and Marcoussis.

Finally, I would like to express my sincere thanks to Henrik Wessing, Henrik Christiansen, Karina Larsen, and Jacob Smitt for proofreading and commenting on this thesis.

Publications.

- M. S. Berger, V. B. Iversen. "Basic principles for MPLS traffic engineering" *International Seminar on Telecommunication Networks and Teletraffic*, pp. 23-30, St. Petersburg, Russia (2002). [63]
- B. B. Mortensen, M. S. Berger, "Optical Packet Switching Demonstrator", *European Conference on Optical Communication (ECOC)*, Copenhagen, 2002. [44]
- B. B. Mortensen, M. S. Berger "Optical Packet Switched Demonstrator" *PS'2002 proceedings*, Cheju Island, Korea, July 2002. [45].
- B. B. Mortensen, M. S. Berger, C. Linardakis, R. Jociles-Ferrer, "Metropolitan Area Network Optical Packet Switch Demonstrator", *proceedings of IST 2003*, Isfahan, Iran, August 2003 [46].
- M. S. Berger, V. B. Iversen, B. B. Mortensen, "Analytical performance evaluation of optical packet network interface", *COIN 2003*, Melbourne, Australia, July 2003. [47]
- B. B. Mortensen, M. S. Berger, "Estimating timeout parameters for packet aggregation" *COIN2003*, Melbourne, Australia, July 2003. [49]
- M. S. Berger, B. B. Mortensen, V. B. Iversen, R. Jociles-Ferrer, "Evaluation of Delay Bound for QoS provisioning in Optical Packet Network Interface", *7th WSEAS International Conference on Communications*, Invited paper, Corfu, Greece, July 2003 [50]
- L. Dittmann, H. Christiansen, M. S. Berger, "Hierarchical MPLS – An approach for efficient resource administration in multi-technology networks", *NOC 2001*, Ipswich, England, 2001. [51]
- M. S. Berger, "Multipath packet switch using packet bundling", *Proceedings of Proceedings of Workshop on high performance Switching and Routing (HPSR) 2002.* Kobe, Japan, May 2002 [78]
- H. Christiansen, M. S. Berger, "Novel, hiearachical, MPLS-based network architectures and their role in migration strategies towards future, optical, packet switched networks", *CIIT 2002*, St. Thomas, USVI, November 2002 [52]
- M. S. Berger, H. Christiansen, B. B. Mortensen, R. Jociles-Ferrer, "Hierarchical Electro-optical Packet Network Architecture", *proceedings of IST 2003*, Isfahan. Iran, August 2003 [53]

- M. S. Berger, "IP Lookup with fast update and low memory requirement", *Proceedings of Workshop on high performance Switching and Routing (HPSR) 2003*. Turin, Italy, June 2003 [108]
- C. Linardakis, B. B. Mortensen, M. S. Berger, R. Jociles-Ferrer, "Implementing the access control of packets on a slotted WDM MAN ring at 1 us add-drop rates", *WSEAS Transactions on Computers*, Issue 4, Volume 3, p. 1069-1074, October 2004

1. Introduction

The concept of packet communication was developed in the early 1960s as means to create an efficient and survivable system for computer communications. Then, in the early 1970s, the Transmission Control Protocol and the Internet Protocol (TCP/IP) were invented. In the early 1980s, Arpanet was upgraded with TCP/IP, and the Internet was born. The Internet began to grow very rapidly in the early 1990s after the invention of the Hyper Text Markup Language (HTML) and the introduction of Web browsers. Today, there is no doubt that packet communication will continue to grow, and there is a need to do research into this field to come up with new and improved network architectures, node architectures and protocols.

Wavelength Division Multiplexing (WDM) has resulted in the ability to have a huge amount of bandwidth on a single fibre, and the switch nodes become the bottleneck because the amount of bandwidth that can be carried on a fibre by far exceeds the processing capability of a node. In this respect, optical packet switching has been proposed as means to exploit the capacity of WDM transmission systems in an efficient manner while at the same time providing the flexibility that can be obtained by packet switching.

However, a lot of new problems arise when optical switching equipment is introduced in networks. This is mainly due to the limited processing functionalities within the optical domain. It is therefore important to determine a network architecture that works with the limitation in optics but at the same time provides the functionality known from today's Internet. Also, the architecture should support a gradual introduction of optical switching. In this respect, Multi Protocol Label Switching (MPLS) is a promising concept due to the separation of network layer control functions and packet forwarding functions in the optical packet switches.

The pan-European research project DAVID that successfully ended in October 2003 aimed at defining a viable scenario for introducing optical packet switching, both in the Metropolitan Area Network (MAN) and in the Wide Area Network (WAN). Several achievements from this project, including the network architecture, will be described in this thesis.

At this point, it is important to stress that optical packet switching is still a prospective technology, and that electrical packet switching will be dominant for several years. It is therefore reasonable to perform research

into this area as well in order to propose more efficient and scalable architectures. The next step in the evolution towards all optical packet networks could be hybrid solutions, i.e. switches with electrical interfaces and optics in the switch backplane.

Increasing speed does not only complicate switching. Also the tasks of address lookup and packet classification become more complicated and there is ongoing research in these areas to come up with new or improved algorithms.

As indicated above, new or improved technology can increase the available network capacity, but a more efficient usage of existing resources is also attractive. The objective of traffic engineering is to obtain high network utilisation by a proper path selection scheme. Furthermore, the network should be able to provide Quality of Service (QoS) in order to support real-time and near-real-time applications that are sensitive to delay and throughput variations and to packet loss. Issues related to Traffic Engineering and QoS are also discussed in this thesis.

1.1. Thesis organisation

Chapter 2 is a general introduction to concepts related to optical and electrical packet switching and packet switched networks. Switch architecture examples are given for optical switches, and the different building blocks are presented. This is followed by a brief introduction to electrical packet switching and a comparison between optical and electrical switch technology. Finally, a short description of protocols and concepts for packet switched networks is given, e.g. IP, ATM and MPLS. The overall goal of this chapter is to provide the background information needed for the remaining chapters.

Chapter 3 covers the DAVID project. This chapter starts out with a general introduction to the whole project followed by sections on the MAN and WAN part, respectively. The topology of the MAN is given by a number of fibre rings interconnected by a Hub. In the MAN section, the main focus is on the ring node architecture and the process of generating fixed size optical slots from variable length client layer packets. A statistical model is presented to determine the delay for this process. The WAN section presents the hierarchical MPLS based DAVID core network and the architecture of the optical packet switch. It is shown how MPLS can be utilised to create a unified network with several technologies. The DAVID consortium has performed a comprehensive benchmarking study to compare the proposed technology with alternatives. A part of this study, comparison of power consumption

between optical and electrical switch fabrics, is presented in the last part of this chapter.

Chapter 4 considers MPLS traffic engineering and the related protocols. The path selection procedure for traffic engineering is described. A traffic engineering scheme for best effort TCP/IP traffic is proposed. The basic idea is to group TCP connections into trunks, and then route the trunks based on MPLS traffic engineering procedures. Finally, constrained based routing, i.e. routing of flows with e.g. bandwidth constraints, is examined in more detail.

The remaining chapters, chapter 5, 6 and 7, shift focus from networking issues to internal electrical switch design. Chapter 5 introduces a multistage/multipath packet switch fabric where packets are collected and transmitted over identical parallel planes as an alternative to traditional load-balancing schemes that require packet re-sequencing at the outputs. A simple scheduling algorithm that applies time stamps to arriving packets and serves packets in order of increasing time stamps is proposed. Worst-case scheduling delay and buffer occupancy are derived for this specific scheduling algorithm and compared to average performance in a simulation study.

Chapter 6 presents a modified architecture for a buffered crossbar switch that overcomes the memory bottleneck with only a minor impact on performance. The proposed architecture uses two levels of backpressure (on/off flow control) with different constraints on round trip time. Buffered crossbars require only a simple scheduler that operates independently for each output queue column. The memory amount required for a buffered crossbar is proportional to the square of the number of ports and the round trip time. The proposed architecture reduces the amount of memory in the buffered crossbar without increasing the scheduling complexity. A simulation study is presented in order to evaluate the performance and to dimension the system.

Chapter 7 focuses on the forwarding task, i.e. address lookup and classification, and the chapter presents an IP address lookup algorithm with low memory requirements and fast updates. The scheme, which is denoted prefix-tree, uses a combination of a trie and a tree search. This is efficient in memory usage because the tree contains exactly one node for each prefix in the routing table. The time complexity for update operations is low for prefix-tree. The lookup operation for the basic binary prefix-tree may require that a high number of nodes are traversed. This chapter presents improvements to decrease lookup time.

Finally, in chapter 8, concluding remarks are given.

2. Electrical and Optical Packet Switching

In future, packet switched networks optical technology is expected to play a key role in the network nodes. Electrical packet switching nodes can scale to high capacities, but optical packet switching technology may deliver the same capacity at a lower cost and with lower power consumption.

This chapter presents basic architectures of electrical and optical packet switch nodes in section 2.1, followed by a brief introduction to packet network protocols in 2.2. The overall goal is to provide background information for the remaining chapters and to discuss optical switching technology versus electrical switching technology.

2.1. Electrical and optical switch nodes

There are a number of differences in the basic architecture of an electrical and optical packet switch, mainly due to technological constraints imposed by the technology. As an example consider contention resolution; electronic packet switches make use of random access memory, which can store packets for a random time period. However, a similar device is not available in the optical domain, and thus completely different contention resolution mechanisms are required. The following subsection will present an overview of electrical and optical packet switch architectures, respectively.

2.1.1. Optical Packet Switch architecture

This section reviews a number of optical packet switch architectures and introduces several important terms. Figure 1 shows the architecture of a generic OPS (Optical Packet Switch) core node consisting of an Input Module, an Optical Switch Matrix, an Output Module and an Electronic

Control block. The optical packets arrive at the input module on fibres carrying one or more wavelengths. The packets can either be of constant or variable length, and the switch operation can be either slotted or un-slotted. Usually the slotted operation in combination with fixed size packets is taken because of the performance gain and lower scheduling complexity, but un-slotted operation is considered for variable length packets.

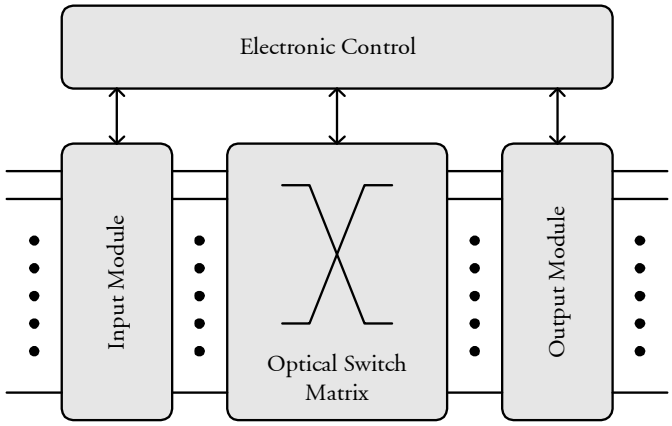

Figure 1: Generic OPS node

2.1.1.1. Input Module

The Input Module performs header processing, synchronisation, wavelength conversion, etc. The header-processing block inspects the packet header and transmits routing information to the Electronic Control block. The Control block calculates a permutation for the Optical Switch Matrix and determines a new packet header (label swapping). The packet header can be encoded in several ways: It can be encoded in-band, and in this case a guard band separates header and payload, which makes header re-writing possible. This approach is taken in DAVID. The header can also be encoded at a lower bit-rate than the payload to reduce the electronic processing costs. Another method is Sub Carrier Multiplexing [4], which is more efficient in terms of timeslot usage, but requires high-speed electronics for high payload bit-rates. The IST project STOLAS (Switching Technologies for Optically Labeled Signals) uses a novel approach with DPSK modulated header information and proposes a simple approach for label swapping with a Mach-Zehnder Interferometer (MZI) [5]. The label-processing techniques described so far are all based on electronic processing, but all optical header processing is an on-going research topic. An optical label-swapping technique using

a Mach-Zehnder Interferometer based XOR gate is proposed in [1]. Another approach is to avoid label-swapping at all; the schemes described in [2][3] use a fixed routing key, which is calculated by the Chinese Remainder theorem based on ID's of the nodes to traverse.

Synchronisation is performed both at a coarse and at a fine level. The coarse synchroniser aligns each packet to the internal slot clock. This is necessary mainly because of propagation delay between optical switches, but also temperature variations may impact the delay. Re-synchronisation at the coarse level is done on a longer time-scale and not packet by packet. The fine synchroniser handles jitter at the packet level and delay variations due to chromatic dispersion. It operates on each wavelength individually. Jitter occurs because of path differences in the preceding optical switch matrix, and the fine synchroniser avoids that jitter accumulates in cascaded switch nodes. The delay elements of the synchronisers can be implemented by fibre delay lines [6].

Wavelength conversion in the Input Module can be done for several reasons; either because of a higher number of internal wavelengths in the switch matrix or because of an internal wavelength routed structure where the wavelength determines the destination switch port.

2.1.1.2. Output Module

Signal regeneration is required when interconnecting many OPS nodes. The regeneration process is usually split into three categories: Re-amplification (1R), Re-amplification and Re-shaping (2R), and Re-amplification, Re-shaping and Re-timing (3R). An O/E/O conversion is a way to perform 3R regeneration, and also the most cost-effective way if the bit-rate does not exceed 10 Gbit/s, according to [7]. Among the all-optical solutions, Semiconductor Optical Amplifier (SOA) based interferometers (e.g. MZI) are promising because of the regenerative effects and potential in terms of integration [7]. The nonlinear transfer-function of the SOA-MZI gives 2R regeneration. 3R regeneration requires a recovered clock, which is injected into one arm in the MZI. All-optical clock recovery has been demonstrated already [8]. Furthermore, it has been demonstrated that packet power variations can be compensated in this structure [9].

The level of regeneration impacts the transparency of the network; 2R regeneration requires that the signal is digital because the un-linear transfer-function will destroy analogue information. However, 2R regeneration supports bit-rate transparency, which is not the case for 3R regeneration because a recovered clock close to a reference clock must be extracted from the optical signal.

2.1.1.3. Optical Switch Matrix

The Optical Switch Matrix is a core component that performs switching and contention resolution. Three well-known approaches to optical switching are Micro Electro-Mechanical Systems (MEMS), Arrayed Waveguide Gratings (AWG) and broadcast and select structures based on e.g. SOA gates operated as ON/OFF switches. Since optical packet switching requires fast reconfiguration, MEMS are not suitable. Architectures based on AWG and SOA have been proposed in the literature, and later in this section, some examples will be shown.

The objective of contention resolution is to avoid packet loss when two or more packets are destined for the same output. Three types of contention resolution are available [10]: (1) deflection routing, (2) optical buffering and (3) wavelength conversion.

Deflection routing works by transmitting contending packets by different outputs. Some of the packets will thus travel along sub-optimal paths, and they may arrive out of sequence. However, deflection routing is considered as a viable alternative to bulky optical buffering approaches.

Electronic routers store packets in random access memory (RAM). In principle photonic RAM is realisable by a bi-stable system using un-linear optical effects, but it is very costly to store a single bit by this approach. Therefore, fibre delay lines are used to implement optical buffers. A buffer with N positions is realised by N fibre delay lines where the length of line number n is n times the length of an optical packet (examples are shown in Figure 2). Furthermore, the buffer capacity can be extended by WDM without adding more fibres. Optical buffers can be placed at various locations in an optical switch, e.g. as input buffer, re-circulating buffer or output buffer.

Finally, wavelength conversion is a tool to solve contention by assigning different wavelengths to packets destined to the same output port. This is similar to deflection routing, but a new wavelength is selected instead of a new port. From a performance point of view, exploiting wavelength conversion greatly improves the performance [11][12] and can reduce the amount of bulky fibre delay lines.

2.1.1.4. OPS examples

A high number of OPS architectures have been proposed in the literature. This section reviews three different optical packet switch architectures shown in Figure 2.

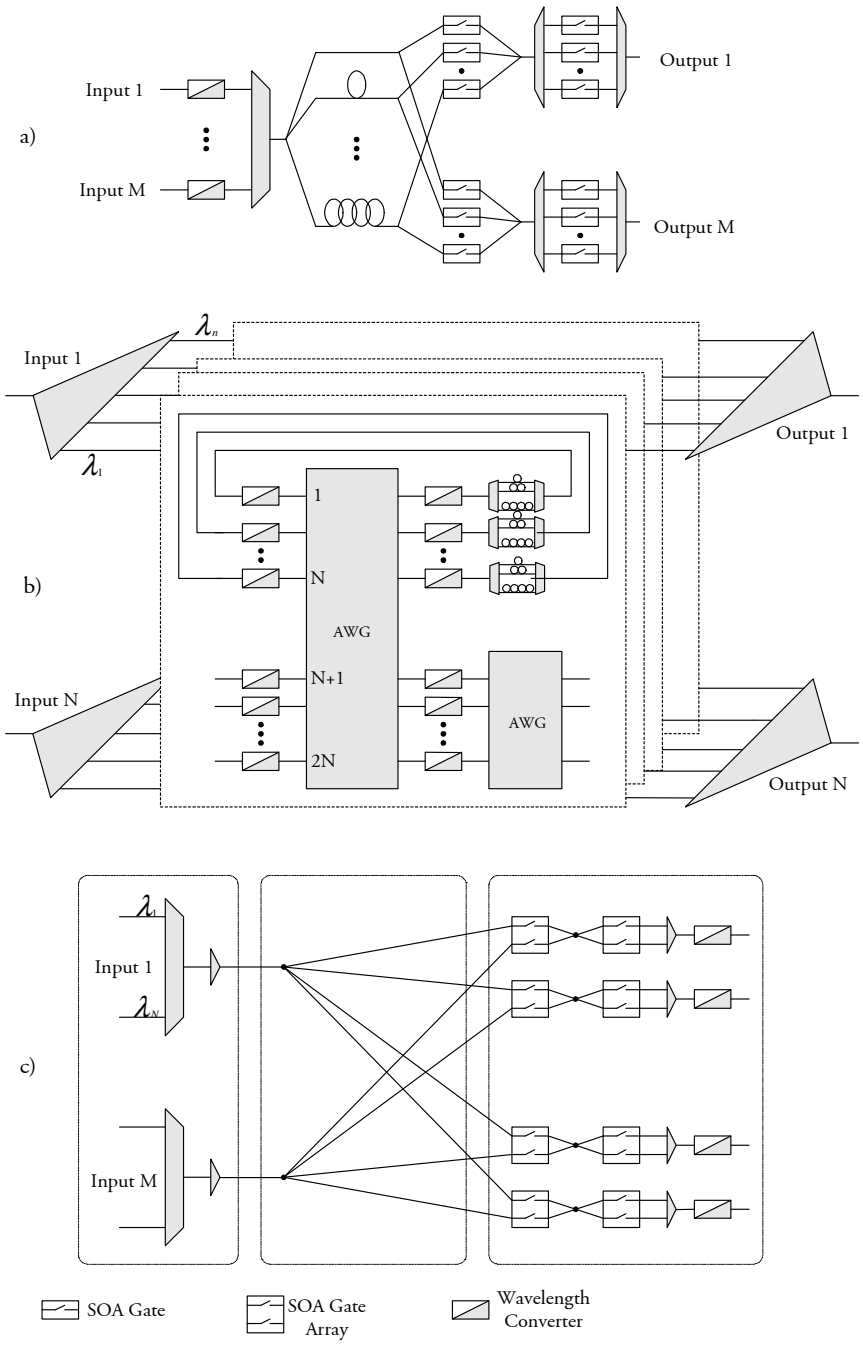

Figure 2: a) KEOPS switch [11], b) WASPNET architecture based on AWG [4], c) DAVID Architecture based on integrated SOA gates [13].

The first example (Figure 2a) shows the broadcast and select switch proposed by the ACTS project Keys to Optical Packet Switching (KEOPS). The switch is designed for a network with a single incoming wavelength on each fibre. The switch works as follows: The incoming packets are wavelength converted at each input to a wavelength specific to each input. The packets are then multiplexed and split to all fibre delay lines. The first level of SOA gates select among delay lines and the second level of gates select the specific wavelength (input). In this architecture, the fibre delay lines are efficiently utilised by WDM to implement an individual output buffer for each switch outlet, but the switch does not support WDM on the external links.

The Wavelength Switched Packet Network (WASPNET) packet switch [4] (Figure 2b) is based on AWG wavelength routers and fibre delay lines feedback buffers. Each plane handles a separate input wavelength. The first *2Nx2N* AWG switches packets to either the output AWG or to the fibre delay line buffers in case of contention. The second *NxN* AWG is required in order to be able to transmit several different wavelengths at the same time destined to the same output. The switch is not non-blocking due to the second AWG and the wavelength converters at the input.

The third example shown in Figure 2c has been proposed in DAVID [13]. The first stage of the switch multiplexes all input wavelengths into a single fibre and amplifies them before the subsequent power-splitting stage. For each output wavelength, one stage of SOA gates selects among inputs, and the second stage selects among wavelengths. Finally, a wavelength converter performs translation to the desired output wavelength. The DAVID OPS uses feedback fibre delay lines for contention resolution (not shown in Figure 2c).

2.1.2. Electrical packet switch architecture

The main building blocks of an electronic packet switch are shown in Figure 3. Later generations of packet switches usually separate the two major functions, control and forwarding. The control plane (CPU block in Figure 3) is responsible for building and maintaining a routing table as well as management functions. Routing protocols are used to exchange information with other routers, e.g. RIP, OSPF, and BGP. Control functions are usually implemented in software.

The data plane is responsible for packet forwarding, which is performed in the traffic manager block shown in Figure 3. The traffic manager performs packet header inspection in order to determine destination and

priority. Furthermore, the traffic manager is responsible for queuing, scheduling, segmentation and reassembly. Network Processor Units (NPU) is a specialised microprocessor device optimised for traffic manager applications. The Network Processing Forum (NPF) [15] establishes common specifications including software API, electrical interfacing and benchmarking.

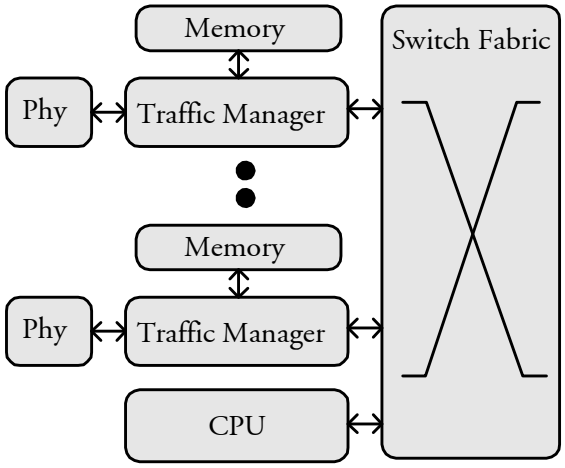

Figure 3: Electrical packet switch architecture

The central part of the packet switch is the *switch fabric*, and its task is basically to connect an input to one or many outputs. The NPF specification CSIX-L1 defines a standard interface between traffic managers (e.g. network processors) and switch fabrics. CSIX-L1 defines the Cframe, which is the basic switching unit for the fabric. The payload can be 1-256 bytes, but typically a smaller maximum is used, e.g. 64 bytes.

The CSIX interface is not optimised for 10 Gbit/s operation, and the result is a new specification, the NPF Streaming Interface (NPFSI) [16]. It uses SPI-4.2 electrical interfaces specified by the Optical Internetworking Forum (OIF) [17].

2.1.2.1. Traffic Manager

The traffic manager performs various tasks, including:

Segmentation and Reassembly (SAR): In the ingress direction (from Phy to switch fabric) reassembly is performed if required as in case of IP over ATM. Furthermore, segmentation into internal switch fabric cells, e.g. Cframes, is performed. In the egress direction, a reassembly function

optionally transforms e.g. Cframes into packets, and finally a segmentation function is invoked if required by the Phy interface block.

Address lookup and classification: A packet header inspection determines the protocol type, destination address, and other forwarding related information such as service class. The classification process inspects several header fields in various protocol layers to determine flow identification, QoS level, compliance to security/firewall rules etc.

Queuing and Scheduling: In the ingress direction, once the packets are identified, they are placed in a specific queue depending on e.g. destination and service level. Packets are also queued in the egress direction to provide e.g. QoS.

Traffic manager functions can be implemented in dedicated hardware (ASIC's), programmable devices or a combination hereof. Dedicated hardware delivers high speed at the cost of reduced flexibility. Network processors offer a programmable solution with an internal architecture optimised for the tasks described above. A network processor typically contains several internal processor cores executing code in parallel or seriel.

This thesis will not cover traffic manager design in particular, but address lookup and classification are covered in more detail in chapter 7.

2.1.2.2. Switch Fabric

A high number of switch fabric solutions have been proposed so this topic could easily take up an entire book [18]. This section will only cover some selected example architectures to provide sufficient background for chapter 5 and chapter 6.

In general, the switch fabric can be classified as *single stage, multi path* or *multi stage*. In a *single stage* switch, all packets will pass the same central stage in the switch. A *multi path* switch has several possible paths between input and output, but the packets will only pass one stage on the trip. A *multi stage* switch is characterised by packets traversing several stages. Furthermore, the number of stages can be variable. The literature contains numerous results on Multistage Interconnect Networks (MIN), which contain a high number of small switching elements [19]. Well-known MIN's are Banyan, Benes and Batcher-Banyan. These types of switching networks were mainly developed to reduce the number of (expensive) crosspoints in circuit-switched nodes and are not commonly used in modern high-speed packet switches.

Single stage: usually, single stage switch architectures are classified by the location of memory, i.e. input buffer, output buffer, shared buffer or

crosspoint buffer. Practical switch solutions often make use of a combination of the buffer schemes [20]. Figure 4 shows a combined input and Output buffered crossbar switch with *N* input and *N* output. Note that ingress and egress port cards are shown as separate entities, but they are typically integrated on the same device. To improve performance, the line rate of the internal links between port cards and switch card will run at a faster speed than the external rate. The speed difference is referred to as *speedup*. The speedup can be in the range from 1 to *N*. A speedup value of 1 corresponds to pure input queuing since the output buffer is superfluous for rate adaptation between switch card and port card. On the other hand, a speedup of *N* is equal to pure output queuing because in each timeslot, all ingress port cards can be served. When the speedup lies between 1 and *N,* both input and output buffers are required.

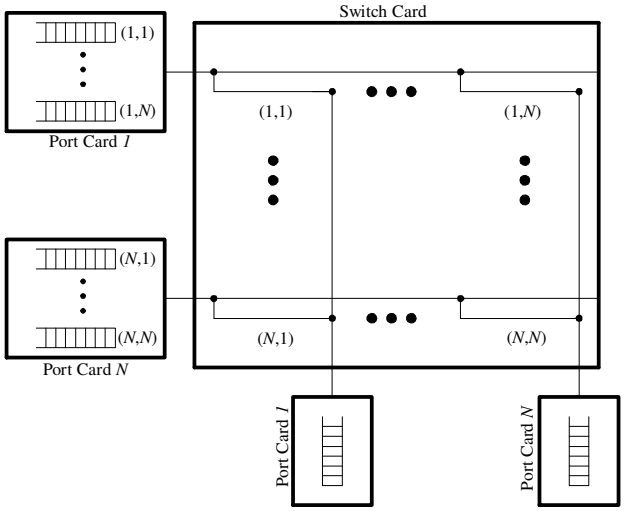

Figure 4: Combined Input and Output Buffered switch

To overcome Head Of Line (HOL) blocking in the input buffers, each input port maintains a separate queue for each output port, also known as Virtual Output Queuing (VOQ). A scheduling algorithm is required to select the packets from the input VOQ buffers that are transferred to the output buffers.

The scheduling problem can be represented by a bipartite graph with *N* vertexes representing input and *N* vertexes representing outputs. There is an edge between input node *i* and output node *j* if VOQ(*i,j*) holds at least one cell. A weight can be associated with each edge that indicates e.g. the age of the cell or the length of the queue. A *maximum* match in the bipartite graph is defined as a match that pairs the maximum number

of input and output. A maximum weight match will furthermore take the weight into account, such that match with the highest sum of weights is selected. Several maximum weight matching algorithms are presented in [21]. 100 % throughput is achieved with no speedup, but the time complexity is in the order of $O(N^{2.5})$, not feasible from an implementation perspective.

In contrast to a maximum match, a *maximal* match has no cell in any input VOQ belonging to an unmatched input destined for an unmatched output. This definition allows for iterative scheduling algorithms that terminate when no further matches can be found. The first algorithm proposed to perform a maximal match is PIM (Parallel Iterative Matching) [22]. An iteration of the algorithm consists of the following three steps:

1. Request: Each unmatched input sends a request to all the outputs for which it has a queued packet.
2. Grant: If an unmatched output receives any requests, it grants to one by randomly selecting among them.
3. Accept: If an input receives grants, it accepts one output randomly selected among those that granted to this input.

The three steps of PIM are illustrated in Figure 5. In this example, the following VOQs have a cell available: (1,1) (1,2) (2,3) and (3,2). After iteration, input 1 is matched to output 2, and input 2 is matched to output 3. This is in fact a maximal match because the remaining unmatched input and output cannot be matched (i.e. input 3 does not contain a cell for output 1).

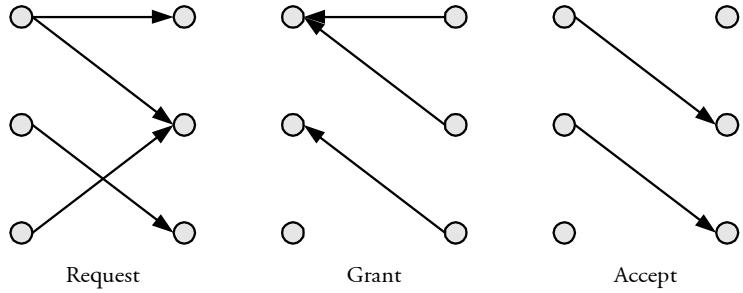

Figure 5: Iteration of PIM in a 3x3 switch.

Nevertheless, the match shown in Figure 5 is not maximum; the size of the match becomes 3 by matching the following input and output: 1-1, 2-3 and 3-2. The example above created a maximal match after the first iteration. However, in general a guaranteed maximal match requires *N*

iterations, and the worst-case time complexity becomes $O(N^2)$. Simulations have shown that *log(N)* iterations are sufficient to obtain a maximal match. Taking this result into account, the algorithm is feasible to implement.

PIM with a single iteration will give a switch throughput of approximately 65 % for uniformly distributed Bernoulli traffic. In [23] an improvement to PIM is proposed. The algorithm called iSLIP, iterative SLIP, is similar to PIM, but in the 3 phases iteration the random selection in the grant and accept step is substituted by a round-robin arbitration. A round-robin scheduler is located at each input and at each output. The output scheduler selects among inputs in the grant phase, and the input scheduler selects among outputs in the accept phase.

SLIP scheduling (an iteration of iSLIP) achieves 100 % throughput for uniformly distributed Bernoulli traffic, but throughput drops for e.g. unbalanced traffic. Running several iterations will improve throughput, but 100 % throughput cannot be achieved because the algorithm performs a maximal match, and not a maximum match. As with PIM, *log(N)* iterations are sufficient.

A speedup between port cards and switch card can improve performance. It has been shown in [24] that a speedup of 2 is sufficient to achieve 100 % throughput for a maximal match scheduling algorithm.

Scheduling complexity can be reduced by introducing small buffers in the crosspoints [87], which is shown in Figure 6.

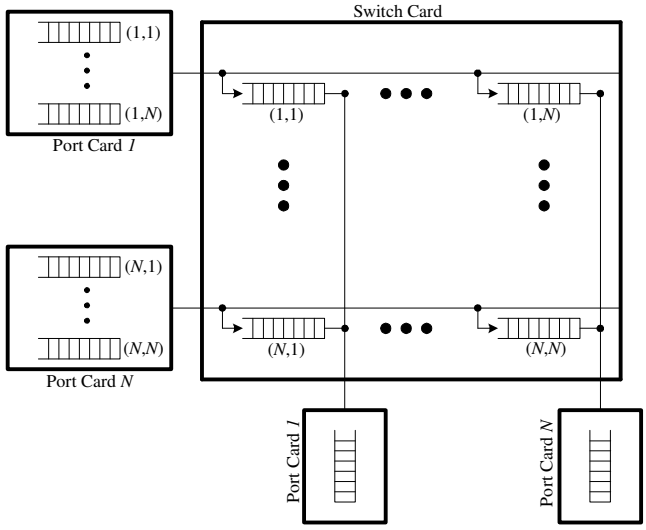

Figure 6: Combined input, crosspoint and output buffered switch.

The buffered crossbar switch requires simple independent schedulers for each output column, and due to the crosspoint buffers, there is less stringent synchronisation requirements between port cards and switch card. A backpressure mechanism is required between VOQ buffers and the switch card to avoid overflow in the crosspoint buffers.

A major drawback is the total memory amount that scales as $O(N^2)$, and each crosspoint buffer must furthermore be sufficiently large to handle the round trip time for backpressure. Chapter 6 presents a modified architecture for a buffered crossbar with two levels of backpressure that overcomes the memory bottleneck with only a minor impact on performance.

Multistage/multipath switch: The 3 stage Clos network shown in Figure 7 is a well-known multistage network architecture. The number of different paths from input to output is equal to the number of middle stages. An additional routing mechanism is required to determine the route of a cell in the Clos network.

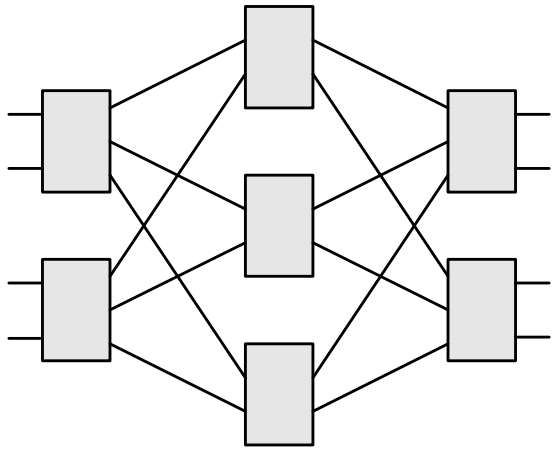

Figure 7: Clos network.

Algorithms for load-balancing have been proposed. An algorithm denoted Concurrent Dispatching [25], used in the ATLANTA chipset from Agilent, balances the load among middle stages in the following way: Each input stage sends requests to each middle stage, which then calculates a permutation between input and output and either accepts or rejects each request. The second stage is composed of bufferless crossbars, which ensure that packet sequence is maintained because constant delay across the middle stages is obtained. The expansion factor is defined as the number of output divided by the number of input at the first stage.

(In Figure 7 the expansion factor is 3/2 = 1.5) An expansion factor greater than one can compensate for inefficiencies in the routing function.

Chapter 5 proposes a load-balancing scheme, which ensures that each middle stage receives identical traffic patterns. The packets will then receive identical delay independent of the selected middle stage, even if the middle stage is a buffered switch.

2.1.3. Optical switching vs. electrical switching

As indicated in section 2.1.2 an electrical packet switch design can be scaled to support very large capacities (e.g. several terabits pr. second) by designing multipath/multistage architectures. The chip count will typically be high complicating backplane design and increasing power consumption. Furthermore, the power consumption of an electrical interconnect will increase proportionally to the bit rate. By moving the signals to optical links, a low and bit rate independent power consumption is obtained, and several wavelengths can be carried simultaneously; a main reason to introduce optical packet switching.

An important design issue is the choice of packet length. In the electrical domain, the packet length is typically in the range of 64-128 bytes to obtain an efficient filling in case of minimum sized IP packets. However, in optical packet switching a larger packet size is required to cope with efficiency reduction due to guard band and scheduling complexity. A feasible optical packet size is considered to be 1 μs in the DAVID project. At 10 gigabit, this is more than 10 times the packet size of an electronic switch making it more difficult to guarantee an efficient payload filling compared to the electronic switch.

Wavelength converters, 3R-regenerators and bulky optical buffers implemented by fibre delay lines dominate the total cost of an optical switch fabric. According to section 2.1.1.2, a full O/E/O regeneration is the most cost-effective solution for bit rates up to 10 Gbit/s. This indicates that optical packet switching might be the preferred solution for 40 Gbit/s whereas electronic packet switching is the most cost-effective solution for 10 Gbit /s and below.

Optical buffering e.g. by fibre delay lines adds to the cost and size of the switch. Using only the wavelength domain for contention resolution is therefore potentially attractive. The network throughput for bufferless optical packet switching has been studied in [29][30]. It is shown that a maximum load of 30 % for a 16x16 switch with 32 wavelengths is allowed for a packet loss rate below 10^{-10}. The result is obtained for

Bernoulli traffic and might be worse for bursty and unbalanced traffic. 30 % load will reduce a 40 Gbit/s pipe to 12 Gbit/s, which is probably not acceptable. Also shown in [30] is the impact of limited wavelength conversion on the switch throughput. With 32 wavelengths, a conversion distance of 10 is sufficient to obtain 99 % of maximum load. The result is interesting because the cost of a wavelength converter depends on the tuneable range.

At this time, it is difficult to predict whether optical packet switching will become successful. Further research is important to devise cost-effective and highly integrated solutions to buffering, wavelength conversion and 3R regeneration. The roadmap towards optical switching could contain hybrid solutions, that is switches with electrical interfaces and optics in the switch backplane. This approach is studied in the Optical Router project at Stanford [31], and a terabit O/E/O/E/O router prototype is reported in [32]. Furthermore, this topic is covered in thematic issues of Journal Of Optical Networking [33] and Journal of Selected Areas in Communications [34].

As stated in the beginning of this section, the reduction in power consumption is a main motivation for introducing optical packet switching. Chapter 3 presents a benchmarking study that estimates and compares the power consumption of electrical and optical packet switches, respectively.

2.2. Packet network protocols

Basically a packet switched network protocol can be classified as either connection-oriented or connection-less. ATM is an example of a connection-oriented protocol where the connection is established by signalling prior to data exchange whereas IP is an example of a connection-less protocol with a routing decision performed in each node.

ATM cells are small compared to the size of a feasible optical slot. However, a number of ATM cells can be collected to form an optical slot. The optical packet network could then constitute a Virtual Path (VP) switched domain within the ATM network. Consider as an example an ATM network that switches at the VP level at the core of the network. All ATM cells assigned a specific VPI value in an ingress VP cross connect will follow the same route of VP cross connects. It is therefore possible to bundle a number of ATM cells that will follow the same path together in a larger optical packet (which is no longer an ATM cell). The benefit of this approach is a higher switching capacity in the core since switching

speed (both for optical and electronic switches) is commonly limited by a maximum number of packets per time unit.

The size of an IP packet is variable and is in the range from a small fraction of an optical slot to several optical slots. It is therefore not feasible to perform IP routing decisions in the optical packet switched domain because it requires potential inspection of several IP packets contained in the optical slot. Multi Protocol Label Switching (MPLS) allows for separation of IP routing and forwarding and is therefore a likely candidate for IP over optical packet networks.

2.2.1. MPLS

The MPLS working group in the Internet Engineering Task Force (IETF) was founded in 1997 in order to create a standard based on the various proposals for IP switching and tag switching. The basic idea in MPLS is that packets are forwarded by label switching, that is, a short fixed size label is appended to the layer 3 header, and the packet is switched across the network, which is similar to the operation of ATM. Originally, the motivation for label switching was to improve forwarding speed of IP routers by replacing the longest prefix match operation with a simple exact match table lookup. MPLS is thus a connection-oriented protocol, and the MPLS connections are denoted Label Switched Paths (LSPs). MPLS separates the control plane and the forwarding component in the Label Switch Routers (LSRs). Routing protocols such as Open Shortest Path First (OSPF) can be used to populate IP routing tables within the control plane. A router will then bind a local label to each entry in the routing table. Label distribution is then required to inform neighbour routers of the label bindings. Label distribution can be performed in various ways e.g. by piggy backing the information on the routing protocol messages or by the use of a specific Label Distribution Protocol (LDP). The labels are assigned in the ingress LSR. Based on the content of the layer 3 header, the Forward Equivalence Class (FEC) is determined. Traffic within one specific FEC is always forwarded the same way across the MPLS domain. The relation between packet forwarding and the management/control plane in MPLS is depicted in Figure 8.

The connection-oriented nature of MPLS facilitates traffic engineering; explicit routes can be set up to optimise the usage of network resources. Explicit routes are normally determined by ingress LSRs and they are established by a signalling protocol e.g. Constrained Label Distribution Protocol (CR-LDP) [27] or Resource Reservation Protocol (RSVP-TE) [28]. In principle, explicit routes could be established within traditional

IP networks. However, this will require layer 3 classifications in each node along the path, and a protocol that distributes the filter information for the classifiers. For MPLS, the complex classification procedure is only performed in the edge node, which is beneficial to core networks comprised of optical packet switches where electronic packet processing is a potential bottleneck.

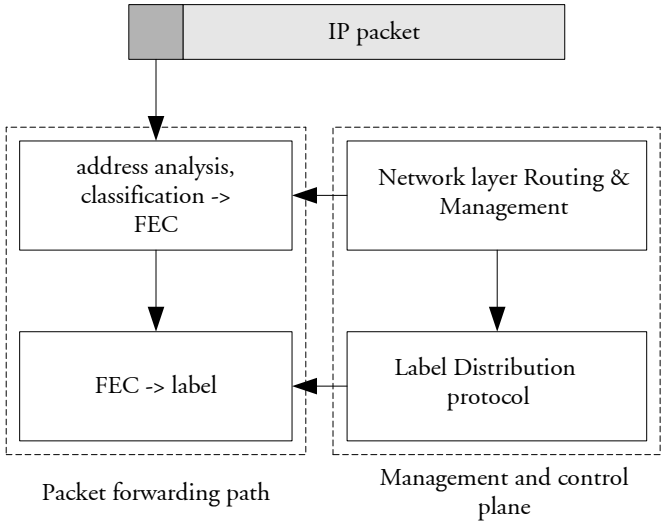

Figure 8: Data and Control flow in MPLS.

Furthermore, the connection-oriented nature of MPLS allows for fast restoration procedures [26], compared to the slow restoration in traditional IP networks.

2.2.1.1. Hierarchy and label stacking

In MPLS, the packets can contain more than one label organised in a *label stack*. The stack can either be pushed or popped in order to add or remove the top label, respectively. The concept of label stacking enables the creation of LSP tunnels. To put a packet into an LSP tunnel, the transmitting endpoint pushes a label for the tunnel onto the label stack and sends the packet to the next hop of the tunnel. The receiving endpoint of the tunnel will then pop the label (or the label can be popped at the tunnel's penultimate hop).

Extending the label stacking to more than two levels introduces the possibility of a MPLS routing hierarchy. This is utilised for the DAVID network architecture (see chapter 3) where the hierarchical level is determined by the actual switching technology.

2.2.1.2. MPLS and circuit switching

Basically, the MPLS control plane enables connection establishments (LSPs) across a network. The MPLS concept can therefore be extended to cover circuit switched networks as well. The framework is known as Generalised MPLS (GMPLS) [37] and covers e.g. SDH, WDM networks and switching at the fibre level. The label is implicitly given; in SDH it indicates a given TDM channel. In WDM, the label indicates a wavelength, and in case of switching at the fibre level, the label indicates a specific fibre in a cable consisting of several fibres.

Although the labels are implicitly given, label stacking is possible at the border between two technology domains, e.g. between a wavelength switched domain and a fibre switched domain; a fibre switched path constitutes a tunnel that carries several wavelengths.

2.3. Summary

The first part of this chapter presented an overview of optical and electrical switching technology, respectively. The main building blocks of a generic OPS node was described followed by three example architectures: KEOPS, WASPNET and DAVID. The section on electrical packet switch architecture described crossbar switch design including both buffer less and buffered architectures. Furthermore, multistage and multipath switch design were outlined as means for increasing the capacity.

The viability of optical packet switching was discussed. In conclusion, it depends on cost-effective and highly integrated solutions to buffering, wavelength conversion and 3R regeneration. Further research will show whether or not optical packet switching is a viable solution.

Finally, this chapter presented protocols for packet networks. MPLS is a likely candidate due to the separation of forwarding and routing and due to interoperability with wavelength and fibre switched GMPLS networks.

3. The DAVID network

The objective of the DAVID project is to propose and demonstrate feasible architectures for optical packet switching. The project covers both metropolitan- and wide area networks, and the focus is on several aspects including electronic and optical technology, traffic performance, control, management and prospective issues.

3.1. Overview of the DAVID project

The European IST project Data And Voice Integration over DWDM (DAVID) was launched in the summer of 2000. Partners contributing to the project are Alcatel SEL (D), Alcatel CIT (F), Research Centre COM (DK), National Technical University of Athens (G), IMEC, Ghent University (B), University of Bologna (I), University of Essex (UK), Laboratorie de Recherche Informatique d'Orsay (F), Politecnico de Torino (I), Institut National des Telécommunication (F), BT-Exact (UK), Universitat Politecnica de Catalunia (E), Telenor (N) and Telefonica (E).

The network architecture proposed by the DAVID consortium comprises both the Metropolitan Area Network (MAN) and the Wide Area Network (WAN) [38]. The physical topology of the MAN network is given by a number of fibre rings interconnected by a Hub. The fibres carry optical packets on several wavelengths, and ring-access is controlled by a MAC protocol running on a separate wavelength. The Hub is a bufferless optical packet switch, and contention resolution is performed by the MAC protocol. The MAC calculates switch matrix permutations, which are announced on the rings. The MAN network is described further in section 3.2. The main focus is on the ring node and the aggregation interface between client layer packets and optical slots. A statistical model of the aggregation process is presented and utilised to evaluate the performance.

The DAVID WAN was designed to fulfil several objectives; the network must be able to handle a wide range of capacities and support a migration path from electrical to optical packet switching. The network is

comprised of several technologies ranging from electronic packet switches to photonic cross-connects working at the fibre level. The concept of MPLS is utilised to establish a unified switching and routing approach covering the entire network. The DAVID WAN is the topic of section 3.3. The hierarchical MPLS based architecture is described, and the OPR node architecture is briefly covered.

Furthermore, the DAVID consortium has conducted a benchmarking study to evaluate the proposed MAN and WAN architectures against existing solutions. The goal of the benchmarking study presented in section 3.4 is to compare power dissipation in optical- and electrical packet switching. The expected power reduction for optical packet switching is one of the main drivers for optical technology in the switch nodes, and quantitative results will lead to better understanding of the design limitations set by the power consumption.

3.2. Metropolitan Area Network

Future Metropolitan Area Networks (MAN) must provide high capacity and improved flexibility. Optical networking will play a key role in this scenario. The first step is widely believed to be the introduction of a transparent optical WDM layer that can deliver sufficient capacity. The next possible step could be the introduction of optical packet switching to provide the required flexibility.

The DAVID project has among its objectives proposed a MAC controlled optical packet switched MAN. In section 3.2.1, a short introduction to the MAN network architecture is given. The main focus is on the ring node and the adaptation from variable length client layer packets to optical slots. A timeout parameter is required to limit the maximum waiting time during the bundling of client layer packets. The problem of bundling is widely studied by simulations [39][40]. In this chapter, a statistical model of this bundling operation is given in section 3.2.2. The model is used to evaluate the performance in section 3.2.3.

3.2.1. Network Architecture

The DAVID Metropolitan Area Network is composed of a number of fibre-rings interconnected by a Hub as shown in Figure 9a. Each fibre carries optical packets on several wavelengths. The wavelengths are divided into wavebands, e.g. 32 wavelengths can be divided into 8 wavebands where each band carries 4 wavelengths. The capacity can thus be upgraded in steps of wavebands. A waveband on a given physical ring

is denoted a logical ring. The optical packets have a fixed size of 8 Kbit, and at 10 Gbit /s the duration is approximately 0.25 us.

The Hub is an all-optical packet switch operating at the waveband level. The Hub employs wavelength converters, which enables conversion between wavebands. There is no buffering in the Hub, but the problem of contention is solved by a global Medium Access Control (MAC) protocol. The MAC protocol performs several tasks; it controls access at each individual logical ring, and it calculates waveband permutations in the Hub. The permutations are announced at the MAC channel of each ring. A detailed description of the MAC protocol and a fairness scheme is found in [41][42]. The feasibility of the MAN concept is verified by building a demonstrator [43][44][45][46]. The demonstrator will not be described further in this thesis.

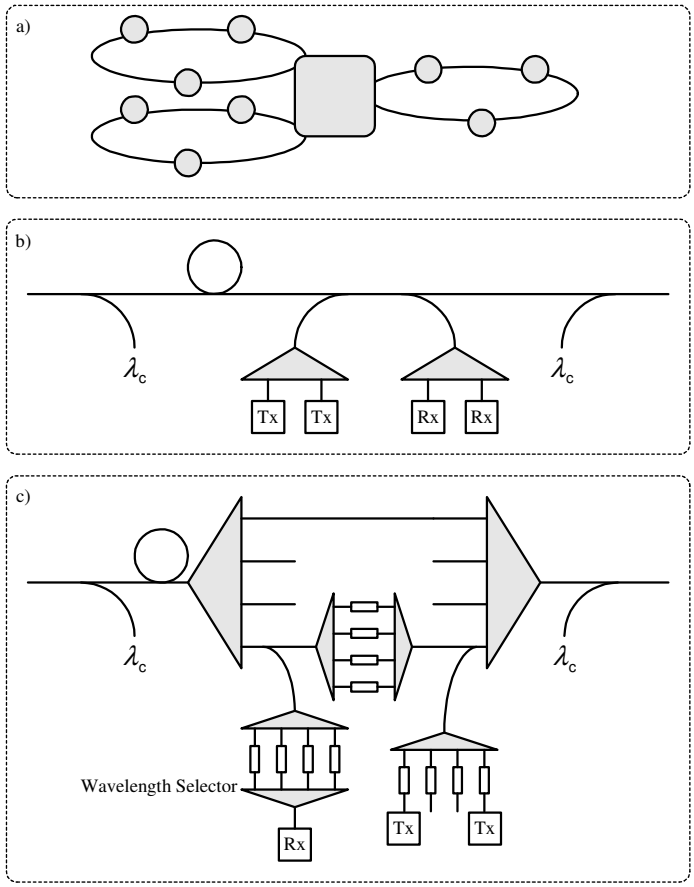

Figure 9: a) MAN network architecture. b) Passive OPADM c) Active OPADM

Figure 9b and Figure 9c show two different implementations of the optical part of the ring node, the Optical Packet Add Drop Multiplexer (OPADM). The first implementation shown in Figure 9b is intended for medium term applications. It only contains passive elements and the received and transmitted signals must reside on different wavelengths. Figure 9c shows a more advanced OPADM architecture intended for long-term applications. It contains active components (SOA gates) to perform wavelength selection at various stages. The OPADM can extract from and insert wavelengths to a single waveband of four wavelengths. In this example, only one wavelength in the waveband can be received at the same time. The wavelength selector between the drop and add part erases the wavelength that has been dropped, allowing for spatial reuse. The control channel resides on a special wavelength. It is terminated and generated in each node. A fibre delay line is inserted in the data-path to compensate for MAC processing delay. In order to further simplify the passive OPADM in Figure 9b, the MAC channel can be replaced by a simple collision avoidance scheme based on a photodiode for power detection. This approach was proposed for the DBORN concept [55][56].

The MAC protocol has two main objectives; it must ensure fair and prioritised access to the rings, and it must calculate permutations in the Hub. Two different fairness schemes have been proposed within the scope of the project. The first approach [42] is basically a generalisation of the token ring technique; a control message named SAT is circulated in a store and forward mode from node to node along the ring. A node forwarding the SAT is granted a transmission quota: the node can transmit up to Q packets before the next SAT reception. When a node receives the SAT, it will immediately forward the SAT to the next node if it is satisfied (hence the name SAT), i.e. if no packets are waiting or if Q packets were transmitted since the last SAT transmission. If the node is not satisfied, the SAT is stored until one of the two conditions is met. In DAVID the SAT scheme is extended to cover several logical rings; there is a SAT signal for each pair of logical rings, SAT(i,j).

The SAT signals can be used to calculate Hub permutations as well [42]. The round trip time for SAT(i,j) can be taken as a measure for the traffic amount from ring i to ring j. All ring pairs are then ordered after the SAT rotation time, and the ring pair with the largest round trip time is added to the permutation if the upstream ring was not previously selected as upstream ring, and if the downstream ring was not previously selected as downstream ring.

The MAC protocol proposal described above (from [42]) considers only best effort traffic and the fairness issue. Further work is presented in [41] that describes an approach for QoS provisioning by introducing traffic segregation; in addition to best effort traffic, the MAC supports high priority and real time traffic as well. The real time traffic is allocated a certain amount of timeslots, emulating circuit switching. The remaining optical slots are dynamically divided between first and second priority traffic. If a node has first priority traffic ready for transmission, but is unable to access the ring because of high load from second priority traffic, it will set a reservation bit in a second priority slot. With the reservation bit set, a high priority node can only reuse the slot (but it does not need to be the node that originally raised the reservation). When the node finally transmits a high priority packet, it must remove any reservation previously set, thus allowing the reservations to fluctuate in accordance with the demand for first priority traffic leaving the rest of the bandwidth for best effort traffic.

Provision of Quality of Service (QoS) is obtained by the Differentiated Services [35] approach for IP networks (See Chapter 4 for more information about Differentiated Services). The ring node performs the packet classification in order to assign the Diffserv class. The mapping between Diffserv classes and MAC QoS levels is straightforward: Expedited Forwarding (EF) is mapped to MAC level 1, Assured Forwarding (AF) is mapped to MAC level 2 and Best Effort (BE) is mapped to MAC level 3

The ring node is basically an access gateway between client layer networks and the optical packet MAN. An overview of the ring node building blocks is shown in Figure 10. The central Load Balancer is required in order to support several wavebands. Each burst mode interface corresponds to one waveband. The MAC channel (not shown) controls access to the wavebands. The ring node performs buffering in the stages before and after the Load Balancer, respectively.

The second stage of buffering (after the load-balancer) performs buffering of optical slots per destination logical ring, which is announced by the MAC channel based on the calculated permutations. (Note that the MAC channel is not shown in Figure 10)

The first stage of buffering generates the optical slots. There is one queue for each waveband for each destination ring node in the MAN. Client layer packets, e.g. IP/Ethernet, do not fit well into the optical slots of fixed length; they are therefore segmented into small (e.g. 64 byte) cells. The segments are then bundled to form an optical packet. In the DAVID network, the optical packet can hold 16 segments at maximum.

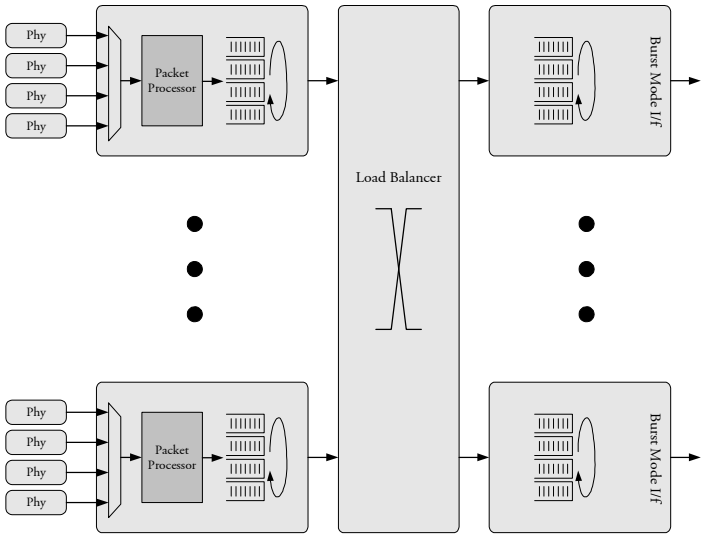

Figure 10: Ring node

The bundling operation is depicted in more detail in Figure 11. The incoming segments are sent to a specific queue depending on the destination. The number of queues is *Q*. The outgoing optical slot will only contain segments from one specific queue at a time, and it can hold up to *k* segments.

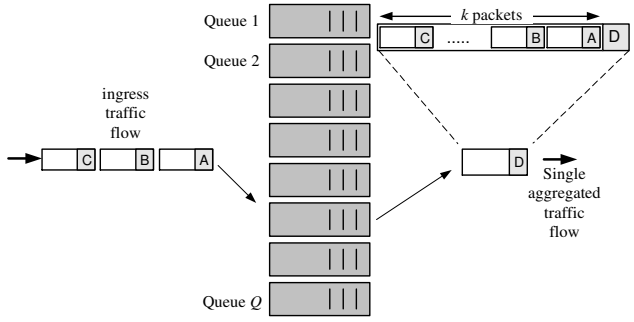

Figure 11: Segment bundling into optical slots

A scheduling mechanism is required in order to select a queue for transmission. The objective of the scheduling algorithm is to ensure that packets are bundled efficiently, but at the same time it must ensure that packet delay is bounded. Assume that a queue can only receive service if it contains at least *k* packets. In this case, the filling efficiency is high, but a packet can be delayed forever. To overcome this, a maximum waiting

time τ is introduced. When a packet arrives to an empty queue, a timer is started. The queue will be selected for transmission when it either contains k packets, or when the timer has expired after a time τ. The reader might jump directly to the WAN section (3.3) in order to avoid all the calculations presented below.

3.2.2. Statistical Model

This section considers a statistical model of the bundling operation shown in Figure 11. This section is based on results presented in [47]. The task is to find the average waiting time from arrival until the optical packet is transmitted. The waiting time is composed of two contributions. The first one is the time it takes from arrival until the optical packet is either full or the timer has expired (aggregation delay). The second contribution is the queuing delay until the optical packet is actually transmitted.

It is assumed that packets arrive to a given queue according to a Poisson process with arrival rate λ. The aggregation process starts when the first segment in a slot arrives. The number of further segments i arriving within time τ is given by a Poisson distribution with distribution function:

$$p_i = \frac{(\lambda\tau)^i}{i!} e^{-\lambda\tau} \quad , \qquad i = 0,1,2.....$$

The Poisson process exhibits the following property, which is utilised to find the mean delay: If i segments arrive during the interval τ, then the arrival times are uniformly distributed across the interval τ, and the average distance between two consecutive segments (or first/last segment and the start/end of interval) is $\tau/(i+1)$ [48].

If $(k-1)$ segments or more are observed inside the interval τ, then the optical slot is transmitted before timeout. Otherwise, the optical slot is transmitted at the end of the interval and contains some empty slots.

If i, $i < k-1$ segments are received within the interval τ, then the packet initiating the interval is delayed τ and the i segments inside the interval are on average delayed $\tau/2$. In total $i+1$ segments are transmitted. The average waiting time is given by:

$$w_i = \frac{\tau + \tau/2 \cdot i}{i+1}, \qquad i < k-1$$

If i, $i \geq k-1$ segments are received within the interval τ, the average distance between the segments in the optical packet is $\tau/(i+1)$, and the optical packet is on average transmitted after $(k-1)\,\tau/(i+1)$ time units. The average waiting time in this case is given by:

$$w_i = \frac{(k-1)\cdot\tau}{2\cdot(i+1)}, \qquad i \geq k-1$$

The average waiting time due to the aggregation process can now be determined:

$$W_1 = \frac{total\ waiting\ time}{number\ of\ packets}$$

$$W_1 = \frac{\sum_{i=0}^{k-2}(i+1)w_i p_i + \sum_{i=k-1}^{\infty} k w_i p_i}{\sum_{i=0}^{k-2}(i+1)p_i + \sum_{i=k-1}^{\infty} k p_i}$$

It is possible to find a simple approximation for the waiting time above when τ is large ($\tau >> k/\lambda$). The optical slot is then aggregated when it contains k segments. The arrival time for segment nr k is given by an Erlang-(k-1) distribution with mean value $(k\text{-}1)/\lambda$. The average segment delay thus becomes:

$$W_1 = \frac{k-1}{2\lambda}, \qquad \tau >> \frac{k}{\lambda}$$

This expression is useful as a verification of the exact expression, which can be evaluated for large τ values and compared to the asymptotic value.

The second delay contribution is the time it takes from an optical packet is ready for transmission until it is actually transmitted. It is assumed that

the number of queues Q is sufficiently high such that the sum of events can be modelled as a Poisson process. The system considered is then a M/D/1 queuing system with average waiting time:

$$W_2 = \frac{\rho h}{2(1-\rho)}$$

where ρ is the load and h is the service time of an optical slot. In order to determine the load, the average number of empty segments in an optical slot must be calculated:

$$n_e = \sum_{i=0}^{k-2} (k-i-1) \cdot p_i$$

The load ρ is now expressed by the load of incoming segments to a single queue ρ_λ:

$$\rho = Q \cdot \rho_\lambda \cdot \frac{k}{k-n_e}, \qquad \rho_\lambda = \lambda \cdot \frac{h}{k}$$

The last term expresses the load increase due to inefficient filling of optical slots. The total delay is

$$W = W_1 + W_2$$

Note that the first term is an increasing function of τ, whereas the last term is a decreasing function of τ. The total average delay is therefore expected to have a minimum value for an optimum value of τ.

3.2.3. Performance Evaluation

In this section, the derived expression for the total average delay will be used to dimension the timeout parameter τ. The expression for aggregation delay contains unbounded summations. They can, however, be converted to bounded summations in the following way:

$$\sum_{i=k-1}^{\infty} p_i = 1 - \sum_{i=0}^{k-2} p_i \ ,$$

$$\sum_{i=k-1}^{\infty} w_i \cdot p_i = \frac{\tau}{2}\frac{k-1}{\lambda\tau}\sum_{i=k-1}^{\infty}\frac{(\lambda\tau)^{i+1}}{(i+1)!}e^{-\lambda\tau} =$$

$$\frac{\tau}{2}\frac{k-1}{\lambda\tau}\sum_{j=k}^{\infty}\frac{(\lambda\tau)^{j}}{j!}e^{-\lambda\tau} = \frac{k-1}{2\lambda}\left(1-\sum_{j=0}^{k-1} p_j\right)$$

In the following, an arrival rate λ=0.24 is taken as an example. The optical slot can hold up to 16 segments, i.e. *k*=16. The time unit is set to the duration of an optical slot, and the number of queues is set to 50, that is *h=1* and *Q=50*. The load pr. queue is

$$\rho_\lambda = \lambda \cdot \frac{h}{k} = 0{,}015$$

The total load offered to the system is thus 50*0.015 = 0.75 = 75 %. Figure 12 shows the total average delay *W* as a function of the timeout parameter τ.

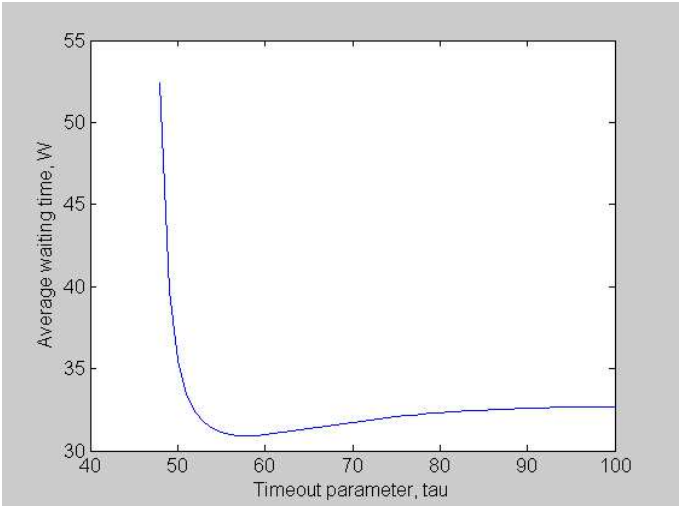

Figure 12: Average Waiting Time *W*(τ). The time unit is *h* (the duration of an optical slot)

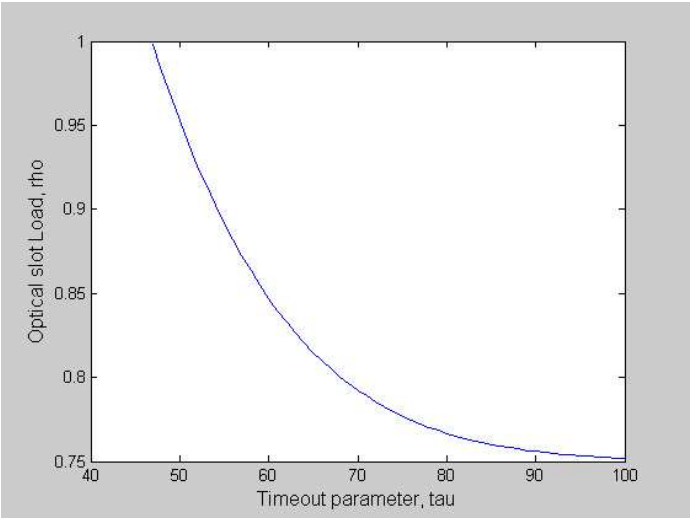

Figure 13: Optical Slot Load as a function of τ. The time unit is *h* (the duration of an optical slot)

There is a minimum possible value of τ =47. For small values of τ, the queuing delay dominates because the load of optical slots increases. Figure 13 shows the load of optical slots as a function of τ. For τ =47, the load is approximately 100 %. For higher values of τ, the delay is dominated by the aggregation delay, and the load decreases to 75 %, equal to the offered load.

The average waiting time W has a global minimum at τ = 58 according to Figure 12. The corresponding load is 86 % according to Figure 13. Reducing the load might be a benefit; otherwise additional delay may be introduced at later stages in the network. Now, the average arrival time for the last segment is $(k\text{-}1)/\lambda$. By setting $\tau = k/\lambda$, it is ensured that on average all segments have arrived before timeout. In this case τ = 16/0.24=66.6 giving a load of 80.6 % according to Figure 13. The load increase is determined by the expression:

$$\frac{k}{k-n_e}=1.075$$

which does not depend on λ when $\tau = k/\lambda$. In this case, a general load increase of 7.5 % is obtained.

Having a τ value that depends on λ requires an adaptive mechanism that measures the load (inter-arrival time) for each queue, which is considered

feasible. Furthermore, a maximum allowed τ value should be defined, which is used for small load where $\tau = k/\lambda$ would lead to unacceptable high delay.

A more detailed investigation of the optical slot generation process can be found in [49][50] that include simulation results from OPNET modeler. The simulation model calculates the overall delay including the segmentation and reassembly delay of client layer IP packets and can handle individual timeout values for the different queues. The analytical results presented here have been utilised to validate the simulation model.

3.3. Wide Area Network (WAN)

The Wide Area Network proposed in the DAVID project was designed to fulfil several objectives; the architecture should encompass several technologies including electrical packet switching, optical packet switching and switching at the fibre level in order to cope with various bit rates and different levels of aggregation. Furthermore, the aim is to provide a unified control and management solution that covers all technologies. In this respect, MPLS is a promising solution because of the separation between routing and forwarding allowing support for a wide range of technologies.

The following subsections contain more detailed description of the DAVID WAN. The focus is on the different technologies and their organisation into hierarchies. The work is based on [51][52][53].

3.3.1. Network topology

Figure 14 depicts a general packet switched network with several levels of hierarchy. In agreement with MPLS, the packet switch is called a "Label Switch Router" or LSR. The hierarchical structure allows for topology aggregation; the topology information from each level of hierarchy is aggregated and sent towards the next lower level of hierarchy. That is, the LSR network at level (N+1) appears as e.g. a single LSR at level (N). Topology aggregation increases routing scalability with the cost of less optimal routing decisions.

It is convenient to reflect the network hierarchy in the MPLS label distribution process. At each level of hierarchy, the label is swapped in the LSRs. A new label is pushed on the label stack when the packet leaves

level (N) and enters level (N+1). This label is popped off the stack when the packet re-enters level (N).

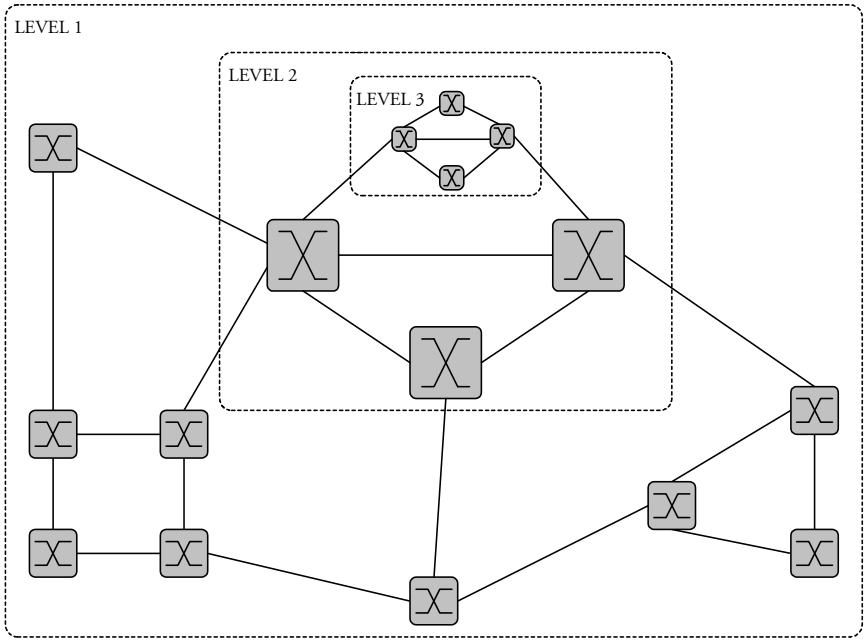

Figure 14: Network hierarchy

3.3.2. MPLS label stack and traffic aggregation

As described in section 3.3.1, each level in the label stack corresponds to the same level in the network hierarchy. It is therefore possible to aggregate traffic from level (N) going to the same destination in level (N+1). That is, a number of level (N) packets with identical level (N+1) labels are sent together in a bigger level (N+1) packet.

It is assumed that switching at each level of hierarchy is limited by a number of packets pr. second. The packet duration is almost constant at each level, but the size increases proportionally to the link bit rate.

Traffic aggregation and label stacking are depicted in Figure 15. This example shows a hierarchy with only 2 levels. Furthermore, only fixed length packets are considered. A discussion on fixed vs. variable length packets is given in the next section. The figure shows that level (N) packets destined for the same Level (N+1) destination are aggregated in the same Level (N+1) packet.

In DAVID, it is assumed that level (0) is comprised of electrical packet switches and level (1) contains optical packet switches. A LSR may participate in both level (0) and level (1) at the same time, and thus it will contain both optical and electrical label switch functionality.

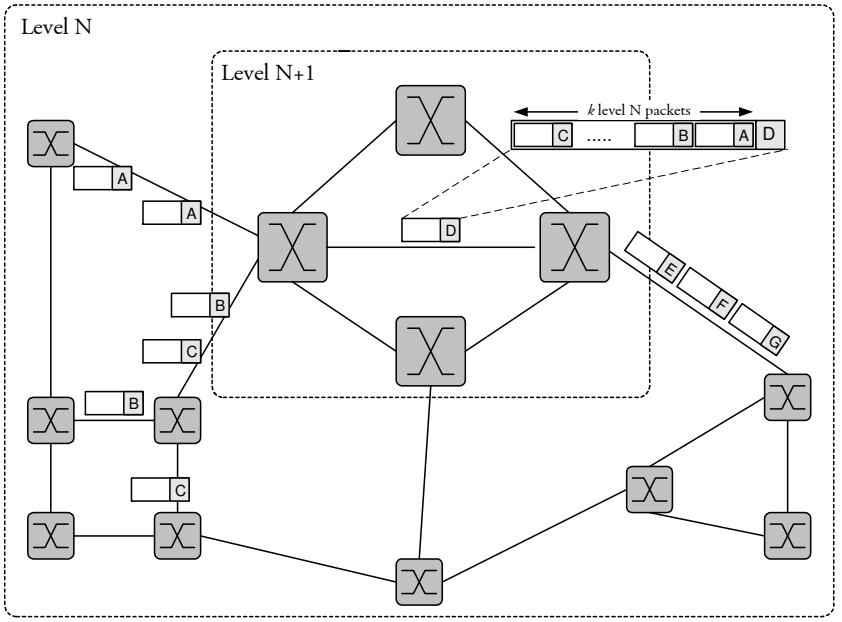

Figure 15: Label stack and traffic aggregation

3.3.3. Routing and Label distribution

Routing and label distribution are performed within each level of hierarchy. As an example, we will assume that the network carries IP traffic. Each LSR will participate in IP routing, including the optical label switches at level (1). Routing information from level (N+1) is aggregated and distributed to level (N). That is, all level (N+1) routers will be presented as a single router towards level (N). During the routing process, information from level (N) is input to the routing decisions at level (N+1). Route calculations are performed within level (N+1) and an aggregated routing table is determined. The aggregated routing table contains translation between address prefixes and destinations at level (N+1). (A destination at level (N+1) is a LSR with level (N) connectivity). Label distribution can be performed within each level following the ideas of distribution in a flat network (e.g. tag switching).

3.3.4. The DAVID hierarchy

The hierarchical level in the DAVID network is determined by the actual switching technology. This section provides a description of each hierarchical level given in Figure 14.

Level 1: This level is composed of ordinary IP/MPLS packet routers that perform electrical switching of packets (i.e. O/E/O conversion is performed). Electrical packet routers are used at the periphery of the WAN where the capacity is sufficient. The interface speed will vary, but will probably not exceed 10 Gbit/s. The packet format is important because of the impact on the aggregation process shown in Figure 15.

Fixed length packets ease the aggregation into optical slots, but introduce a potential scalability issue; a desired but not mandatory feature of an MPLS LSR with fixed length packets is the ability to handle multipoint to point LSPs. This means that several flows can be merged into one single flow with one label. If flow merging is not supported then packets from each flow may be interleaved, and it is necessary to assign a distinct label to each flow in order to distinguish the flows in the egress LSR. To support flow merging, the LSR must be able to buffer packets until all packet segments have arrived. Flow merging solves a scalability issue regarding label assignments. Consider a network with N edge LSRs. In case of flow merging, we need only O(N) different labels to address each edge device. Without flow merging, the number may increase to O(N*(N-1)) because each ingress node must establish a LSP to all other edge LSRs.

Variable length packets require a segmentation function at the optical interface. The segments carried in the optical slots must then be reassembled when they leave the optical domain, but the reassembly function is also necessary in case of fixed length packets with flow merging.

Level 2: Optical packet switching is employed at this level, and the payload is switched transparently to provide high throughput. Packet forwarding is based on label swapping (in line with MPLS), and the wavelength dimension is used to solve contention by assigning different wavelengths to packets destined for the same output fibre. The minimum packet duration is among other things dependent on the packet header processing time; the scheduling algorithm must examine all packet destinations every timeslot to determine, whether packets are forwarded or transmitted to the buffer in case of output contention. The high throughput is partially obtained at the cost of decreased granularity, therefore, packets from level 1 is aggregated (bundled) into layer 2

packets as shown previously in Figure 15. The bit rate at the optical level is higher compared to the electrical level, e.g. 40 Gbit/s.

Level 3: The switching granularity at level 3 will be lower than that at level 2. Since the optical packet switches at level 2 do not distinguish between individual wavelengths on a fibre, the next lower level of switching granularity will be switching at the fibre level. The switch nodes at level 3 are thus optical crossconnects with the ability to interconnect fibres (e.g.. based on MEMS technology). A wavelength or waveband routed layer 3 is in principle feasible, but requires limitations imposed on the wavelength conversion in the OPS nodes, reducing their ability to solve contention.

3.3.5. OPS network node

The basic structure of the DAVID broadcast and select switch was shown in Figure 2 in Chapter 2. The switch has M input ports each carrying N wavelengths. The product MN determines the total number of input streams, and thereby the total capacity. The total number of SOA gates is NM(N+M). Assuming that the capacity (NM) is fixed, the number of SOA gates is minimised for M=N. As an example, consider a switch with 8 ports and 32 wavelengths. Internally, the number of SOA gates is minimised by having M=16 ports with N=16 wavelengths. Thus, additional wavelength converters are required at the input to convert from external to internal wavelengths.

Contention resolution is performed by an optical Fibre Delay Line (FDL) re-ciculating loop buffer as shown in Figure 16. B wavelength ports on the switch matrix are connected to the buffer. Two different buffer configurations have been examined. The first configuration uses B fixed size fibres with the length equal to one optical slot. The second configuration uses multiple FDL lengths ranging from 1 to B times the optical slot length. The scheduling algorithm for the first approach is quite simple. Firstly, for each output fibre of the OPR, elect at most (M-B) packets to be forwarded directly. Secondly, from the remaining packets, elect at most B to be put in the buffer. Any other packets will be lost. Election of packets to be forwarded and buffered is based on the packet age, that is, the time it has spent in the buffer already. Old packets have the highest priority.

The scheduling algorithm for the second approach is more complex because the B buffer ports are not equivalent. Four different buffering strategies were proposed in the context of the DAVID project [54]:

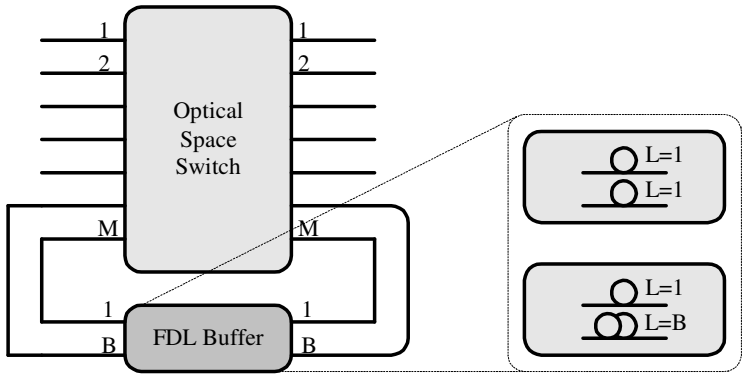

Figure 16: OPR with re-ciculating FDL buffer. Two buffer configurations are shown

MinDelay: For each packet entering the buffer, the free buffer port with smallest corresponding FDL length is chosen.

NoOvr: To buffer packet P, take the FDL with smallest length L such that no more than B-M packets will leave the buffer at now+L for the same output fibre of the OPR; otherwise drop the packet.

AvoidOvr: First seek the free port with smallest FDL length that would not cause overload; enter the packet at the free port with the smallest FDL length if no such port can be found.

Balance: Contending packets are spread in time. To buffer a packet P count (N) for each available FDL length L, the packets scheduled at now+L for the same output port destination as P. The packet is then put in the free FDL with the smallest count N.

The buffering strategies were compared by a simulation study [54]. In conclusion, the balance strategy outperforms the others for Poisson and bursty GeoOnOff models. For heavy tail ParetoOnOff traffic, no significant reduction of PLR can be achieved by any buffering strategy. The burstiness of the core network traffic is limited because the traffic is aggregated from many client layer traffic streams, so even with a high degree of self-similarity it is not expected to impact the performance significantly.

3.4. Power Benchmarking

The objective of the benchmarking study presented here is to compare the power consumption of electrical packet switching and optical packet switching. The total aggregate capacity is 2.56 Tbit/s for both packet switches, and 10 Gbit/s port speed is assumed.

The first part covers the electrical packet switch (EPS). The power consumption of the different buildings blocks (shown in Figure 3 in chapter 2) is calculated/estimated separately. The second part summarises the power consumption results for an optical packet switch employing the broadcast and select architecture used in DAVID.

At this point, it should be noted that the two systems compared here are quite different with respect to functionality and features, and the comparison might look irrelevant. However, the goal is to demonstrate that optical packet switching, even with the lower level of functionality, might be the preferred solution, e.g. due to reduced power consumption.

3.4.1. Electrical Switch Fabric

The objective is to determine the approximate power consumption of a 2.5 Tbit/s electrical packet switch fabric. The central fabric considered in this section does not include port cards and O/E/O modules. They are included in section 3.4.2. The fabric design considered in this study is shown in Figure 17.

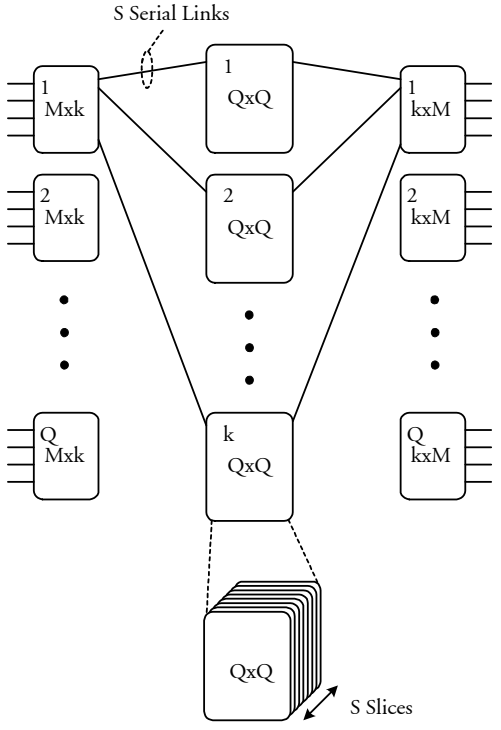

Figure 17: Switch fabric

The Clos fabric is composed of *k QxQ* central switch elements, consisting of S identical slices, and additionally Q *Mxk* and Q *kxM* input and output elements. In the following, these elements are denoted IO modules, and are connected to the central switch elements by S high speed serial interconnects.

In order to realise a 256x256 fabric with 10 Gbit/s interface speed, the following parameters are proposed for the switch fabric in Figure 17: Since *Q=256/M*, if *Q=64,* then *M=4.* A speedup of 1.5 is assumed, that is, *k=1.5*M = 6.* This value is close to the value for a clos network used in [25]. The speedup compensates for a non-ideal routing/load-balancing function across the middle stage elements and possibly backpressure due to limited internal buffer resources. The bit rate of high speed serial interconnects is set to 2,5 Gbit/s. It is assumed that eight 2,5 Gbit/s interconnects are required to carry 10 Gbit/s due to internal overhead, additional speedup and line coding. This gives the number of slices: S=8. The table below summarises the switch parameters:

Table 1: Switch parameters

VARIABLE	EXPLANATION	VALUE
N	Size of Switch *(NxN)*	256
M	Number of Inputs on IO module	4
k	Number of outputs on IO module. Also equal to number of centre stages	6
Q	Size of central switch *(QxQ) Q=N/M.* Also equal to number of IO modules.	64
S	Number of Slices. Also equal to number of serial interconnects in a logical interconnect.	8

The power consumption of the described switch fabric is the sum of the IO module cores, the central switch slice cores and serial interconnects, SerDes. Table 2 shows estimated power consumption values and count for these components. The estimation of core chip power consumption is based on the IBM datasheet [57]: The Q-64 Gbit/s packet switch with 32 SerDes consumes approximately 20 W. It is assumed that half the power (i.e. 10 W) is consumed by the SerDes macros and half by the central switch slice. The power used by one SerDes macro is thus 0,3 W (10 W/32). For a 64x64 switch slice, the power consumption is assumed to be 4-5 times larger than that of the IBM (a 64x64 crossbar is 4 times larger

than a 32x32 crossbar) here taken as 50 W. The core power of each IO module is assumed to be a little less, i.e. 40 W.

Table 2: Power consumption and count of the switch fabric building blocks

VARIABLE	EXPLANATION	VALUE	NUMBER
$P_{Core,IO}$	Power consumption in IO chip core.	40 W	Q
$P_{Core,CS}$	Power consumption in centre stage chip core.	50 W	$S \cdot k$
P_{SerDes}	Power consumption of SedDes module on chip.	0.3 W	$2 \cdot Q \cdot S \cdot k + Q \cdot S \cdot M$

Now, the total power consumption of the switch fabric can be computed:

$$P = Q \cdot P_{Core,IO} + S \cdot k \cdot P_{Core,CS} + (2 \cdot Q \cdot S \cdot k + Q \cdot S \cdot M) \cdot P_{SerDes}$$

The result is shown in Table 3.

Table 3: Total power values of the switch fabric

BUILDING BLOCK	VARIABLE	POWER
IO modules	$Q \cdot P_{Core,IO}$	2560 W
Central switch elements	$S \cdot k \cdot P_{Core,CS}$	2400 W
Interconnects	$(2 \cdot Q \cdot S \cdot k + Q \cdot S \cdot M) \cdot P_{SerDes}$	2700 W
Total switch fabric Power	P	7660 W

3.4.2. Traffic Manager and Phy. IO

It is assumed that the traffic manager board is equipped with a 10 Gbit/s network processor and memory devices. The Intel IXP2800 10 Gbit/s network processor consumes 30 W, worst-case [58]. Furthermore, with external memories and additional devices, it adds up to approximately 40 W. We assume an EPS for a core network application, hence the need for transmitters and receivers compatible with long reach. According to [60], the power consumption in a 10 Gbit/s optical receiver and transmitter, for a long distance system with external modulator, is 9 W. For comparison, a 10 Gbit/s Ethernet transceiver for Extended Range

[59], with a directly modulated laser and maximum distance of 40 kilometres, consumes 6 W.

The total consumption of O/E/O and traffic manager board is thus 50 W. Having 256 boards, the power adds up to 12.8 kW. (10240 W from traffic manager and 2560 W from O/E/O conversion)

3.4.3. Total Power Consumption

The total power consumption is shown in Table 4.

Table 4: Total power consumption of EPS

DEVICE	POWER
switch fabric	7.7 kW
traffic manager	10.2 kW
O/E/O	2.6 kW
TOTAL	20.5 kW

The power consumption impacts the physical dimensions of the switch; a typical rule of thumb is a maximum allowed power dissipation of 2 kW in a single rack. The traffic manager board including O/E/O conversion consumes 50 W, allowing for up to 2 kW / 50 W = 40 boards in a single rack. This number indicates that the power consumption does not limit the number of Traffic manager cards in one rack. Several racks are needed for the traffic manager cards (6-7), but the number of cards pr rack is limited by physical dimensions and not power consumption, On the other hand, the switch fabric must be divided across four racks, as the fabric dissipates almost 8 kW. The total number of switch chips in the fabric shown in Figure 17 is (Q + S*k). The average number of chips per rack is thus $(Q + S*k)/4 = 28$. The number of interconnections between two specific racks is approximately $(2QSK)/4 = 256$, and each rack will thus require approximately 1000 high speed serial link connections routed to other racks leading to a very complex design.

3.4.4. Optical packet switch

A detailed investigation of the various contributions to power dissipation in an optical packet switch is given in [61]. The main results are presented in Table 5.

Table 5: Power consumption in optical packet switch

	KW
Input Interface	2.44
Output Interface	2.31
Control Unit	2.50
Switch Matrix	0.84

The Input Interface includes 3R regeneration and O/E conversion with CDR at the header bit rate (2.5 Gbit /s). The output interface is quite similar to the input interface, e.g. 3R regeneration. Furthermore, header erasure and rewriting are performed. The switch matrix contains SOA gates in the space selection stage and EDFA's to compensate for splitting loss in the broadcast stage. Finally, the control unit performs scheduling of packets, and it is assumed to be fully implemented in electronics.

3.4.5. Summary and comparison

Table 6 summarises the estimated values of power consumption in EPS and OPS. This study suggests that power consumption may be significantly reduced for OPS compared to EPS. Even if stand-alone header erasure/reinsertion and synchronisation would be required and amount to a couple of kW, the total power consumption would still be around half of that of an EPS.

Table 6: Power consumption EPS and OPS

	EPS (kW)	OPS (kW)
Interfaces	2.6	4.8
Switch Matrix	7.7	0.8
TMB / Control Unit	10.2	2.5
SUM	**20.5**	**8.1**

A main difference lies in the traffic manager/control unit. The sum of all the traffic managers power consumption on each input board seems to consume far more power than the control unit for an OPS.

A major constraint for integration of an EPS is the power dissipation in the switch fabric. This is mainly a problem because the switch fabric chips are heavily interconnected. In fact, a single-rack can typically

dissipate around 2 kW of power since the fan speed is in practice limited by the system's mechanical stability. This target can be met by OPS switch matrix, enabling implementation in a single rack, thereby solving the severe problem of rack-to-rack interconnection. It is therefore expected that optical technology will play a key role in future terabit switch systems, and that the first step will be the introduction of optics in the switch core. Integrating optical switch matrix with O/E interfaces will probably benefit from using a packet format such as assumed for OPS.

3.5. Summary

In this chapter, the European research project DAVID was presented. The project studies feasible architectures for optical packet switching, both in MAN and WAN. The first part described the MAN part of the network with focus on the ring node. The segment bundling process forming optical slots was investigated. A statistical model of the bundling operation that also takes a timeout into account was presented. The derived analytical result for the average waiting time was exploited to determine an optimal value of the time-out parameter.

The second part of this chapter introduced the hierarchical DAVID WAN network. The hierarchies were formed by different technologies, and the concept of MPLS was utilised to create a unified switching / routing approach covering all layers. Finally, the OPS node was described with focus on buffering strategies.

The power comparison performed in the benchmarking study shows that the power consumption limits the level of integration possible in the electrical domain. On the other hand, the optical switch fabric power consumption is sufficiently low to allow a smaller and more manageable design.

4. QoS and Traffic Engineering

This chapter reviews principles for balancing Quality of Service (QoS) and utilisation in packet switched networks, exemplified by Multi Protocol Label Switching (MPLS) networks. QoS can only be guaranteed by giving priority, e.g. by reservation of a certain bandwidth. High utilisation can be obtained by proper traffic engineering, i.e. cost efficient routing of traffic streams. By combining the two adverse principles, one may at the same time obtain a high utilisation and guarantee the end-to-end QoS. This chapter is based on [63].

4.1. Introduction

The objective of traffic engineering is to obtain a high utilisation of the network infrastructure for a given quality of service. It is important for service providers to have a high usage of transmission capacity and to avoid that certain parts of the network are congested, while other parts are under-utilised. Multi Protocol Label Switching (MPLS) provides a framework for traffic engineering with the introduction of Label Switched Paths. They enable establishment of virtual circuits across a packet switched network. MPLS means multi-protocol both at layer 2 (link) and layer 3 (network). However, the main focus is on IP at layer 3, and MPLS is therefore a tool to provide traffic engineering capabilities in the IP backbone network. This chapter reviews principles for MPLS traffic engineering. QoS is discussed in section 4.2. The path selection/routing problem is discussed in Section 4.3. Routing and traffic engineering can be done off-line or on-line. Off-line routing requires knowledge of all the demands in the network, whereas on-line routing determines a path on demand. Section 4.3.1 briefly discusses the methods for path establishment. Section 4.3.3 provides a framework for traffic engineering in a TCP/IP based network. The basic idea is to group TCP connections into trunks, and then route and establish the trunks using MPLS traffic engineering procedures. Constraint Based Routing is the topic of section 4.3.4. This section introduces a bandwidth dependent link weight that can improve the network utilisation.

4.2. QoS

This section reviews the basic mechanisms for QoS provisioning in IP/MPLS networks. QoS in IP is supported by Integrated Services (Intserv) [65] and Differentiated Services (Diffserv) [64]. QoS can only be guaranteed by reserving resources end-to-end by some mechanism. On the other hand, for variable packet traffic a high utilisation can only be obtained by exploiting statistical multiplexing, i.e. by sharing resources. By reserving a certain minimum capacity, we are able to guarantee a flow a certain QoS and by restricting the maximum capacity, we protect the other flows against overload. By proper traffic engineering [70], we are able to guarantee a certain QoS and at the same time ensure a high utilisation.

4.2.1. Integrated services

Intserv makes use of the Resource Reservation Protocol (RSVP) to establish QoS connections. Three different service classes are supported:

Guaranteed Service, Controlled Load and Best Effort.

Guaranteed Service [72] requires that each node contains schedulers that can allocate a minimum amount to each flow, e.g. Packet Generalised Processor Sharing [62], Weighted Fair Queuing or similar approximations to Generalised Processor Sharing. The traffic is described by a flow spec containing Rspec (R for reserve) and Tspec (T for traffic). The Tspec describes the traffic profile by a rate and a bucket size, and the Rspec specifies the amount of bandwidth, which should be reserved in each node. Each node is responsible for reserving sufficient buffer space to avoid packet loss. It can be shown that the end-to-end delay is upper-bounded, and the bound is given in [72].

The Controlled Load service [71] is between Guaranteed Service and Best Effort. Controlled Load service should make the network appear as lightly loaded for the application, but does not guarantee zero packet loss and upper-bounded delay.

Intserv can easily be integrated with MPLS; the RSVP protocol is used by MPLS to perform path establishment so at the same time, it can reserve resources and set up the Label Switched Path.

IntServ with RSVP signalling requires microflow state handling and soft state signalling at each hop adding significant complexity to the network. The IntServ framework is only implemented in a limited number of

networks, and the alternative approach Diffserv has appeared as a more viable solution.

4.2.2. Differentiated services

The complexity of Intserv with RSVP resulted in the definition of the Differentiated Services (Diffserv) architecture [64]. Diffserv divides the traffic into so-called Behaviour Aggregates (BA), one for each Class of Service (CoS). Three main CoS classes are defined: Expedited Forwarding (EF), Assured Forwarding (AF) and Best Effort (BE).

The Diffserv approach is also compatible with MPLS. In Diffserv, the service that a packet will receive is determined from the Differentiated Service Code Point, DSCP, in the packet header. This information can easily be encoded in the MPLS labels [74]. Furthermore, the IETF traffic engineering working group has work in progress on Diffserv aware MPLS traffic engineering [75].

4.3. Traffic Engineering

MPLS provides mechanisms for traffic engineering because of the support of explicit routed Label Switched Paths (LSPs). The creation of LSPs requires two steps; the first step is 'path selection', and the second one is 'path establishment'. Path selection is covered in subsection 4.3.1, and path establishment is covered in subsection 4.3.2. Subsection 4.3.3 on TCP traffic engineering proposes a scheme for path selection in best effort TCP/IP networks. Finally, subsection 4.3.4 discusses Constraint Based Routing (CBR). Constraint Based Routing is used in conjunction with path selection to determine a path under certain (e.g. bandwidth) constraints.

4.3.1. Path selection

Path selection can be performed either off-line or on-line. Off-line path selection requires knowledge of the traffic matrix describing the demands between all edges LSRs in the MPLS domain.

According to RFC 2676 [68] the MPLS network is described by a graph G = (V,E,c) where V is the set of Vertices (LSRs) and E is the set of Edges (Links). The capacities and constraints are denoted c. The demands (i.e. the traffic matrix) are described by a graph H = (U,F,d).

U is a set of edge LSRs that originates or terminates an LSP. The edges in H are F, which represents an LSP between two nodes in U. The parameter d is the set of demands and restrictions associated with F. H is named the induced MPLS graph. The objective of traffic engineering is to map the induced graph H to the topology graph G.

4.3.1.1. Off-line traffic engineering

In this study, it is assumed that H is given, and the task is to map H to G. This is a well-known optimisation problem. Typically, the problem can be formulated as a Linear Programme (LP), and LP solvers can be applied to find the LSP routes that maximise or minimise the objective function. As an example, the objective function could be to minimise the maximum load on any link or to maximise the total revenue. An Integer Linear Program (ILP) formulation of MPLS traffic engineering can be found in [66]. This section demonstrates that the ILP formulation rather easily can be converted into programme code for an optimisation tool. As an example, this ILP formulation has been converted into Optimisation Programming Language (OPL) developed by ILOG [66]. The first part of the OPL code is the network and traffic matrix definition:

```
int nbLinks = ...;
int nbFlows = ...;
int nbNodes = ...;

range Links  1..nbLinks;
range Flows  1..nbFlows;
range Nodes  1..nbNodes;

// Topology of Network:
int+   u[Links] = ...;
int+   v[Links] = ...;
float+ linkbw[Links] = ...;
float+ cost[Links] = ...;

// Flows:
float+ effbw[Flows] = ...;
int+   s[Flows] = ...;
int+   d[Flows] = ...;
float+ h[Flows] = ...;

var int x[Flows,Links] in 0..1;
```

The values of variables, network topology and flows are defined in an input file. Each flow is defined by effective bandwidth, a source *s* and destination *d* and a maximum hop count *h*. Each link in the network has a cost associated. The last line defines the variable *x* that will contain the routing of flows after optimisation. Note that *x* is an integer, and the problem is thus an integer linear program. The next part of the program defines the optimisation task:

```
Minimize
   sum(l in Links)
      cost[l] * ( sum(i in Flows) effbw[i] * x[i,l] )

subject to
{
   forall(l in Links) // (1)
      sum(i in Flows) effbw[i] * x[i,l] <= linkbw[l];

   forall(i in Flows) // (2)
      sum(l in Links) x[i,l] <= h[i];

   forall(n in Nodes) // (3)
      forall(i in Flows : s[i] = n)
         sum(l in Links : u[l] = n) x[i,l] = 1;

   forall(n in Nodes) // (4)
      forall(i in Flows : d[i] = n)
         sum(l in Links : v[l] = n) x[i,l] = 1;

   forall(n in Nodes) // (5)
      forall(i in Flows : s[i] <> n & d[i] <> n)
         sum(l in Links : u[l] = n) x[i,l] =
         sum(l in Links : v[l] = n) x[i,l];
};
```

The objective is to minimise the total cost subject to different constraints. Each constraint is described in the following: Constraint (1) ensures that link capacity is not exceeded. Constraint (2) puts a maximum on the number of hops in the path of a given flow. Constraints (3) and (4) bind the flows to their source and destination, respectively. Finally, constraint (5) ensures that a flow, which enters a node, will also leave that node as long as the node is neither source nor destination for that flow.

The ILP problem is NP complete, which means that it cannot be solved by any known polynomial time algorithm. However, heuristics are available (and e.g. implemented in the ILOG solver) that can make the problem tractable even for larger networks.

4.3.1.2. On-line traffic engineering

It is now assumed that the induced MPLS graph H is not known beforehand. Connection requests will arrive one after another and a suitable path must be determined for each request. In general, the flows must be routed subject to QoS constraints. This routing procedure is referred to as Constraint Based Routing (CBR). RFC~2676 [68] describes extensions to OSPF to support Constraint Based Routing. OSPF utilises Dijkstra's algorithm to calculate the shortest path with respect to a given metric (e.g. hop count). When a connection request arrives, the requested bandwidth is compared to the remaining bandwidth of each link. If the available bandwidth on a given link is insufficient, the link is pruned from the graph representing the network. Dijkstra's algorithm is then applied to the resulting graph.

This method requires a computation of Dijkstra's algorithm for each connection request. To avoid this, a pre-computation may be performed. This is complicated with Dijkstra's algorithm because the requested bandwidth is unknown beforehand. The Bellman-Ford algorithm is better suited for this task because the iteration parameter is the hop count. Dijkstra's algorithm finds the shortest path from a given node to all other nodes by developing the paths in order of increasing path length. The Bellman-Ford algorithm takes a different approach: First it finds the shortest path subject to the constraint that the path contains at most one link, then it finds the shortest path with the constraint that the path contains at most two links, and so on. It is assumed that the cost of a path is equal to the hop count, that is, each link is assigned the cost of one. The extension to Bellman-Ford [68] is explained by the graph shown in Figure 18. A general description of Dijkstra's and Bellman-Ford's algorithms is given in [76].

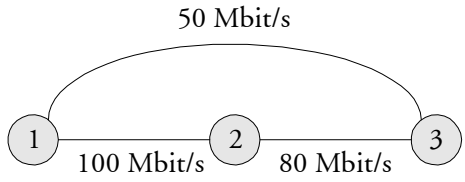

Figure 18: Simple network graph

The task is to find the shortest path from node 1 to node 2 and 3. In the first step with hop count equal to one, the shortest path to node 2 is the direct path 1-2 with 100 Mbit/s of bandwidth. The shortest path to node 3 is 1-3 with 50 Mbit/s bandwidth. In the second step of the algorithm (hop count = 2), the shortest path with a maximum distance of two hops is determined. In the original Bellman-Ford algorithm, the shortest path from 1 to 3 is already determined, however, it is noticed that node 3 can be reached by the path 1-2-3 with a minimum bandwidth of 80 Mbit/s. In this way, the maximal supported bandwidth between two nodes can be determined as a function of hop count. When the connection request arrives, it is easy to determine the path that with minimal hop count is able to support the requested bandwidth.

The routing protocol must distribute topology and resource information at regular intervals. With best effort IP routing, the topology information is only distributed in case of changes to the topology. However, in case of Constraint Based Routing, the resource information from a given link must be distributed when the link utilisation changes over a given threshold. A low threshold value will result in a more accurate view of the resources, but the network load from routing information will become higher.

4.3.2. Path Establishment

The path selection procedure must be followed by a path establishment procedure. In case of best effort IP traffic, the Label Distribution Protocol (LDP) can be used to bind labels to each prefix in the routing table. Incoming labels can be assigned by a local binding and outgoing labels by a remote binding from a downstream node. Typically, LDP works with local control between neighbour nodes. However, in the case of Constraint Based Routing where the path is calculated by the ingress router, there is a need for a protocol that can establish a path across the network.

Currently, two different protocols are standardised: TE-RSVP [28] and CR-LDP [27]. The main difference between these protocols is that RSVP is soft state, while CR-LDP is hard state. As a soft state protocol, RSVP requires periodic transmission of refresh messages; otherwise the connection will be deleted. The soft state operation introduces a scalability problem since the number of refresh messages in proportional to the number of flows.

4.3.3. TCP traffic engineering

The most common transport protocol for IP networks is TCP. It is used for World Wide Web access and file transfers via FTP. Today TCP/IP delivers only best effort transport, and network congestion is handled by the TCP congestion control mechanism. In general, an IP network utilises shortest path routing protocols such as OSPF. A drawback of this approach is that congestion can occur at the least cost path, while other longer paths are under-utilised. Two hosts that communicate over TCP establish a TCP connection before exchanging data. A TCP connection is identified by source and destination IP addresses and source and destination TCP port numbers. The basic idea behind TCP traffic engineering is to assign Label Switched Paths, LSPs, to the TCP connections. This means that, when a connection request arrives, an LSP is established. The LSP route is determined by the available capacity on the links in the network. However, the number of TCP connections (and connection requests) is too large to make the solution scalable. Instead, a number of TCP connections can be grouped together to form a traffic trunk, and only one LSP is established for that traffic trunk. Of course, the TCP connections in a traffic trunk share source and destination edge LSRs.

In the following, the operation of the TCP traffic engineering scheme will be described in more detail. TCP connection requests arrive at the ingress LSR. The destination address determines the egress LSR. N denotes the maximum number of TCP connections that will be carried within one specific traffic trunk. If N connection requests already have arrived for a trunk, then the next request will be allocated to a new trunk. The LSP for the new trunk can be determined and established during arrivals to the previous trunk. When all N connections within a trunk have been released, the trunk is no longer in use, and the associated LSP is released.

The resources that a TCP connection requires in terms of bandwidth are unknown. The TCP protocol will adjust to the maximum transfer rate that can be supported by the network without packet loss. Routing of new trunks can therefore only be based on the number of trunks routed across the various links.

The link load is defined as the number of trunks on the link divided by the link capacity. The objective of the routing mechanism is then to determine the path between an ingress LSR and an egress LSR that has the minimal value of the maximum load of the links belonging to the path.

Flooding of topology information and link utilisation and routing of trunks can be achieved by a routing protocol similar to OSPF. The LSP setup can be performed by standard MPLS protocols such as RSVP or CR-LDP. Flooding of topology information and LSP setup take up network resources, and the number of operations must be limited. The number of operations depends on N, i.e. the number of TCP connections in a trunk.

4.3.4. Constraint Based Routing

As stated in section 4.3.1.2, Constraint Based Routing (CBR) is used in conjunction with on-line traffic engineering. This section will examine the performance of Constraint Based Routing in more detail. It is assumed that incoming connection requests are routed based on Dijkstra's algorithm. Each connection request specifies a bandwidth requirement. The connection is accepted if the network is able to determine a path with sufficient resources, otherwise it is rejected.

The following simulation study will cover two different scenarios. In the first scenario, it is assumed that the link weights indicate the delay of the links, and the sum of weights along the links, which is the total delay, will be minimised by Dijkstra's algorithm. The total delay is composed of a transmission delay on the links and a queuing delay in the nodes. The queuing delay depends on the load on the link ρ. It is assumed that the queuing delay is given by the average waiting time for a M/D/1 queuing process [48]:

$$d_q = \frac{\rho \cdot h}{2(1-\rho)}$$

h is the packet duration (measured in seconds). In this study, the weight used in Dijkstra's algorithm is calculated from both link delay and queuing delay:

$$w = d_l + \alpha \cdot d_q = d_l + \alpha \cdot \frac{\rho \cdot h}{2(1-\rho)}$$

A new parameter α has been introduced in order to modify the weight of queuing delay compared to the weight of link delay.

To evaluate the efficiency of the CBR routing protocol, a simulation study has been performed. It is assumed that connection requests arrive with source and destination selected at random. The connections are not taken down. The number of successful connection attempts before the

first unsuccessful (blocked) connection attempt is counted. The network used in the simulation study is shown in Figure 19.

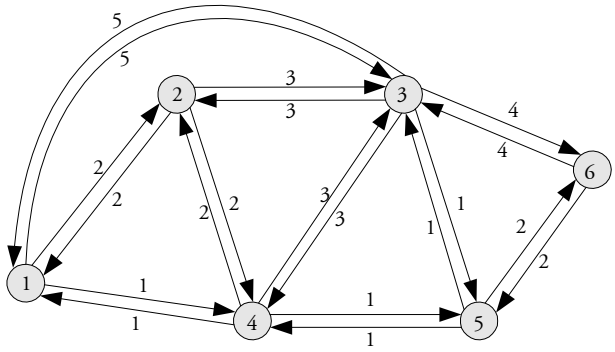

Figure 19: Network with link weights

The link bandwidth is set to 100 bandwidth units for all links. The arriving connections occupy 3 bandwidth units each. Figure 20 shows the number of accepted connections as a function of α. For each α value, 2000 experiments have been carried out in order to determine the mean number of connections. Two different cases are shown in Figure 20. In case A, the new connection is taken into account in the load calculation, contrary to case B where the link load is not adjusted with the new connection.

The benefit of taking the link load into account is clearly seen in Figure 20. The main reason is congestion at node 5 for small α values, because many 'shortest paths' traverse this node. On the other hand, with large α values, better utilisation of network resources is achieved, but the link weight might not represent the link delay anymore. Several experiments have been performed with different link delay weights for the network in Figure 19. The results show that the benefit of introducing the load in the weight calculation highly depends on the actual value of the delay weights, but in general the network utilisation will always be improved by taking the load into account. Case A yields better results than case B, and the difference will be even larger if each connection takes up a larger amount of the total link bandwidth. However, case B allows for path pre-computation using the Bellman-Ford algorithm instead of the Dijkstra algorithm as explained in 4.3.1.2.

Another approach for increasing utilisation for on-line traffic engineering is proposed in [73]. The concept is denoted 'Minimum interference

routing' and tries to minimize the interference from an arriving connection on future connections. The scheme requires calculation of min-cut links for all possible source and destination pairs each time a net connection arrives. The paper reports improvements in utilisation up to 25 %.

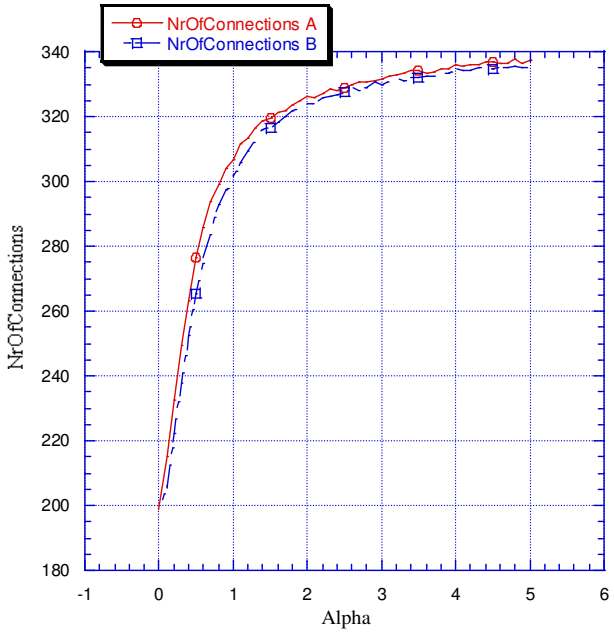

Figure 20: Number of accepted connections as a function of α

4.4. Summary

This chapter considers MPLS traffic engineering and the related protocols. The path selection procedure can be classified as either off-line or on-line. Off-line traffic engineering is used when the traffic demands are known beforehand, whereas on-line demands are routed at request time.

A traffic engineering scheme for best effort TCP/IP traffic was proposed. The basic idea is to group TCP connections into trunks and then route the trunks based on MPLS traffic engineering procedures.

Constrained based routing was examined in more detail. A heuristic formula for the link weight was shown to improve network utilisation. The link weight takes both transmission and queuing delay into account thereby increasing the cost of heavy loaded links.

5. Multipath Packet Switching

The basic concept of packet bundling is to group smaller packets into larger packets based on e.g. quality of service or destination within the packet switch. This chapter presents novel applications of bundling in packet switching. The larger packets created by bundling are utilised to extend switching capacity by use of parallel switch planes. During the bundling operation, packets will experience a delay that depends on the actual implementation of the bundling and scheduling scheme. Analytical results for delay bounds and buffer size requirements are presented for a specific scheduling algorithm and compared to simulation results.

5.1. Introduction

Networking technologies such as ATM, IP and MPLS have one thing in common, which is the need for packet switch fabrics within the switches or routers. The capacity of a packet switch node is often limited by the minimum packet size that is supports. If the packet size is sufficiently long then the switch fabric capacity is easily extendable by cutting the packet into slices that are switched over parallel planes. However, if the packet size is too long it is impossible to obtain an efficient filling.

High switching capacity can be achieved in multipath/ multistage switch systems where small switch units are interconnected to form a larger switch fabric. Banyan and Clos are examples of such interconnection networks [37]. A multistage fabric may have more than one route between each pair of inputs and outputs, and a routing function is then required [77].

This chapter presents another approach for scaling the switch capacity: As discussed above, increasing the packet length can increase switch capacity. This can be achieved by bundling a number of smaller packets into one larger packet at a higher bit rate. This principle is widely utilised within Time Division Multiplexed (TDM) networks, e.g. PDH and SDH/SONET. A number of lower order frames are multiplexed into one

higher order frame, for instance in SONET, which may group four STS-3 signals into one STS-12 frame. Thus, the frame length measured in seconds is identical for lower and higher order frames.

In this chapter, packet bundling within packet switching is considered in order to determine the feasibility of this concept. The chapter is based on work in [78]. Traffic bundling generally requires buffering and scheduling because packets are grouped together subject to specific constraints such as QoS class and destination. The switch fabric architecture and the concept of packet bundling are presented in section 5.2. In section 5.3, the queuing and scheduling issues related to packet bundling are considered. The bundling operation will delay packets, and a scheduling algorithm that can provide bounded delay is presented. Analytical expressions for the delay bound and maximum queue size are then derived. Section 5.4 presents simulation results in order to compare the delay bound with actual delay distributions for different traffic distributions and to compare different scheduling algorithms. Finally, in section 5.6, concluding remarks are given.

5.2. Switch Architecture

The principle of traffic bundling is illustrated in Figure 21.

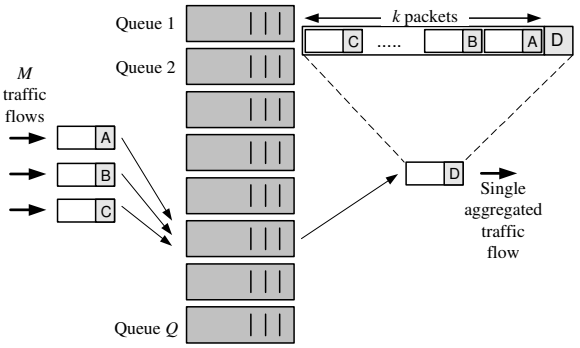

Figure 21: Traffic bundling unit

M incoming packet flows are aggregated into one outgoing flow. It is assumed that the packet size is fixed, and that the packet size (in bits) of the outgoing flow is *k* times that of the incoming flow. Thereby, the outgoing packet can hold *k* incoming packets as a maximum. It is also assumed that the duration of incoming and outgoing packets are identical, which implies that the bit rate of the outgoing packets is *k* times that of the incoming packets. Note that *k* must be greater than or

equal to *M*, otherwise it is impossible to operate the queues without packet loss.

This is due to the fact that if *M* packets arrive at each timeslot, then *M* packets must be removed on average. *k* must be greater than *M* in order to provide bounded delay because it may sometimes be necessary to transmit fewer than *M* packets. As compensation, more than *M* packets must be transmitted in some other timeslots.

The header of the incoming packet determines the destination queue in the bundling unit. The number of different queues is denoted *Q*. A specific queue can for instance be related to a specific destination and service class within a switch. Therefore, the outgoing packet contains only packets from one specific queue in each timeslot.

The switch architecture that employs packet bundling is shown in Figure 22. It is a $(M \cdot Q)x(M \cdot Q)$ switch comprised of *k* *Q*x*Q* switch elements and additional *M*x*k* and *k*x*M* input and output elements. The switch elements form a Clos-network. Note that *M* and *k* in Figure 22 correspond to *M* and *k* within Figure 21. The *Mxk* input elements perform the packet bundling. The scheme shown in Figure 21 needs to be modified slightly because the *k* packets are now sent across *k* parallel planes and not in one larger packet. This means that the aggregated packet is cut into *k* slices. The *k* switch planes will therefore receive an identical input traffic distribution, and the delay through each plane will be identical, thereby ensuring that no packet will arrive out of sequence. Each bundling unit in the ingress *M*x*k* block holds a number of queues *Q,* which equals the number of inputs and outputs in each switch plane times the number of service classes. In case less than *k* packets are transmitted from a specific bundling queue, empty packets must be transmitted (to the same destination) over the remaining switches. In order to provide bounded delay in the bundling unit, *k* must be greater than *M* to compensate for the case where empty packets are transmitted.

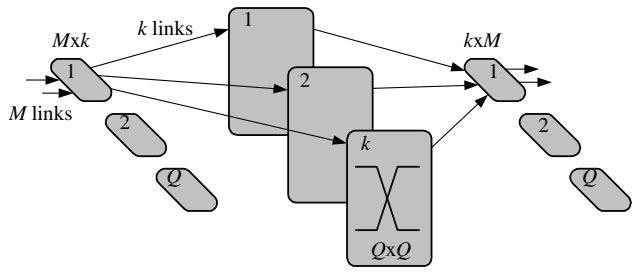

Figure 22: Multipath packet switch

The switch must be able to support multicast. The destinations of a multicast packet are determined by its multicast group identifier, which is converted into a multicast mask with Q bits. Packets with identical group id can be bundled together, however, this will require a bundling queue for each multicast id. Another option is to manipulate multicast masks of bundled packets by calculating the logical OR of masks. Thereby identical traffic distributions across each switch plane are achieved. By having larger multicast fan out than specified by the sender, the excess packets must be discarded by the egress kxM stage. The simplest solution is a scheme where all multicast traffic is bundled as one type, and this might waste a lot of bandwidth. If one of the packets is a broadcast packet, then all packets in the bundle will be broadcasted. A better solution is to make several multicast bundling queues and in the bundling try to gather packets with limited differences in their multicast masks. If the multicast mask only differs slightly, the bandwidth waste will be limited.

Switch fabric protection can easily be achieved by adding an additional QxQ switch plane, i.e. by increasing k. Furthermore; this will increase the performance during normal operation.

Many multistage/multipath switch architectures have been proposed in the literature; one scheme is denoted single stage port expansion [80][81] where the number of switch chips grows quadratically with the expansion factor. The number of switch chips only grows linearly with the expansion factor for the bundling scheme in Figure 22. The Atlanta architecture [82] is based on a Clos network similar to that in Figure 22. The central switch elements are bufferless crossbars so all packets will receive identical delays independent of the selected crossbar slice. This scheme requires a so-called concurrent dispatching algorithm to solve output contention in the crossbars. The required speedup (expansion factor) for non-blocking operation is 5:8, that is, M=5 and k=8 [25]. The bundling scheme can be non-blocking for a smaller expansion factor of e.g. 5:6 as shown in the next section.

The paper [83] proposes an "envelope" scheme where variable length packets are collected in large envelopes. The envelopes are switched in a VOQ based bufferless crossbar. The paper assumes that each traffic flow (from input i to output j) is leaky bucket constrained, and that the scheduler is weighted according to the bucket rate. Under these assumptions, it was shown that introducing envelopes did not reduce the throughput. The bundling scheme, however, does not assume constrains on the traffic parameters.

5.3. Scheduling

This section analyses the traffic bundling unit shown in Figure 21 in more detail with respect to queuing and scheduling. A scheduling algorithm is generally required in order to determine the queue from which to transmit an outgoing packet. The objective of the scheduling algorithm is to ensure that packets are bundled efficiently, but at the same time it must ensure that the packet delay is bounded. The objective is not to ensure fairness among traffic from the Q queues so scheduling algorithms like Weighted Fair Queuing (WFQ) or similar approaches [79] are not considered.

The scheduling scheme has its impact on the amount of buffering that is required and on the maximum delay that a packet will experience. A scheduling algorithm easy to implement is round-robin (RR) where backlogged queues are selected in turn. If the objective is to maximize the utilisation of the outgoing packets, then a queue is backlogged if it contains at least k packets. It is impossible to provide any delay guarantees in that case since a single packet can wait forever in a specific queue. To overcome this problem, a queue must be considered backlogged if it contains at least one packet. In this case, the maximum queue size and the maximum delay are bounded if the traffic is distributed equally across the queues; the size of each queue will initially grow until the system reaches equilibrium where M packets can be removed from the queue each timeslot. However, a traffic pattern for which the delay and queue size are unbounded exists. Consider a situation where a single packet is destined for each of the first (Q-1) queues. It is then assumed that packets are destined for the last queue in the subsequent timeslots until all the first (Q-1) queues have received service. If this scenario is repeated, the delay and queue size of queue number Q will grow infinitely (assuming that $Q>>k$).

It can be avoided that a queue grows infinitely by selecting a scheduling algorithm that always serves the longest queue. In the following, this scheme will be denoted LQ. The total queue size is upper-bounded by $M \bullet Q$ because if the total queue size is at this bound then at least the longest queue must contain M packets. In this case, at least M packets are removed, and a maximum of M packets will arrive. Thus, the total queue size will not increase. However, the delay is not bounded for LQ since a single packet in a specific queue may wait forever to receive service when another longer queue exists.

In the following, another scheduling approach is considered, which can provide bounded delay for packets entering the bundling buffers. The

scheduler works as follows: Each arriving packet is time stamped, and the backlogged queues are sorted according to the time stamp of the head of line packet. The queue with the lowest time stamp value is selected for transmission, and up to k (incoming) packets are removed from that queue. This scheme is denoted Time Stamp (TS). The maximum delay D, measured in timeslots, is given by:

$$D = \left\lceil \frac{(Q-1)\cdot(k-1)}{k-M} \right\rceil \tag{1}$$

And the maximum total queue size B, measured in packets, is given by:

$$B = D \cdot M = \left\lceil \frac{(Q-1)\cdot(k-1)}{k-M} \right\rceil \cdot M \tag{2}$$

From equation (1) it is observed that k must be greater than M in order to provide bounded delay. The minimum value of k is thus $k = M+1$. In this case, the following equations are obtained:

$$D = (Q-1)\cdot M \ , B = (Q-1)\cdot M^2 \tag{3}$$

Equations (1) and (2) are derived as follows: The worst-case scenario must be identified where a high number of outgoing packets only contain a single incoming packet. The input traffic distribution is selected in such a way that a single packet is transmitted to each of the first Q-1 queues. This is repeated as often as possible with the restriction that only one packet must be transmitted from these queues, i.e. each of the (Q-1) queues must as a maximum contain one packet. In the meantime, incoming packets are transmitted to queue number Q. Figure 23 shows the en-queue and de-queue operations for this worst-case scenario. The value of M is 2, and the value of k is 4. The squares show arriving packets, and the circles show departing packets. It is assumed that packets can be transmitted from the queue in the timeslot where they arrive. However, this assumption has no impact on the worst-case delay. Note that two packets are en-queued at each timeslot, which gives a slope of (–2) for the 'en-queue' graph. At time t1, each queue has received one packet, and the first Q/2 queues have received service. At time t2, service starts for the packets that arrived in the interval [t1:t2].

The last packet that arrived within this interval is transmitted at t5. In the interval [t2:t3] packets are en-queued in FIFO number *Q*. t3 is selected such that the en-queue and the de-queue graphs intersect at t5. If t3 is moved forward in time, some of the first (*Q*-1) queues will contain more than one cell at the time of transmission, and it is no longer a worst-case scenario. On the other hand, if t3 is moved backward in time, then a higher number of packets will be en-queued to queue number *Q*, which can be efficiently removed, thereby leaving the worst-case situation. The packets en-queued in [t2:t3] are transmitted in the interval [t4:t6]. The duration of [t4:t6] is 2/3 of [t2:t3] because 2 timeslots are required to de-queue six packets.

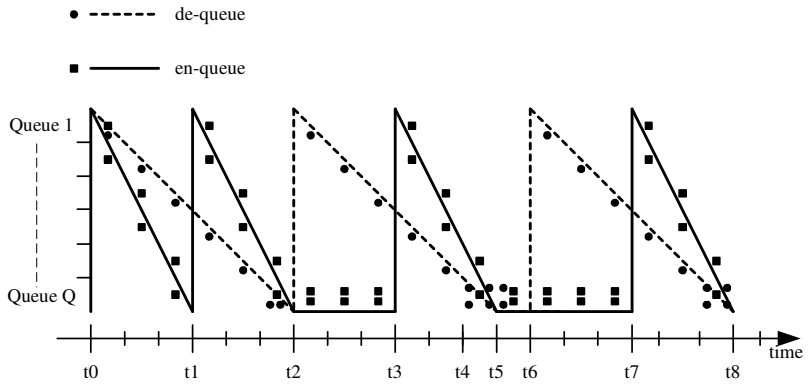

Figure 23: Worst-case packet service (*M*=2, *k*=4, *Q*=6)

The duration of the interval where packets are transmitted to queue number *Q* is increasing with time (e.g. [t5:t7] > [t3:t2]). However, after a given amount of time, the system will be in equilibrium. The equilibrium condition is shown in more detail in Figure 24. Note that the shown period corresponds to the interval [t2;t6] in Figure 23. It is still assumed that $M = 2$ and $k = 4$. Furthermore, the number of queues Q is set to 6 in this example. Note that the total number of squares equals the total number of circles because of the equilibrium condition.

It is now possible to calculate the number of timeslots in Figure 24. This number is equal to the maximum delay of a packet. To see this, consider a packet arriving at t4. This packet will receive service at time t8, that is, one period later. The number of timeslots is denoted D and is given by the following equation:

$$M \cdot D = (Q-1) + k \cdot (D-1-(Q-1)) + r \qquad (4)$$

The left side is the number of squares, which is always M per timeslot. The right side expresses the number of circles. There is one circle for the first (Q-1) timeslots, and the following (D-1-(Q-1)) timeslots contain k circles each. The last timeslot contains r circles $(M < r \le k)$. Solving for D gives:

$$D = \frac{(Q-1)\cdot(k-1)}{k-M} + \frac{k-r}{k-M} \tag{5}$$

Since D is an integer and the last part of the equation above is less than 1, the result given by equation (1) is finally obtained.

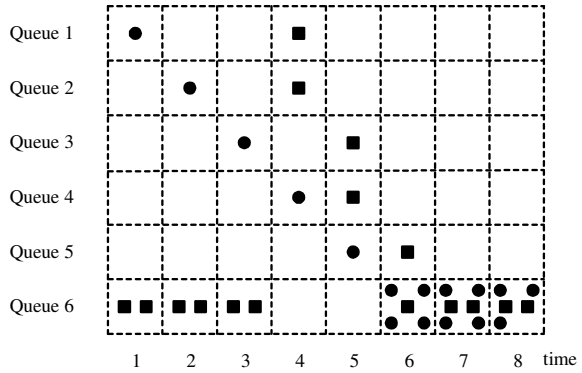

Figure 24: One period of de-queue and en-queue operations

The worst-case delay and buffer requirements are useful in an actual physical implementation because the maximum number of different time stamp values is D, and B gives the memory requirement.

The total number of packets in the queues will show a local maximum at time t4, t8… At equilibrium, this is also the global maximum. To obtain the number of packets at e.g. t4, the en-queue operation is stopped at that time, and the numbers of packets leaving the queues in the following period are counted. At equilibrium, this number is $M \cdot D$ which leads to equation (2).

5.4. Simulation and Results

The goals of the simulation are to examine the presented bundling scheme with respect to mean delay and delay variations. The delay

distribution of packets in the queue system in Figure 21 depends on the distribution of arriving packets and the scheduling mechanism.

In general, the packet inter-arrival time and the destination queue are given by stochastic variables. However, it is assumed that the system is fully loaded with a packet arriving in each timeslot on each of the M channels. Thus, the only stochastic variable is the destination queue. It is assumed that the different queues are selected with equal probability $1/Q$. A switch fabric of size 128x128 is considered, which is generated from three 64x64 switch elements; the parameters in Figure 22 are thus: $Q=64$, $M=2$ and $k=3$. Figure 25 shows the (un-normalized) probability distribution for delays of three different scheduling methods, LQ, RR and TS. The mean values are as follows: LQ=17.7, RR =34.3 and TS = 28.7. The delay bound can be calculated for the TS scheduler according to equation (1) as D= 126. It is noted that the LQ scheduler has the lowest mean value so that most of the packets obtain a low delay, however, the tail of the distribution is much longer than for RR and TS. Actually, the LQ distribution takes values far beyond 126. The mean value of RR is higher than for LQ, but the tail of RR is reduced compared to LQ, which makes RR more attractive than LQ. The TS scheduler has a mean value that is lower than RR. Furthermore, the slope of the distribution falls steeply towards zero at a delay value around 50, which is far below the theoretical maximum at 126. The fact that TS has a lower mean value and a smaller tail than RR makes TS the most well-performing scheduling scheme for this traffic scenario.

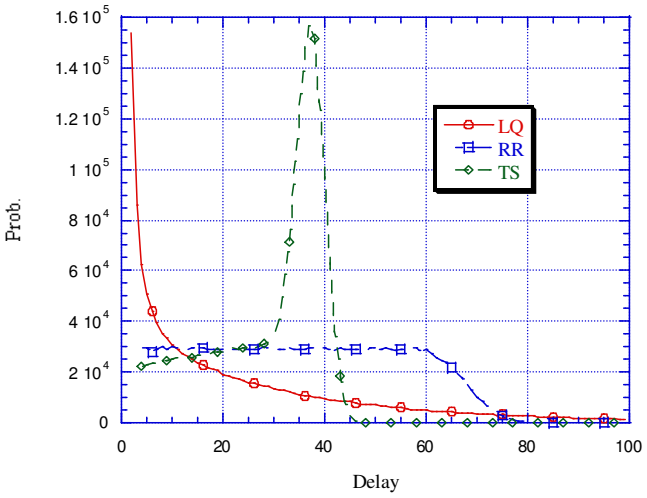

Figure 25: Probability distribution for delay (*M=2,k=3,Q=64*)

A number of experiments has been carried out with different distribution functions for the destination queue: In experiment one, the probability of selecting a specific queue was proportional to the queue number, and in experiment two the queue number was selected according to a exponential distribution (truncated and normalized). The resulting probability distributions for delay does not vary much from that shown in Figure 25 so the above conclusions regarding LQ, RR and TS hold for a number of different destination queue distributions.

The probability distributions will now be examined for a larger switch fabric of size 256x256 defined by the following parameters: *Q=64*, *M=4* and *k=5*. The result is shown in Figure 26.

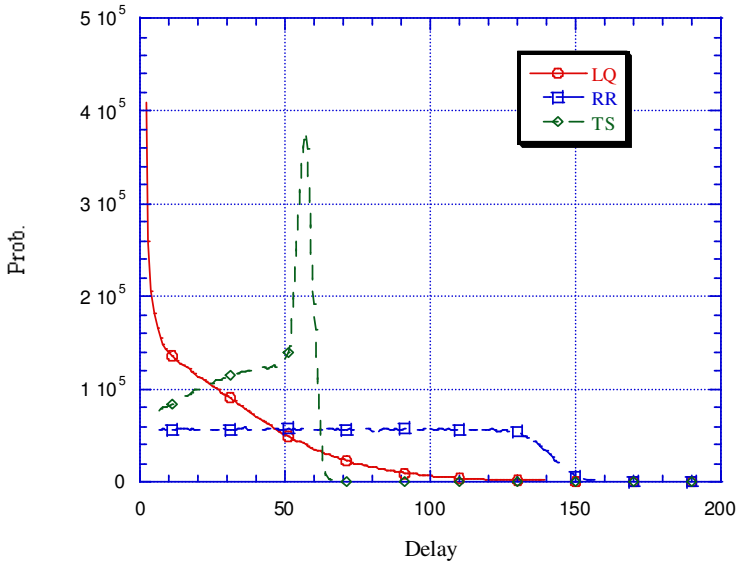

Figure 26: Probability distribution of delay (*M=4,k=5,Q=64*)

The mean values are LQ = 25.1, RR = 70.6 and TS = 38.0. The worst-case delay for TS given by equation (1) is 252. By comparison between Figure 26 and Figure 25, the same conclusions regarding LQ, RR and TS are reached, actually the TS scheduler performs even better than the other two in the *M*=4 case than the *M*=2 case.

For a given switch size e.g. 256x256, *Q*=64, *M*=4, the value of *k* can be increased in order to reduce the bundling delay. Also the worst-case delay given by equation (1) is reduced towards (*Q*-1) for large *k* values.

Table 7 shows the delay mean values for LQ, RR, TS and the maximum for different values of *k*. The worst-case bound for TS is shown as well.

Table 7: Mean delay vs. *k*

k	LQ	RR	TS	TSmax
5	25.1	70.6	38.0	252
6	25.1	42.4	32.9	158
7	25.1	35.7	31.3	126
8	25.1	34.0	30.8	111
16	25.1	33.0	30.6	79

As previously discussed, protection can be introduced by increasing the value of *k*. By using the value of 6 instead of 5, the mean value is reduced by 13 % and the maximum value by 37 % for the TS scheduler according to

Table 7. Non-blocking operation is still possible if one of the six switch plane fails, but the cost is increased delay.

In order to evaluate the significance of the delay, it is assumed that the link speed is 10 Gbit/s, and that the packet size is 80 Bytes. The duration of a slot (packet) is thus 64 *ns*. With *D*=126, the worst-case delay becomes 126*64*ns* = 8*us*. This is approximately equal to the transmission delay of 2 km fibre, which shows that the bundling delay is insignificant compared to the total end-to-end delay.

5.5. Speedup

In section 5.2, it was assumed that the duration of an en-queue timeslot was equal to the duration of the de-queue timeslot. The relationship between bandwidths at the en-queue side and de-queue side is denoted *speedup*. It is defined as *S=k/M*. Thus, the bundling efficiency is given by *1/S*. If *k=5, M=4,* then S=1.25 which implies that a bandwidth of 50 Gbit/s is needed to aggregate four 10 Gbit/s flows. In order to reduce the required speedup, the de-queue timeslot is assumed to be larger than the en-queue timeslot. In Figure 24, the number of de-queue and de-queue timeslots is denoted *D*. The number of en-queue timeslots is now *D(M),* and the number of de-queue timeslots is *D(k) (D(k)>D(M))*. Equation (4) is now modified accordingly.

$$M \cdot D(M) = (Q-1) + k \cdot (D(k) - 1 - (Q-1)) + r \qquad (6)$$

Equation (6) can be re-written to:

$$k \cdot D(k) - M \cdot D(M) = (Q-1)(k-1) + (k-r) \tag{7}$$

Note that $D(k) = D(M) = D$ is a possible solution to the equation when D is given by equation (1). Assume that n is a positive integer, other solutions are:

$$\begin{aligned} D(k) &= D + n \cdot M, \\ D(M) &= D + n \cdot k \end{aligned} \tag{8}$$

This is easily verified by inserting equation (8) into equation (7), which gives equation (5). The speedup is now given by $S=(k/M)*(D(k)/D(M))$. For larger values of n, the speedup converges to 1 with the cost of additional delay. Other solutions to (7) than those given by (8) may exist, since the r value in (7) can be modified. In general, the expression at the left side of equation (7) can give all multiples of the largest common divisor in k and M, $lcd\{k,M\}$, which means that r can be changed in steps of $lcd\{k,M\}$ with the restriction $(M < r \le k)$.

As an example, the previous configuration with ($M=2,k=3,Q=64$) is considered. The worst-case delay is D=126. Now it is assumed that the link speed of the central switch elements is reduced to 80 % of the original speed, hence $D(k)/D(M)$ = 0.8. The number n can be calculated from equation (8) as follows:

$$n = \frac{D\left(1 - \dfrac{D(k)}{D(M)}\right)}{k \cdot \dfrac{D(k)}{D(M)} - M} \tag{9}$$

In this case, n=63 is obtained. Now the worst-case delay can be determined from equation (8): D(M) = 126+63*3 = 315. The worst-case value is increased by a factor of 2.5. The typical delay for a more realistic traffic pattern will not increase by the same factor, but will instead lead to a more efficient filling of the bundled packets.

The reduction in link speed for the central (QxQ) switch elements reduces the power consumption, and also makes the switch less challenging to implement.

5.6. Summary

This chapter demonstrated that the concept of traffic bundling has attractive properties that can be utilised within multistage/multipath packet switch fabrics where aggregated packets are transmitted over identical parallel planes.

A simple scheduling algorithm, which applies time stamps to arriving packets and serves packets in order of increasing time stamp, was proposed. Worst-case scheduling delay and buffer occupancy were derived for this specific scheduling algorithm. The proposed scheduling algorithm performs bundling efficiently (i.e. with the smallest possible bandwidth overhead) and at the same time, bounded delay is provided. Simulation results demonstrated that the actual delay for different distributions is much smaller than the derived worst-case value.

The time-stamp scheduling algorithm was compared to round-robin and Longest Queue First scheduling, and simulation results indicated that the time-stamp algorithm showed the best overall performance even though the Longest Queue First scheduler provided the lowest average delay.

Finally, speedup was discussed. The speedup can be reduced by introducing a different number of en-queue and de-queue timeslots in a period. The impact on the worst-case delay was calculated.

6. Buffered Crossbar Switch

This chapter presents a modified architecture for a buffered crossbar switch that overcomes the memory bottleneck with only a minor impact on performance. The proposed architecture uses two levels of backpressure with different constraints on round trip time. Buffered crossbars are considered an alternative to bufferless crossbars mainly because the latter requires a complex scheduling algorithm that matches input with output. Buffered crossbars require only a simple scheduler that operates independently for each output queue column. The memory amount required for a buffered crossbar is proportional to the square of the number of ports and the round trip time. The proposed architecture reduces the amount of memory in the buffered crossbar without increasing the scheduling complexity.

6.1. Introduction

Crossbar switch fabrics have been studied extensively in the literature. In combination with Virtual Output Queuing (VOQ), the architecture provides a scalable solution with respect to memory access bandwidth. The crossbar can be either unbuffered or contain a small amount of buffering in each crosspoint. A bufferless crossbar requires a complex scheduling mechanism that matches input with output. The scheduling algorithm can either calculate a maximum match or a maximal match. A maximum match algorithm pairs the maximum number of input and output, whereas a maximal match has no cell in any input queue destined to an unmatched output. A number of maximum weight matching algorithms have been presented in [21]. Their main disadvantage is timing complexity, leading to an interest for maximal matching algorithms such as PIM and iSLIP [84]. SLIP matches input with output by having a round-robin scheduler for each input and output. The input schedulers independently select an output, and the output scheduler selects among contending inputs. The iterative SLIP, iSLIP, performs a number of iterations of SLIP. To compensate for the

lower performance of a maximum matching algorithm, speedup can be introduced between the VOQs and the crossbar. A speedup of 2 is sufficient to obtain 100 % throughput [24].

Due to the complexity of scheduling algorithms for bufferless crossbars, buffered crossbars are considered as an alternative. By adding a small buffer capacity in each crosspoint, it is possible to perform the scheduling decision independently among the output columns. The crosspoint buffers generate backpressure signals towards the VOQs in the port card to avoid overflow. The minimum crosspoint buffer size to maintain full throughput is determined by the round trip delay for the backpressure mechanism. As an alternative to small crossbar buffers in combination with VOQ, one may consider pure crosspoint buffering, however, this requires large buffer capacity in each crosspoint to reduce cell loss.

It has recently been shown that a buffered crossbar switch with a speedup of 2 can emulate a pure output queued switch [85]. A similar result is available for bufferless crossbars: Emulation of an output buffered switch of size *NxN* is obtainable with a speedup of 2-1/*N* [86]. The emulation algorithm proposed in [86] is, however, much more complex than the one proposed in [85]. This result indicates that QoS is more easily supported in the buffered crossbar architecture.

The performance of buffered crossbars with VOQ has been studied in various papers. The architecture was originally proposed in [87] where a simple round-robin scheduling scheme was compared to a more advanced scheme taking into account buffer size and cell age. In [88], a stability analysis is performed for a CICQ (Combined Input and Crosspoint Queued) switch with one cell sized crosspoints. The switch uses Longest Queue First VOQ schedulers. Different combinations of scheduling algorithms are compared in [89]. Longest Queue First, Oldest Cell First and round-robin were considered for VOQ scheduling in combination with Oldest Cell First and round-robin for the crossbar. The paper concludes that the performance is quite similar and recommends the round-robin approach due to its simplicity. The combined input one cell crosspoint buffered switch (CIXB) is compared to iSLIP and pure output queuing (OQ) in [89]. The delay performance of CIXB is better than iSLIP and close to that of an OQ switch. For unbalanced traffic, that is traffic with an uneven distribution of destinations, the CIXB will not support 100 % throughput even if the traffic is admissible. The throughput for unbalanced traffic is, however, better for CIXB compared to iSLIP. In [92] the study is extended to cover more than one buffer location in the crosspoints. Due to the round trip time for backpressure signals, a single buffer location in each crosspoint is

not feasible. The high memory consumption is the main drawback of this architecture, and the results are mainly interesting from a theoretical point of view.

Another benefit of a buffered crossbar compared to a bufferless crossbar is the less stringent synchronisation requirement between the port cards and the switch cards [93]. Bufferless crossbars require that all port cards are synchronized to the same clock.

This chapter presents a modification to the buffered crossbar architecture that overcomes the memory problem with only a small impact on performance. Each crosspoint buffer is reduced to a minimum size of one cell, and a small, shared VOQ memory is added in front of each row of crosspoints. This configuration requires two levels of backpressure; a fast mechanism between the small crosspoints and the on-chip crossbar VOQs, and a slower mechanism between the VOQs in the port card and VOQs in the crossbar. The switch architecture is described in more detail in section 6.2. The simulation study presented in section 6.3 compares the performance of this switch architecture to a standard buffered crossbar system. Moreover, the simulation study is used as a guideline for system dimensioning. Finally, concluding remarks are given in section 6.4.

6.2. Switch Model

A crossbar buffered switch system CIXB of size $N \times N$ consists of N Input/Output port cards and a switch card implementing the N^2 crosspoint buffers as shown in Figure 27. Each input port card contains VOQs with one buffer for each of the N outputs. The switch model uses round-robin scheduling between VOQs in the port cards and also between crosspoint buffers in an output column. The output port card contains a buffer to store cells in case of speedup. In order to avoid overflow, the crossbar buffers will generate a backpressure signal towards the corresponding VOQ buffer in the port card. The round trip time for backpressure *RT_BP* is defined as the number of timeslots it takes to stop the cell flow to a specific crosspoint buffer measured from the time when backpressure is asserted by that crosspoint. The round trip time is composed of a propagation delay for the backpressure signal, the time it takes before the port card scheduler is blocked, and the data path delay from the port card scheduler to the crosspoint. To achieve 100 % throughput, the minimum crosspoint buffer size is 2**RT_BP*. Backpressure is then asserted if the buffer level is larger than or equal to *RT_BP*.

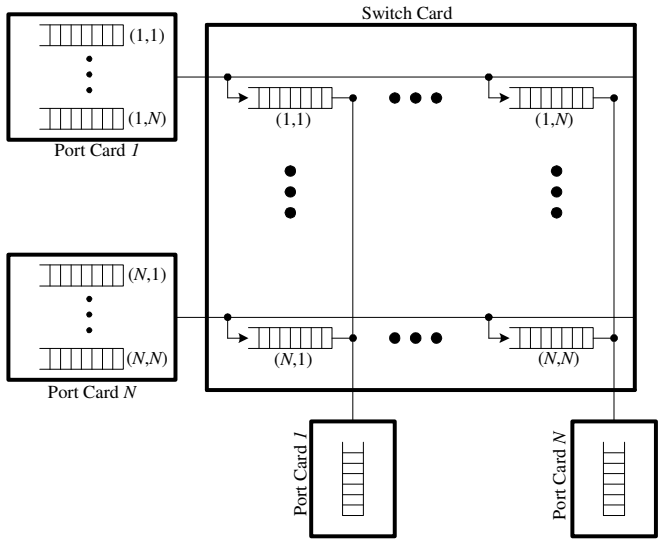

Figure 27: CIXB switch

The main advantage of the CIXB switch architecture shown in Figure 27 is the low scheduling complexity between crosspoint buffers in an output column. A simple round-robin scheduler can be implemented very efficiently [94]. However, the total amount of storage is $O(2 \cdot RT\_BP \cdot N^2)$. With a round trip time of four, a *32 x 32* switch will contain 4M memory bits for a packet size of 64B. Having e.g. eight traffic priorities results in 32 M bits memory, which is a very large amount, and this is not feasible on a single chip.

From a performance point of view, the CIXB switch behaves like an output buffered switch for very large crosspoint buffers, and 100% throughput is achieved for all admissible traffic patterns. This is, however, not the case for limited size crosspoint buffers. In [90], the reduction in switch throughput has been investigated for unbalanced traffic. To increase throughput, a speedup can be introduced between the port card and the switch card. In the following, this is referred to as *External Speedup*. The egress port card must then contain buffering to adapt between the different rates, as shown in Figure 27.

Figure 28 presents a modified switch card architecture that uses a smaller amount of memory compared to CIXB. The objective of the proposed architecture is to reduce the crosspoint queues to a minimum possible size of one cell. This is achieved by adding an additional queue system in front of each crosspoint row. The new queue system has a dedicated VOQ for each crosspoint in that row. The VOQs are implemented in a

shared memory following e.g. a linked list approach. In the following, the system is denoted CISXB.

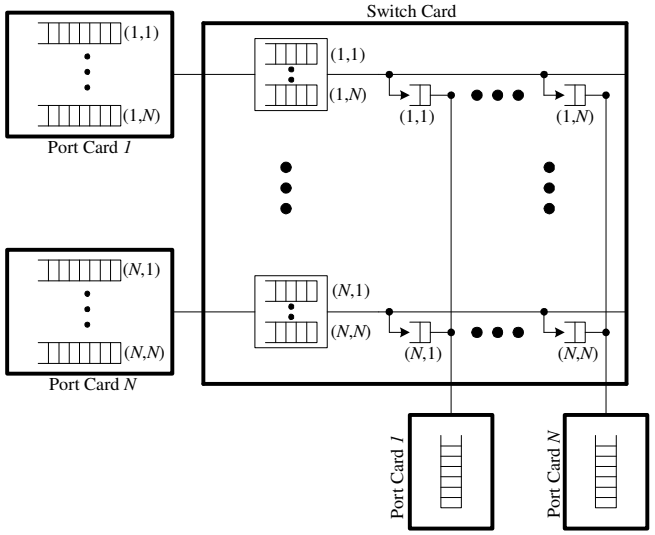

Figure 28: CISXB switch

The CISXB switch requires two levels of backpressure; the first level controls the filling of the crosspoint buffers, and the second level controls the filling of the shared memory. With a crosspoint buffer size of only one cell, the first level of backpressure must have very low latency. This can be achieved because the crosspoint buffers and the shared memory reside on the same switch chip. Each crosspoint in a row generates backpressure towards the corresponding VOQ in the shared buffer. The backpressure between the VOQs in the shared memory and the VOQs in the port cards has a higher round trip delay equal to *RT_BP* for the CIXB switch in Figure 27. The VOQs on the switch card share memory, which implies that the VOQs can accommodate buffer space for a limited number of destinations at the same time. If the shared memory occupancy exceeds a global threshold, then a global backpressure signal concerning all VOQs in the shared memory is generated.

The CISXB architecture requires an additional scheduler to select between the VOQs in the shared memory. In this work, a round-robin scheduling mechanism is considered. The switch chip has then two levels of round-robin arbitration; first among the VOQs in the shared memory, and then among the crosspoints in an output column. Other scheduling algorithms could be considered, but the main objective in this chapter is to compare the CIXB and the CISXB architectures.

In the following, the number of cells in VOQ number i in the shared memory is denoted Q_i. The backpressure threshold for queue i is B_i, that is, a backpressure signal is generated if $Q_i \geq B_i$. Due to the round trip time for backpressure signals, the size of queue i can grow to $Q_{i,\max} = B_i + RT\_BP$. The total number of cells in the shared buffer is $Q = \sum Q_i$. The total capacity of the shared memory S is typically much smaller than $\sum Q_{i,\max}$, therefore a global backpressure threshold B is introduced to avoid queue overflow. The global backpressure signal is then asserted if $Q \geq B$. The global threshold must be selected such that $B + RT\_BP \leq S$ in order to avoid overflow in the shared buffer.

The total memory capacity of the CISXB switch chip in Figure 28 can be reduced compared to the CIXB in Figure 27 if the added size of the shared memory is smaller than the reduction in crosspoint memory size. In principle, the shared buffer could be as small as $2 \cdot RT\_BP$ to avoid queue overflow, however, the performance of the switch will suffer from frequent blocking due to the global backpressure signal. The size should be large enough to reduce the global blocking to a minimum; the CIXB switch can achieve 100 % throughput for uniform Bernoulli traffic [90]. The CISXB switch will not be able to support 100 % throughput if the global backpressure is invoked because the transmission from the port card is blocked completely. In general, the size of the shared buffer will depend on the traffic profile (e.g. uniform, bursty) and the load. In section 6.3, this subject is investigated further by a simulation study.

The performance of the CISXB switch can be increased by internal speedup between the shared buffers and the crosspoint buffers in the switch card. With an internal speedup of *IS*, the round-robin scheduler for the shared queue performs *IS* scheduling decisions for each decision of the crosspoint column scheduler. Internal speedup will not affect the bandwidth between the port cards and the switch card, but the internal bandwidth between the shared memory and crosspoint memory must be *IS* times higher. Note that the system behaves like an output buffered switch if *IS*=*N* and the shared buffer is sufficiently large. An internal speedup of more than 2 is, however, seldom feasible.

6.3. Simulation and Results

A simulation study has been carried out in order to compare the CISXB switch with a well known buffered crossbar system CIXB. Each port card receives cells from a source. In each timeslot, the source generates a cell

with probability equal to the load ρ. The switch size is 32x32. The destination is selected randomly according to a uniform distribution. Assigning the same destination to a number of consecutive cells generates bursty traffic. Figure 29 shows the average delay as a function of load for a burst length of 0, 10 and 20, respectively.

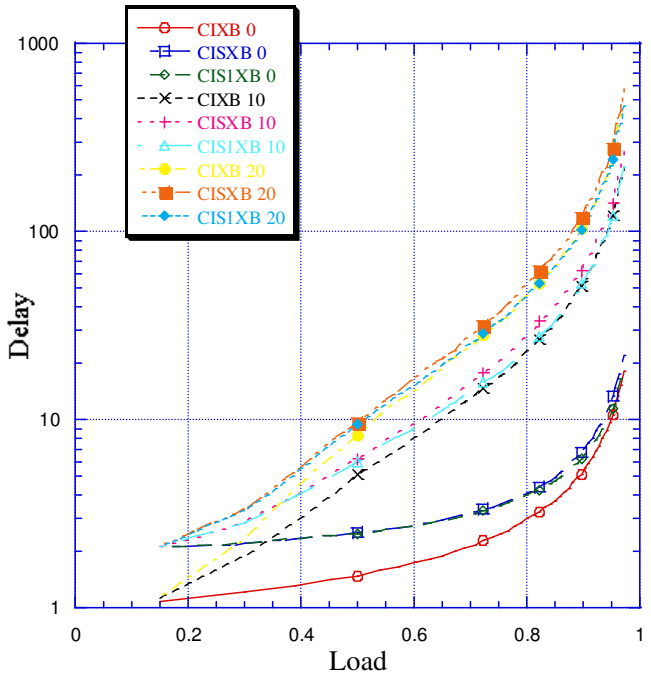

Figure 29: Delay vs. Load. The burst size is 0, 10 and 20, respectively

The round trip time for backpressure is set to four for both CIXB and CISXB. The backpressure threshold for each individual queue in the shared input buffer is set to the minimum value of four. The global threshold for the shared buffer is set to 92, which has shown to be sufficiently large to avoid global backpressure during all simulations. The total size of the shared buffer is then 92+4 = 96. The crosspoint size of the CIXB switch is at the minimum possible size of eight positions. The total number of cell positions in CIXB is then 8*32*32 = 8192. The number is 96*32 + 1*32*32 = 4096 for the CISXB switch, i.e. half the amount. For each burst length, the plot shows the delay for CIXB, CISXB and CISXB with an internal speedup of two, CIS1XB. The average delay of CISXB is a little larger than of CIXB. For small load values, the delay of CISXB is close to that of CIS1XB because the average number of cells in the shared crossbar buffer is very small, and

consequently, there is no gain in having internal speedup. For large load values there is a reduction in delay from internal speedup. In this case, the delay of CIS1XB is close to that of CIXB. Since the average delay for an output buffered switch is very close to that of a CIXB, the gain in performance from further increasing the speedup is limited.

The larger delay of CISXB is mainly due to delay in the shared input buffer. The average size of the shared input buffer is depicted in Figure 30 for burst lengths of 0, 10 and 20, respectively. The buffer size depends strongly on the load, but not on the burst size. For load values close to 100 %, the required buffer size is quite large. The switch card load is reduced by use of external speedup between the port card and the switch card. Thereby, the shared buffer size is reduced as well. If the external speedup is e.g. 1.5, then the maximum load of the switch card becomes 66,7 %. From Figure 30, it is seen that the average buffer size is below 10 even for a burst size of 20. Note that the difference in buffer size between a burst length of 10 and 20 is rather small compared to the difference between 0 and 10. This indicates that the size of the buffer only slightly depends on the traffic burstiness. In the following, the dependence on bursty traffic is investigated in more detail.

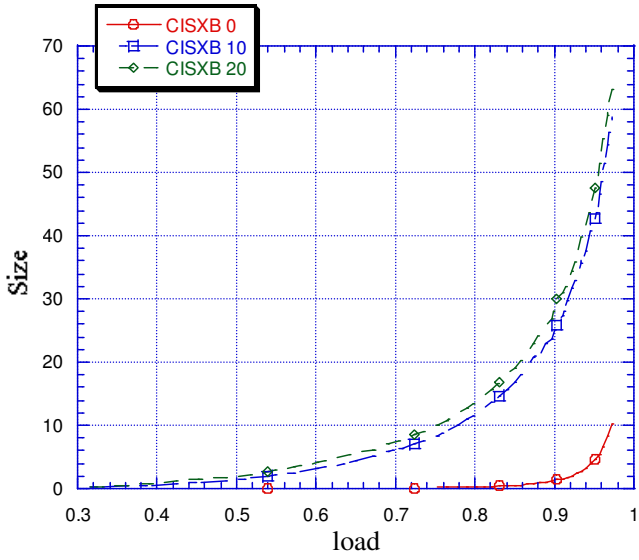

Figure 30: Size (occupancy) of shared input buffer vs. load. The burst size is 0, 10 and 20, respectively

As explained in section 6.2, the size of the shared crossbar buffer should be sufficiently large to ensure that the probability of global backpressure

is minimized. In order to determine the size of the shared crossbar buffer, the average buffer size as a function of burst length has been determined. The results are shown in Figure 31. The figure shows both the average buffer size in the port card and the shared memory size in the switch card. The load values are 70 %, 80 % and 90 %. The size of the shared crossbar buffer increases rather slowly with the burst size. A detailed investigation of the plot shows that the size grows only logarithmically as a function of burst size. This result is used to dimension the buffer by taking only the system load into account. Assuming an external speedup of 1.5 (load = 66,7 %), the average shared buffer size will be below 10 according to Figure 31. By allocating 32 buffer locations, global backpressure becomes extremely rare even for very bursty traffic. The total number of memory locations for the CISXB architecture then becomes 32*32 + 1*32*32 = 2048, a reduction of 75 % compared to the CIXB architecture. In average, only 2 cells per crosspoint are needed. The performance in terms of cell delay is only slightly degraded. For unbalanced traffic, the shared buffer size requirement becomes even lower even if the overall throughput is reduced so the system behaves properly under both bursty and unbalanced traffic scenarios.

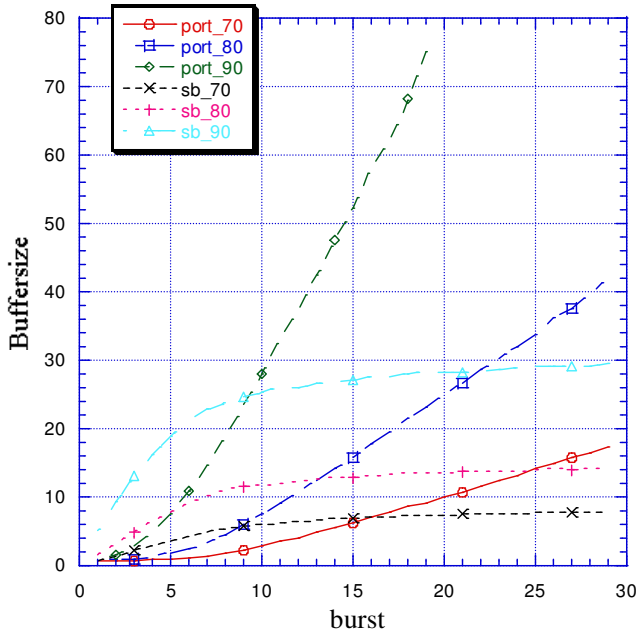

Figure 31: Size (occupancy) of shared input buffer and port card buffer vs. burst size. The load is 70 %, 80 % and 90 % respectively

In the previous discussion a speedup of 1.5 was taken as an example without further explanation. The switch behaviour under unbalanced traffic is now investigated in more detail. The model for unbalanced traffic from [90] is used below. The model is commonly used to describe unbalanced traffic [91]. The unbalanced weight ω defines the degree of unbalance. The load from input port s to output port d is denoted $\rho_{s,d}$:

$$\rho_{s,d} = \begin{cases} \rho(\omega + \frac{1-\omega}{N}) & \text{If } s=d \\ \rho(\frac{1-\omega}{N}) & \text{Otherwise} \end{cases}$$

Note that

$$\sum_s \rho_{s,d} = \sum_d \rho_{s,d} = \rho$$

The traffic matrix is thus admissible, and all input and output have a load equal to ρ. If $\omega = 0$, there is no unbalance, and if $\omega = 1$, the traffic is completely unbalanced. Figure 32 shows the performance degradation under unbalanced traffic. The switch parameters are identical to those used in Figure 29. The throughput penalty is highest for the CISXB switch. The throughput for a CIXB switch with 1-cell crosspoint buffers is shown in [90], and the result is quite close to that of CISXB shown in Figure 32. The CIXB with 8-cell crosspoint buffers has a smaller reduction in throughput according to Figure 32. It is concluded that the throughput reduction for unbalanced traffic mainly depends on the crosspoint buffer size. Internal speedup between the shared input buffer and the 1-cell crosspoints increases performance; the throughput for CIS1XB is slightly better than for CIXB. The CIS1XB switch, however, requires double internal speed and data path bandwidth so it is more feasible to use CISXB with a slightly higher external speedup.

The throughput for iSLIP with four iterations is shown in [90]. The minimum throughput is 0.8, which is lower than for CISXB with a minimum value around 0.85 according to Figure 32.

To compensate for the throughput reduction of unbalanced traffic for CISXB, an external speedup is required. To equalise the difference in throughput between CIXB and CISXB, the CISXB must have speedup that is approximately 10 % higher compared to CIXB. Also, with a 10 % increase in speed, the delay performance of CISXB will reach that of CIXB according to Figure 29.

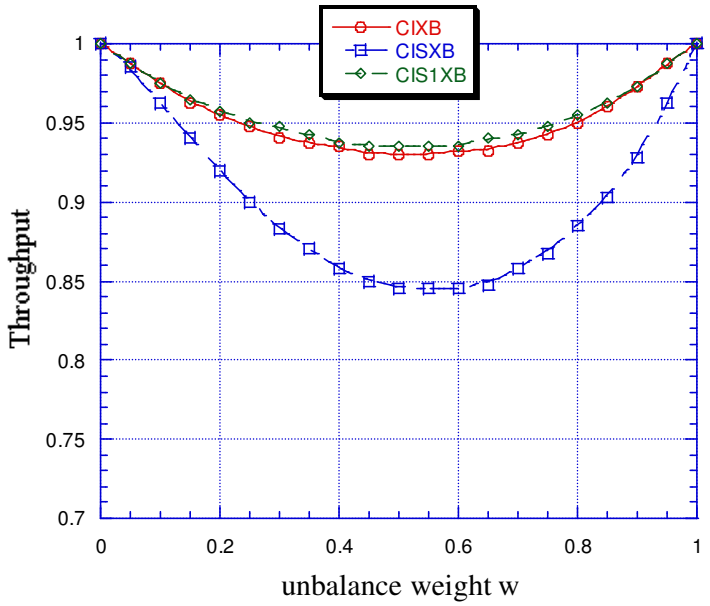

Figure 32: Throughput with unbalanced traffic

The results obtained above are valid for a single priority only. However, it is believed that the reduction in memory would be even higher if the switch supports more than one priority: For the pure buffered crossbar, the memory consumption will become *P times* higher with *P* priorities in the system. The modified architecture with 2 levels of backpressure will also require *P* times more memory for the one-cell crosspoint buffers, but the size of the shared input buffer is expected to be less than *P* times higher. The exact buffer requirements for the shared input buffer in case of priorities are for further study.

The proposed architecture solves another potential problem related to large switching systems covering several racks. This could lead to round trip times between port cards and switch card much higher than four, which was assumed in the simulations. This will lead to very high memory consumption for a buffered crossbar switch, whereas the modified architecture requires only additional memory in the shared input buffer to compensate for the increased delay. Also, for a bufferless crossbar with iSLIP scheduling, it has been shown that large round trip delays lead to a significant drop in throughput [95]. However, the reduction in throughput could be prevented by a modification to iSLIP so that only packet arrivals (and not the whole state of the VOQs) are

send to the central scheduler. Of course this introduces the usual problem of reliability when only state changes are communicated.

6.4. Summary

Buffered crossbars have several advantages compared to non-buffered crossbars including simpler arbitration, synchronisation relaxation and better performance. The main drawback, however, is the total amount of crossbar memory, which is proportional to the square of the number of input/output ports and backpressure latency.

This chapter introduced a new architecture for a buffered crossbar that uses two levels of backpressure to reduce the amount of memory used in the switch card. The proposed switch uses a small, shared VOQ memory in combination with a one-cell deep crosspoint buffer. Each shared queue system uses independent schedulers so the time complexity of the arbitration is identical to that of a pure crossbar buffered switch card.

The performance has been investigated by a simulation study. It was shown that the amount of memory is reduced significantly with only a small reduction in performance. The switch was insensitive to the burstiness of traffic, and the study shows that a reduction of 75 % in memory could be obtained for a 32x32 switch with a backpressure round trip time of 4 timeslots. The performance reduction can be compensated by an additional speedup of 10 %, or it can be compensated by an internal speedup of 2 between the shared input buffer and the crosspoint buffers. It is also expected that the memory savings will become even higher in a switch containing several priorities.

7. IP lookup & Classification

The Traffic Manager functions include among others address lookup & classification, queuing & scheduling and segmentation & reassembly. These functions can either be implemented in specific hardware (ASIC) or in programmable network processors (NPU), of course with a trade-off between speed and flexibility. In this chapter, the forwarding task i.e. address lookup and classification will be in focus, and the chapter presents an IP address lookup algorithm with low memory requirement and fast updates. The scheme, which is denoted prefix-tree, uses a combination of a trie and a tree search, which is efficient in memory usage because the tree contains exactly one node for each prefix in the routing table. The time complexity for update operations is low for prefix-tree. The lookup operation for the basic binary prefix-tree may require that a high number of nodes be traversed. This chapter presents improvements to decrease lookup time, including shortcut tables and multi-bit trees. The prefix-tree is compared to a trie and a path compressed trie using prefixes from a real routing table.

7.1. Introduction

The packet forwarding decision typically includes both address lookup and packet classification. The address lookup operation determines the next hop and output interface of the router. IP lookup algorithms have been studied extensively in the literature. The introduction of Classless Inter Domain Routing (CIDR) has reduced the size of forwarding tables, but the lookup procedure is more complex because exact matching is replaced by longest prefix matching [97][98]. Before the introduction of CIDR, the address space was divided into classes (class A,B,C) distinguished by the leading bits in the address: class A : 0, class b : 10 and class C : 110. The remaining bits of the address were divided into a network field and a host field, and the routing decision was based on the network field. The size of the network field for class A is 7 bits, for class B 15 bits and for class C 23 bits. This scheme did not work well for several reasons: Firstly, the address space of class A takes up 50 % of the available

address space, and secondly, the 8 bit host field of class C was too small for many networks which resulted in many consecutive class C addresses to represent a single network. The first problem led to exhaustion of the address space, and the second problem resulted in unnecessarily large routing tables. With the growing number of terminals, it was necessary to move to CIDR. In this classless scheme, the route 192.38.77.0/22 represents the concatenation of 4 class C addresses. With CIDR, some addresses can match several prefixes, e.g. 192.38.77.0/22 and 192.38.77.0/24. In this case, the longest match is selected as the best match.

The classification task requires a search on multiple fields [98]. In case of TCP/IP, the fields could include source address, destination address, source port, destination port and protocol. Packet classification is used for various tasks including DiffServ/Intserv, firewall, access control, accounting and traffic engineering. The individual fields can either specify an exact value, a prefix or an interval. Rules are typically associated with a priority since a given packet may match several rules. Now, the number of rules that exists is typically much lower than the number of prefixes in the routing table. In [99], the size of classifiers has been measured, and only a few classifiers contain more than 1000 rules. However, taking the age of this reference into account, the number today is somewhat higher. Several algorithms for packet classification have been proposed, e.g. Grid of Tries [100], Area Based Quad Trees [101] and Parallel Packet Classification [102]. Furthermore, a wide range of algorithms is described and compared in [98]. A very fast packet classification can be obtained by TCAM (Ternary Content Addressable Memory). Ternary means that the CAM can store 0, 1 and don't-cares. The TCAM memory array stores rules in decreasing order of priorities and compares an input pattern with every stored element in parallel. TCAM with 64 K entries and 288 bit wide search keys are commercially available today [103], and TCAM is thus an appealing solution if the number of rules is not too high (e.g. micro-flow classification in core routers). The Parallel Packet Classification scheme reduces the size of the TCAM by having preceding independent field searches in parallel. The independent field search reduces the number of bits in the search pattern.

TCAM's can potentially be utilised for longest prefix match IP lookup operations as well. However, the rules must be stored in decreasing order of prefix length, which may require a huge table re-organisation in case of routing updates. Alternative search table based lookup algorithms are therefore more suitable for IP lookup. A trie structure [104] is a convenient way to represent the prefixes in the forwarding table. The lookup and update procedures are simple, but the lookup procedure may

traverse up to B nodes if B is the number of address bits. A trie will contain a high number of intermediate nodes, which do not contain any prefixes. Removing any intermediate nodes with only one child node can reduce the number of intermediate nodes. The resulting trie is called a 'path compressed trie' or 'Patricia trie'. Another approach is Level Compressed (LC) tries [106]. They can increase lookup speed with the cost of a more time-consuming update procedure. The time complexity of the update procedure is important because frequent routing updates may occur [107].

The lookup speed can be improved by having balanced trees, e.g. range search trees. If N is the number of prefixes, the time complexity of lookup is O(log(N)). The main drawback is the update time; to keep the tree balanced, it is usually necessary to re-construct the whole tree.

This chapter presents a new tree structure for storing the IP forwarding table. It is based on an exact match VPI/VCI search algorithm developed in [96]. Modifications have been introduced to support variable length prefixes. The prefix-tree structure uses a combination between a trie and a tree; in each node a comparison is performed similar to a tree. Branches are performed as in tries based on the address bits. The algorithm is efficient in terms of memory usage because of a strict linear bound on memory usage. The scheme is therefore suitable for implementation using on-chip memory, which is of particular importance if several parallel implementations are required, for instance in the Parallel Packet Classification scheme. The algorithm is described further in section 7.2. The description is based on results from [108]. Further improvements to speed up the lookup operation is given in section 7.3. The performance of the proposed algorithms is compared to trie based approaches in section 7.4. Finally, in section 7.5, concluding remarks are given.

7.2. Algorithm

The IP lookup algorithm organises the IP prefixes in a tree structure. Each node in the tree contains exactly one prefix so the size of the tree is equal to the size of the routing table. Consider the routing table given in Table 8. It contains seven different prefixes belonging to seven different routes. Note that several IP addresses will match both e.g. R5 and R6 and in this case, the longest match i.e. R6 is taken.

Figure 33 shows the tree structure for the routing table given in Table 8. Each node contains a prefix and two pointers pointing to successive tree nodes. The prefix size must be greater than or equal to the level where the prefix is located. E.g. the prefix size of '11*' is two, and the level is 1.

in Figure 33. Note that the prefix-tree in Figure 33 is only one out of several possible trees based on Table 8.

Table 8: Routing table example

Prefix	Route
01*	R1
11*	R2
0*	R3
001*	R4
10*	R5
1000*	R6
1010*	R7

The remainder of this section explains the search, insert and delete operations by examples followed by a pseudo code description.

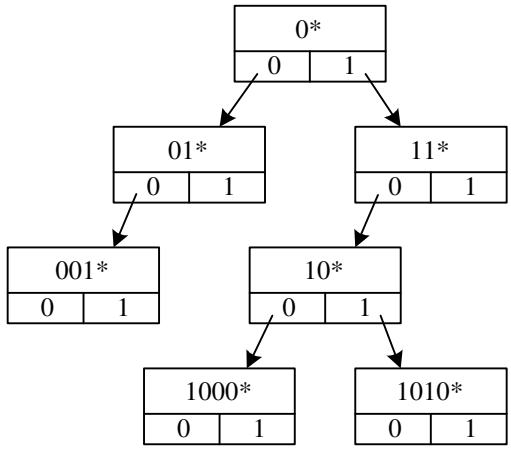

Figure 33: Prefix-tree

7.2.1. Lookup operation

The example below explains the lookup operation. Assume that '10101010' is used as input to the lookup tree. The lookup procedure works as follows: The first step is to compare the address with the prefix in the root node. The root node prefix does not match the address so the

result of the comparison is not stored. The first bit in the address is then used to determine the next node in the tree. The first bit is a '1' so the next visited node is '11*'. Again, there is no match between the prefix '11*' and the address '10101010'. The search procedure is continued using the next bit in the address, which is a '0'. This leads to the tree node containing prefix '10*'. Now there is a match between the prefix and the address. Since this is the longest prefix match until now, the R5 route is stored. The next address bit is '1', which implies that the next visited node is '1010*'. Again, there is a match between the address and the prefix. This is the longest prefix match until now so the route R7 is stored and replaces R5. Since '1010*' is a leaf node, the lookup operation has finished. The longest prefix match is route R7.

The time complexity of the lookup operation is O(B) where B is the number of address bits. The pseudo code for the lookup operation is shown below. '**Ref**' means that the variable is called by reference. 'pmax' is initially set to 0, and a match is found if 'pmax' is larger than 0 when the call returns. The procedure call is as follows: *lookup* (addr, 0, root, &pmax, &route)

```
lookup(addr,level,node,ref pmax,ref route) {

 if (node == NULL)
   return

 p = prefix_match_size(addr,prefix(node))

 if (p > pmax) {
   pmax = p
   route = route(node)
 }

 if (addr[level] == '1')
   lookup(addr,level+1,right(node),pmax,route)
 else
   lookup(addr,level+1,left(node),pmax,route)
}
```

7.2.2. Insert Operation

Now the insert procedure will be illustrated by an example. The resulting tree is shown in Figure 34. It is assumed that a new route R8 with prefix '1*' will be added to the tree. The first step is to examine the root node. The level of the root node is 0, which is smaller than the prefix length of one for route R8. The first bit of the new prefix is used to determine the next node in tree, which is '11*'. Now the level is 1, which is equal to the prefix length of R8. The new node must then be inserted at this location since the level in the tree must never be higher than the prefix length. The second prefix bit of the old prefix '11*' is 1, and the old node is therefore inserted to the right of '1*'.

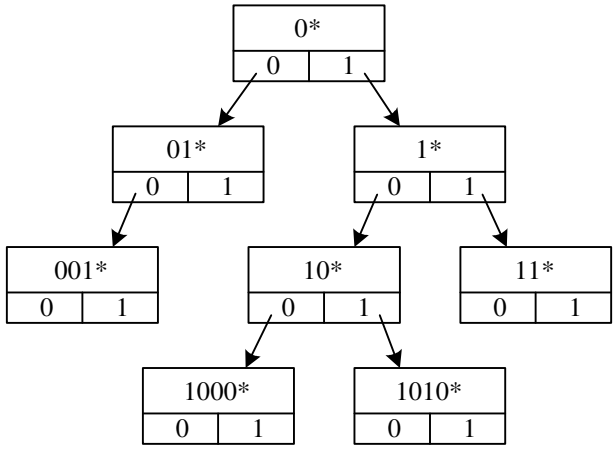

Figure 34: Prefix-tree after addition of '1*' prefix

The time complexity of the insert procedure is O(B). The pseudo code is shown below. The procedure call is as follows: *insert*(prefix,route,0,&root).

```
insert(prefix,route,level,ref node) {

 if (node == NULL) {
   node = new node(prefix,route)
   return
 }
```

```
  if (length(prefix) == level) {

    p1 = prefix(node)
    r1 = route(node)
    prefix(node) = prefix
    route(node) = route
    if (p1[level] == '1')
      insert(p1,r1,level+1,right(node))
    else
      insert(p1,r1,level+1,left(node))

  }
  else {
    if (prefix[level] == '1')
      insert(prefix,route,level+1,right(node))
    else
      insert(prefix,route,level+1,left(node))
  }
}
```

7.2.3. Delete Operation

The following example shows how a node is removed. The remove operation is shown in Figure 35. The selected node is '1*', which was just inserted in the previous example. A node without child nodes can easily be removed. If the node has one or two child nodes, it is necessary to reconstruct the tree. This can be done by finding a leaf node in one of the child trees and then insert this node at the location, which was just removed. The selected leaf node is '1000*', which is now inserted as the right child of the root node.

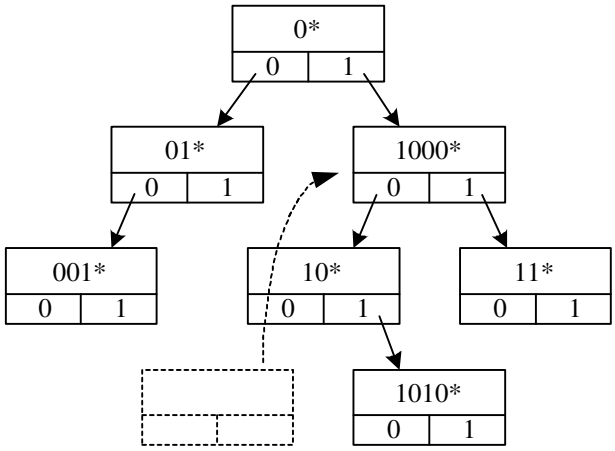

Figure 35: Prefix-tree after removal of '1*' prefix.

The time complexity of the delete procedure is O(B). The pseudo code is shown below. The procedure call is as follows: *delete*(prefix,&root,0).

```
delete(prefix,ref node,level) {

 if (prefix == prefix(node)) {

  leaf_node = find_leaf_node(node)
  if (leaf_node == NULL) {
   delete(node)
   node = NULL
   return
  }
  else {
   right(leaf_node) = right(node)
   left(leaf_node) = left(node)
   delete(node)
   node = leaf_node
   return
  }
 }
 else {
```

```
    if (prefix[level] = '1')
      delete(prefix,right(node),level+1)
    else
      delete(prefix,left(node),level+1)
  }
}
```

7.3. Improvements

The following section describes a number of improvements to speed up the lookup operation. The first approach shown in Figure 36 uses an additional level field in each tree node. The level value determines the next bit in the address used for searching. A similar technique is utilised for path-compressed tries.

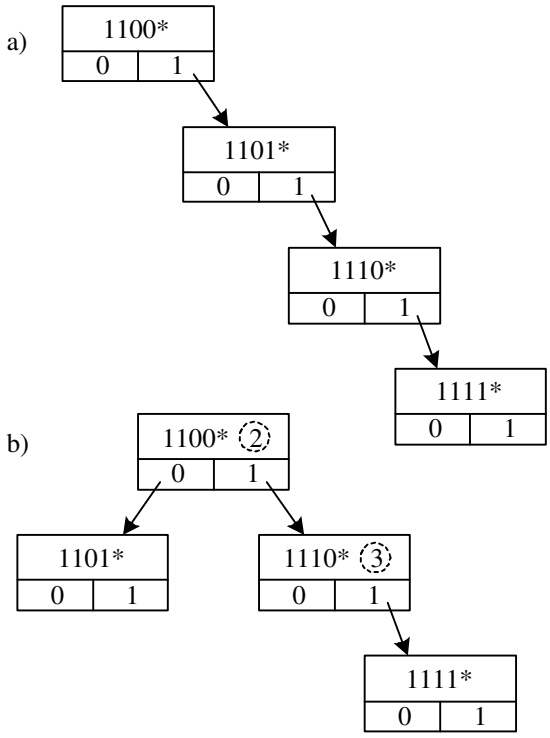

Figure 36: a) Prefix-tree, b) Level-tree

Figure 36a shows four prefixes stored in a prefix-tree. The resulting level-tree is shown in Figure 36b. Note that the root node has level = 2, which

means that bit number 2 (third bit from the left) in the address determines the child node. In this example, the level-tree has fewer levels (3) compared to the prefix-tree (4). However, in general, the worst-case tree height is not reduced by the level-tree.

Introducing a lookup table that provides shortcuts into the tree can reduce the maximum number of levels. This is suggested for tries in [109], but can be exploited for prefix-trees as well. Assume that the number of bits in the address is B. The basic tree in Figure 33 has at maximum B levels. An example is shown in Figure 37 with B = 8 bits.

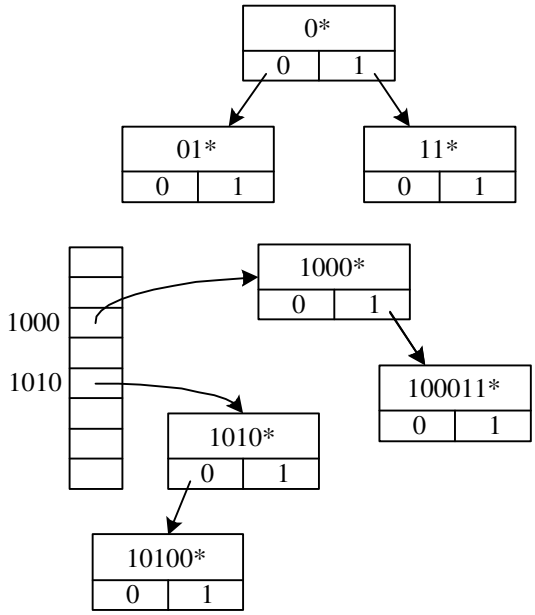

Figure 37: Prefix-tree with shortcut table

The first B/2 bits of the address are used as index to the lookup table. If the entry contains a valid pointer, it will point to the sub-tree that will be traversed. If the entry does not contain any valid pointer, the basic tree will be used. The basic tree contains prefixes between 0 and B/2. With the introduction of a shortcut table, the maximum search depth is reduced to B/2. IPv4 prefixes requires a lookup table of size 16 bits giving 65536 locations.

The lookup time can be further improved by increasing the number of lookup tables. In case of Ipv4, the address space of 32 bits can be subdivided into 4 regions with 8 bit in each. Having three lookup tables, the first table will examine the first 8 address bits, the second table will

use the first 16 bits, and the third table will use the first 24 bits. The maximum depth of each sub-tree will then become 8 levels. The main drawback is the size of the third lookup table with 24 address bits. However, the number of prefixes with length above 24 is very limited according to Table 9 [110], and it is therefore more efficient to store the prefixes in a CAM.

Search speed can be further improved by increasing the number of leaf nodes. With four leaf nodes, it is necessary to examine two address bits to find the next node in the tree. Each node must now store more than one prefix. E.g. the prefixes '*', '0*' and '1*' must be stored in the same node because nodes at a higher level will contain at least two bits more in the prefixes. Figure 38a shows a new node. The size of the new node is at least twice as big because it contains 3 prefixes and 4 pointers. If the node is too big for a single memory line, it can easily be split into two consecutive memory lines. The content of the two memory lines is shown in Figure 38b. The first bit among the two bits used for searching is then used to determine the memory line. Note that the prefix '*' must be repeated in both lines because it matches addresses starting with both '0' and '1'.

A multi-bit node may contain between 1 and 3 prefixes so the total number of prefixes that can be stored is not directly given by the total amount of memory measured in number of nodes, which is the case for the basic prefix-tree. The worst-case can, however, be determined. It occurs when the tree is balanced and when all leaf nodes contain only one prefix. Assume that the highest level in the tree contains l nodes and that the total number of nodes above that level is m. It can be shown that $m=3l+1$ for a four child node. The total number of prefixes is $p=3l+m$, and the total number of nodes is $n=l+m$, thus $p=2.5n-0.5$. The smallest number of prefixes that always can be stored for a given memory space n is thus the integer part of $2.5n-0.5$.

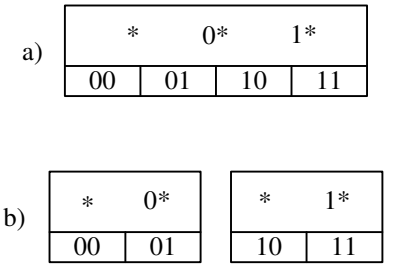

Figure 38: Multi-bit node

Using both tree shortcut lookup tables and larger 2-level nodes, the lookup operation can be performed in 4-5 memory cycles. Using synchronous SRAM with 10 ns access time, the lookup time will be in the range 40-50 ns. This is sufficient for 10 Gbit Ethernet with minimum packet duration of 51.2 ns.

7.4. Performance

This section compares the performance in terms of tree size of the prefix-tree and level tree with a basic trie and a path compressed trie (patricia). The routing table is from the Mae-West router 03/15/02 [110]. It contains 29587 prefixes. Table 9 shows the number of prefixes at each level of the trie/tree. The prefix distribution is depicted graphically in Figure 39.

Table 9: Prefix distribution

LEVEL	TRIE	PATRICIA	PREFIX TREE	LEVEL TREE
0	0	0	1	1
1	0	0	2	2
2	0	0	4	4
3	0	1	7	7
4	0	0	11	14
5	0	1	20	28
6	0	9	32	53
7	0	15	52	95
8	10	19	92	162
9	3	40	165	286
10	311	49	295	526
11	4	114	535	958
12	14	377	917	1575
13	37	930	1403	2366
14	68	1735	1902	3129
15	128	2568	2430	3801
16	2339	3262	2932	4234
17	531	4030	3424	4175
18	887	4397	3638	3501

19	2248	4309	3426	2550
20	2041	3487	2893	1451
21	1399	2568	2324	552
22	1971	1334	1679	106
23	2253	337	1057	11
24	15627	4	346	0
25	15	1	0	0
26	2	0	0	0
27	2	0	0	0
28	1	0	0	0
29	1	0	0	0
30	2	0	0	0
31	0	0	0	0
32	1	0	0	0

Table 10 shows the total number of nodes and the number of dummy nodes. A dummy node is an intermediate node that does not contain any routing information. The average and maximum height does not differ much for the patricia trie and prefix-tree, but the number of nodes is almost twice as high for the patricia trie. Note that the level-tree gives a small reduction in the average and maximum height compared to the prefix-tree. However, it is expected that the advantage of level-tree is larger for a more hierarchically organised prefix database e.g. in combination with IPv6.

Table 10: Comparison

NODE	TRIE	PATRICIA	PREFIX TREE	LEVEL TREE
Total	111747	56295	29587	29587
Dummy	82160	26708	0	0
Av. Height	22.0	17.7	17.4	15.7
Max height	32	25	24	23

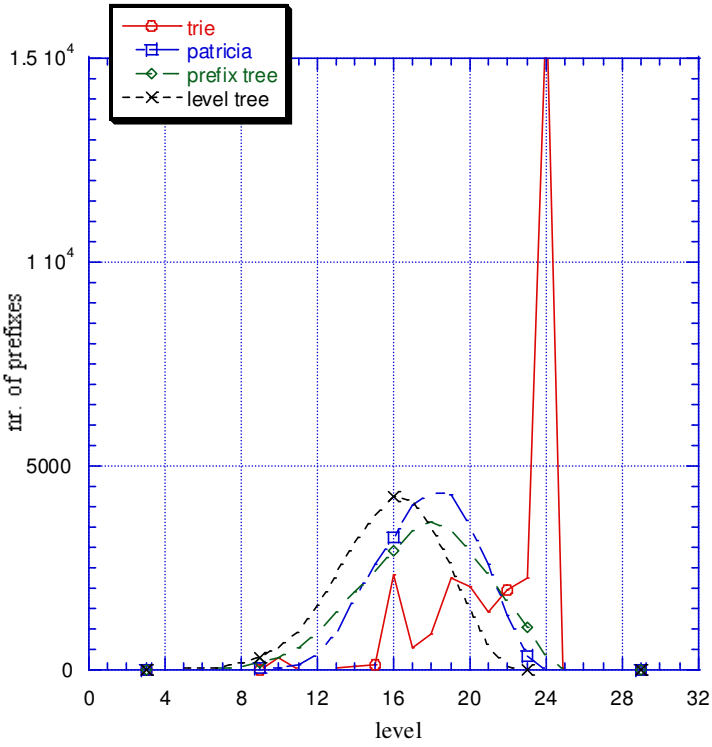

Figure 39: Prefix distribution

7.5. Summary

The first part of this chapter described the IP lookup and classification problem. IP lookup requires Longest Prefix Match search because of the routing table organisation specified by Classless Inter Domain Routing. Classification is a more difficult task because it requires inspection of several packet header fields. On the other hand, the number of rules is typically lower than the number of entries in the routing table so a fast TCAM based solution is feasible.

Trie-based IP lookup schemes have several advantages including simple procedures for update and lookup. The time complexity of the lookup operation is, however, proportional to the number of address-bits so a basic trie structure is not scalable to gigabit speed. Using multi-bit tries and shortcut lookup tables can increase the lookup speed. In terms of memory usage, the main drawback of the trie approach is the high number of intermediate nodes. Even for the patricia trie, the number of

intermediate nodes is almost as high as the number of entries in the forwarding table.

The proposed prefix-tree overcomes this drawback by combining a tree and a trie such that each node contains exactly one prefix. The memory requirement for the prefix-tree is therefore exactly given by the size of the forwarding table. This chapter also proposed the level-tree that has slightly better performance than the prefix-tree, but in order to increase the lookup speed significantly, shortcut lookup tables and multi-bit nodes can be introduced. Using both tree shortcut lookup tables and larger two-bit nodes, the lookup operation can be performed in 4-5 memory cycles.

8. Conclusion

Packet switching is the basis of many networking technologies, e.g. IP and ATM, and packet switching is surely the future technology of data networking. The interesting question is thus: what direction will the technology take, and how fast will it go? As discussed in the introduction of this thesis, the switch nodes could evolve towards many small, distributed switch units or towards fewer large, centralised switch units. This thesis has the latter scenario in focus. It is assumed that larger switch nodes with high link- and aggregated capacity are needed in the future. It is therefore necessary to invent new concepts and solutions, e.g. optical packet switching, to reach the required capacity.

The EU-funded IST project DAVID is a central part of this thesis. The objective of the DAVID project was to propose and demonstrate feasible architectures for optical packet switching, both in the MAN as well as in the WAN. Two different approaches for contention resolution were taken for the MAN and WAN, respectively; the MAN uses a MAC protocol to control access to the ring, whereas the WAN uses fibre-delay lines for contention resolution. Both the MAN and WAN were described in chapter 3. The main focus in the MAN part was on the ring node and the generation of optical slots from variable length client layer packets. A statistical model was presented, and the derived analytical results for the average waiting time were exploited to determine an optimal value of the timeout parameter. The second part introduced the hierarchical DAVID WAN network. It was shown that the concepts of MPLS could be utilised to create a unified routing and switching approach covering all technologies. Finally, a benchmarking study was carried out to compare the power consumption of electrical and optical switch nodes. It was concluded that optical technology significantly reduces power consumption, and that power consumption puts a limit on the scalability of electrical switch nodes. Furthermore, this also explains the increasing interest for optical technology in the switch backplane of electrical switch systems.

Chapter 4 considered MPLS traffic engineering and related protocols. The path selection procedure for traffic engineering can be classified as either off-line or on-line. Off-line traffic engineering is used in cases where the traffic demands are known beforehand, whereas on-line demands are routed at request time. A traffic engineering scheme for best effort TCP/IP traffic was proposed. The basic idea is to group TCP

connections into trunks, and then route the trunks based on MPLS traffic engineering procedures. Finally, constrained based routing was examined in more detail. A heuristic formula for the link weight was shown to improve network utilisation. The link weight takes both transmission and queuing delay into account, thereby increasing the cost of heavy loaded links.

The remaining chapters, chapter 5, 6 and 7, moved focus from networking issues to internal node design. Also, these chapters consider only electrical packet switching. Electrical packet switching is believed to be sufficient for the years to come; at the moment, switching systems with 10 Gbit/s interface speed and aggregated capacity of several hundreds of gigabits per second are available.

Chapter 5 introduced a multistage/multipath packet switch fabric where bundled packets are transmitted over identical parallel planes. This was proposed as an alternative to traditional load-balancing schemes that require packet re-sequencing at the outputs. A simple scheduling algorithm was proposed that applies time stamps to arriving packets and serves packets in order of increasing time stamp values. Worst-case scheduling delay and buffer occupancy were derived for this specific scheduling algorithm. The proposed scheduling algorithm performs bundling efficiently and at the same time, bounded delay is provided. Simulation results demonstrated that the actual delay for different distributions is much smaller than the derived worst-case value. The time-stamp scheduling algorithm was compared to round-robin and Longest Queue First scheduling, and simulation results indicated that the time-stamp algorithm showed the best overall performance even though the Longest Queue First scheduler provided the lowest average delay. Finally, speedup was discussed. The speedup can be reduced by introducing a different number of en-queue and de-queue timeslots in a period. The impact from speedup on the worst-case delay was calculated.

Chapter 6 introduced a new architecture for a buffered crossbar, which uses two levels of backpressure to reduce the amount of memory used in the switch card. The proposed switch uses a small, shared VOQ memory in combination with a one-cell deep crosspoint buffer. Each shared queue system uses independent schedulers so the time complexity of the arbitration is identical to that of a pure crossbar buffered switch card. Buffered crossbars have several advantages compared to non-buffered crossbars including simpler arbitration, synchronisation relaxation and improved performance. The main drawback, however, is the total amount of crossbar memory, which is proportional to the square of the number of input/output ports and backpressure latency, and this

scalability issue is solved by the proposed architecture. The performance has been investigated by a simulation study. It was shown that the amount of memory is reduced significantly with only a small reduction in performance. The switch was insensitive to the burstiness of traffic, and the study shows that a reduction of 75 % in memory could be obtained for a 32x32 switch with a backpressure round trip time of 4 timeslots. The performance reduction can be compensated by an additional speedup of 10 %, or it can be compensated by an internal speedup of 2 between the shared input buffer and the crosspoint buffers. It is also expected that the memory savings will become even higher in a switch containing several priorities.

Finally, chapter 7 described the IP lookup and classification problem. IP lookup requires longest prefix match searching because of the routing table organisation specified by Classless Inter Domain Routing. Trie-based IP lookup schemes have several advantages including simple procedures for update and lookup. The time complexity of the lookup operation is, however, proportional to the number of address-bits so a basic trie structure is not scalable to gigabit speed. Using multi-bit tries and shortcut lookup tables can increase the lookup speed. In terms of memory usage, the main drawback of the trie approach is the high number of intermediate nodes. Even for the patricia trie, the number of intermediate nodes is almost as high as the number of entries in the forwarding table. The proposed prefix-tree overcomes this drawback by combining a tree and a trie such that each node contains exactly one prefix. The memory requirement for the prefix-tree is therefore given exactly by the size of the forwarding table. This chapter also proposes the level-tree that has slightly better performance than the prefix-tree, but in order to increase the lookup speed significantly, shortcut lookup tables and multi-bit nodes can be introduced. Using both tree shortcut lookup tables and larger two-bit nodes, the lookup operation can be performed in 4-5 memory cycles.

Based on the work done and referred to in this thesis, a possible and likely evolution scenario for packet switching networks can be proposed. In the years to come, electrical packet switching will be dominant. It is still possible to meet an increasing capacity demand, e.g. by introducing the architectures proposed in chapter 5 and 6. However, at some point, the capacity will be limited mainly by the possible level of integration, which is restricted by power consumption as shown in chapter 3. The next step could be hybrid solutions, i.e. switches with electrical interfaces and optics in the switch backplane. This approach eliminates the need for optical buffering, wavelength conversion and 3R regeneration (functionality that is required in a pure optical packet switched network),

and is therefore feasible to implement, even today. In the very long term, optical packet switching might be introduced. As demonstrated by the DAVID project, this is possible if a sufficiently pragmatic approach is taken. The DAVID MAN is even feasible on a shorter timescale since having a global MAC protocol eliminates the problem of optical buffering.

9. References

[1] T. Fjelde, D. Wolfson, A. Kloch, C. Janz, A. Coquelin, I. Guillemot, F. Gaborit, F. Poingt, B. Dagens and M. Renaud. "Novel scheme for efficient label-swapping using simpel using XOR gate" *European Conference on Optical Communication (ECOC 2000)*, Paper no. 10.4.2, pp. 63-64, Munich, Germany (2000).

[2] H. Wessing, H. Christiansen, T. Fjelde, and L. Dittmann. "Novel scheme for packet forwarding without header modification in optical networks" *IEEE Journal of Lightwave Technology*, vol. 20 (8), pp.1277-1283 (2002).

[3] H. Christiansen, T. Fjelde, H. Wessing, "Novel label processing schemes for MPLS", *Optical Networks Magazine*, Vol. 3, Number 6, November/December 2002.

[4] D. K. Hunter, M. H. M. Nizam, M.C. Chia, I. Andonovic, K. M. Guild, A. Tzanakaki, M. J. O'Mahony, L. D. Bainbridge, M. F. C. Stephens, R. V. Penty, I. H. White, "WASPNET: a wavelength switched packet network", *IEEE Communications Magazine*, Vol.37 Issue.3, Page no. 120-129, 1999

[5] T. Koonen, G. Morthier, J. Jennen, H. de Waardt, P. Demeester,"Optical packet routing in IP-over-WDM networks deploying two-level optical labeling", *ECOC 2001 proceedings*, October 2001, Amsterdam, The Netherlands.

[6] N. Le Sauze, D. Chiaroni, O. Rofidal, A. Dupas "New optical packet synchronizer for optical packet routers" *PIS'2001 proceedings*, Monterey, USA , June 2001

[7] B. Lavigne "All-Optical 3R Regeneration based on Semiconductor Technology", *PIS'2002 proceedings*, Cheju Island, Korea, July 2002.

[8] C. Bornholdt, S. Bauer, M. Mohrle, H.-P. Nolting, B. Sartorius, "All optical clock recovery at 80 GHz and beyond" 2001. *ECOC'01.*

[9] H. Wessing, B. Sørensen, B. Lavigne, E. Balmefrezol, O. Leclerc, "Combining control electronics with SOA to equalize packet-to-packet power variations for optical 3R regeneration in optical networks at 10 Gbit/s". *OFC 2004.*

[10] Shun Yao, B. Mukherjee, , S. Dixit, "Advances in photonic packet switching: an overview", *IEEE Communications Magazine*, Issue Vol.38 Issue.2, pp. 84-94,2000

[11] P. Gambini, M. Renaud, C. Guillemot, F. Callegati, I. Andonovic, B. Bostica, D. Chiaroni, G. Corazza, S. L. Danielsen, P. Gravey, P. B. Hansen, M. Henry, C. Janz, A. Kloch, R. Krahenbuhl, C. Raffaelli, M. Schilling, A. Talneau, L. Zucchelli, "Transparent optical packet switching: network architecture and demonstrators in the KEOPS project", *IEEE Journal on Selected Areas in Communications*, Vol.16 Issue.7, Page no. 1245 -1259, 1998.

[12] S. L. Danielsen, B. Mikkelsen, C. Jørgensen, T. Durhuus, K. Stubkjær, "WDM Packet Switch Architectures and Analysis of the Influence of Tunable Wavelength Converters on the performance", *IEEE Journal of Lightwave Technology*, February 1997

[13] D. Chiaroni et al "First demonstration of an asynchronous optical packet switching matrix prototype for Multi-Terabit-class routers/switches", *ECOC 2001 proceedings*, October 2001

[14] C. Develder, J. Cheyns, E. Van Breusegem, E. Baert, A. Ackaert, M. Pickavet, P. Demeester, "Node Architectures for Optical Packet And Burst Switching", *COIN + PS2002*, July 2002, Cheju Island, Korea.

[15] Network Processing Forum, *http://www.npforum.org/*, March 2003.

[16] Network Processing Forum, "Streaming Interface (NPSI) Implementation Agreement", October 18, 2002 Revision 1.0.

[17] Optical Internetworking Forum, *http://www.oiforum.com/*, March 2003.

[18] H. J. Chao, C. H. Lam, E. Oki, "Broadband Packet Switching Technologies", Wiley, 200, ISBN 0-471-00454-5

[19] Rainer Händel, Manfred N. Huber, Stefan Schröder, "ATM Networks", *Addison-Wesley*, 1994.

[20] C. Minkenberg, T. Engbersen "A Combined Input and Output Queued Packet-Switched System Based on PRIZMA Switch-on-a-Chip Technology. *IEEE Communications Magazine*, Issue Vol.38 Issue.12, December 2000.

[21] N. McKeown, A. Mekkittikul, V. Anantharam, J. Walrand, "Achieving 100% throughput in an input-queued switch", *IEEE Transactions on Communications*, Vol.47 Issue.8 1999.

[22] T. E. Anderson, S. S. Owicki, J. B. Saxe, C. P. Thacker, "High-speed switch scheduling for local-area networks", ACM

Transactions on Computer Systems, Vol.11 Issue.4, pp 319-352, 1993.

[23] N. McKeown, "Scheduling algorithms for Input Queued Switches", Ph.D. thesis, UC Berkeley, 1995.

[24] J. G. Dai, B. Prabhakar, "The throughput of data switches with and without speedup", *Proceedings IEEE INFOCOM 2000*, p556-64 vol.2 2000.

[25] F. M. Chiussi, J. G. Kneuer, V. P. Kumar, "Low-cost scalable switching solutions for broadband networking: the ATLANTA architecture and chipset" *IEEE Communications Magazine*, 35(3).

[26] S. Dong and C. Phillips, "Adaptive Segment Path Restoration (ASPR) in MPLS Networks", *IFIP & IEEE Net-Con' 2002*, October 2002.

[27] B. Jamoussi, L. Andersson, R. Callon, R. Dantu, L. Wu, P. Doolan, T. Worster, N. Feldman, A. Fredette, M. Girish, E. Gray, Sandburst, J. Heinanen, T. Kilty, A. Malis, "Constraint-Based LSP Setup using LDP", *Internet Engineering Task Force* RFC3212, January 2002

[28] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", *Internet Engineering Task Force* RFC3209, December 2001.

[29] S. L. Danielsen, C. Jørgensen, B. Mikkelsen, K. Stubkjær, "Optical Packet switched Network Layer Without Optical Buffers", *IEEE Photonics Technology Letters*, June 1998

[30] Z. Zhang, Y. Yang, "Performance Modeling of Bufferless WDM Packet Switching Networks with Wavelength Conversion", *Globecom 2003*, San Fransisco, December 2003

[31] Incorporating Optics into Internet Routers, http://klamath.stanford.edu/or/index.html.

[32] D. Chiaroni, A. Jourdan, G. Eilenberger, D. Verchere, F. Masetti, T. Atmaca, G. Hebuterne. "Feasibility and Performance issues of edge and core routers for the next generation of optical IP networks", *PIS'2002 proceedings*, Cheju Island, Korea, July 2002.

[33] A. Smiljanic "High-capacity packet-switched fabrics: introduction to the focus issue", *The Journal of Optical Networking*, Vol. 2, No. 7 - July 2003

[34] M. Hamdi, H. J. Chao, D. J. Blumental, E. Leornardi, Chumming Qiao, K. Y. Yun, R. Ramaswami, "Guest editorial high-performance optical switches/routers for high-speed internet", IEEE Journal on

Selected Areas in Communications, Volume: 21, Issue: 7, Sept. 2003

[35] L. Dittmann (editor), C. Develder, D. Chiaroni, F. Neri, F. Callegati, W. Koerber, A. Stavdas, M. Renaud, A. Rafel, J. Solé-Pareta, W. Cerroni, N. Leligou, Lars Dembeck, B. Mortensen, M. Pickavet, N. Le Sauze, M. Mahony, B. Berde, G. Eilenberger, "The European IST Project DAVID: a Viable Approach towards Optical Packet Switching", *JSAC Special Issue*, 2003.

[36] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "An Architecture for Differentiated Services",IETF RFC 2475, December 1998.

[37] E. Mannie (editor) "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", Internet Draft, Expiration date: Nov. 2003, draft-ietf-ccamp-gmpls-architecture-07.txt.

[38] L. Dittmann, D. Chiaroni, "DAVID - an approach towards MPLS based optical packet switching with QoS support", *PS2001 proceedings*, Monterey, USA, June 2001.

[39] S. Yao, F. Xue, B. Mukherjee, S. J. B. Yoo, S. Dixit "Electrical ingress buffering and traffic aggregation for optical packet switching and their effect on TCP-level performance in optical mesh networks", *IEEE Communications Magazine*, Vol.40 Issue.9, 2002.

[40] C. Lam, D. Simeonidou, "Optical Packet Switch Modelling and its Traffic Shaping Effects", *European Conference on Optical Communication (ECOC)*, Copenhagen, 2002.

[41] J. Angelopoulos, H. C. Leligou, H. Linardakis, A. Stavdas. "A QoS-sensitive MAC for slotted WDM metropolitan rings", *ONDM-Optical Networking Design and* Modelling, Torino, 2002,.

[42] A. Bianco, G. Galante, E. Leonardi, F. Nero, M. Rundo "Access Control Protocols for Interconnected WDM Rings in the David Metro Network" *Tyrrhenian International Workshop on Digital Communications*. 2001,.

[43] H. Linardakis, N. Leligou, S. Zontos, A. Stavdas, "Controlling high speed slotted data channels in WDM metro rings", *High Performance Switching and Routing*, Torino, 2003.

[44] B. B. Mortensen, M. S. Berger, "Optical Packet Switching Demonstrator", *European Conference on Optical Communication (ECOC)*, Copenhagen, 2002.

[45] B. B. Mortensen, M. S. Berger "Optical Packet Switched Demonstrator" *PS'2002 proceedings*, Cheju Island, Korea, July 2002.

[46] B. B. Mortensen, M. S. Berger, H. Linardakis, R. Jociles-Ferrer, "Metropolitan Area Network Optical Packet Switch Demonstrator", *proceedings of IST 2003*, Isfahan.

[47] M. S. Berger, V. B. Iversen, B. B. Mortensen, "Analytical performance evaluation of optical packet network interface", *COIN 2003*, Melbourne, Australia, July 2003.

[48] V. B. Iversen, "Data- og teletrafikteori", ISBN 87-7381-079-7, Lyngby 1999.

[49] B. B. Mortensen, M. S. Berger, "Estimating timeout parameters for packet aggregation" COIN2003, Melbourne, Australia, July 2003.

[50] M. S. Berger, B. B. Mortensen, V. B. Iversen, R. Jociles-Ferrer, "Evaluation of Delay Bound for QoS provisioning in Optical Packet Network Interface", *7th WSEAS International Conference on Communications*, Corfu, Greece, July 2003

[51] L. Dittmann, H. Christiansen, M. S. Berger, "Hierarchical MPLS – An approach for efficient resource administration in multi-technology networks", *NOC 2001*, Ipswich, England, 2001.

[52] H. Christiansen, M. S. Berger, "Novel, hiearachical, MPLS-based network architectures and their role in migration strategies towards future, optical, packet switched networks", *CIIT 2002*, St. Thomas, USVI, November 2002

[53] M. Berger, H. Christiansen, B. Mortensen, R. Jociles-Ferrer, "Hierarchical Electro-optical Packet Network Architecture", *proceedings of IST 2003*, Isfahan.

[54] C. Develder, M. Pickavet, P. Demeester, "Strategies for an FDL Based Feed-Back Buffer for an Optical Packet Switch with QoS Differentation" *PS'2002 proceedings*, Cheju Island, Korea, July 2002.

[55] N. Le Sauze, A. Dupas, E. Dotaro, L.Ciavaglia, M.H.M Nizam, A. Ge, L. Dembeck, "A novel, low cost optical packet metropolitan ring architecture", *European Conference on Optical Communication (ECOC),* Copenhagen 2002.

[56] N. Le Sauze, E. Dotaro, L. Ciavaglia, A. Dupas, D. Chiaroni, A. Ge, M.H.M Nizam, K.Sridhar, L. Dembeck, W. Koerber, "DBORN: a shared WDM Ethernet bus architecture for optical packet metropolitan networks", *Photonics in Switching (PIS),* Cheju Island, Korea, 2002.

[57] IBM PowerPRS™ Q-64G Packet Routing Switch Datasheet.

[58] http://www.intel.com/design/network/products/npfamily/ixp2800.htm, July 2003.

[59] www.optillion.com, July 2003.

[60] H. Matsuda, R. Takeyari, K. Harada, H. Serizawa, "10 Gbit Optical Transmission Modules Supporting an Optical Network System. Hironari.", *Hitachi Review* Vol. 48 (1999), No. 4.

[61] N. Le Sauze, D. Chiaroni, M. Nord, M. S. Berger, J. F. Palacios, J. F. Lobo, D. Careglio, J. Solé-Pareta, S. Spadaro, A. Rafel, A. Hill, S. Sygletos, H. Skoufis, A. Stavdas, H. Lønsethagen, T. Olsen, F. Callegatti, F. Neri, A. Bianco, G. Galante, M. Mellia "Network concepts validation and benchmarking", *DAVID Deliverable D101(Public)*, December 2003.

[62] A. K. Parekhi, R. G. Gallager "A generalised processor sharing approach to flow control in integrated services networks. The single-node case" IEEE/ACM Transactions on Networking, Vol. 1 (1993),3, 344--357.

[63] M.S. Berger, V.B. Iversen. "Basic principles for MPLS traffic engineering" International Seminar on Telecommunication Networks and Teletraffic, pp. 23-30, St. Petersburg, Russia (2002).

[64] S. Blake, M. Carlson, D. Davies, Z. Wang, W. Weiss, "An architecture for differentiated services" *RFC-2475*, December 1998.

[65] R. Braden, D. Clark, S. Shenker, "Integrated services in the Internet architecture: an overview". *RFC-1633*, June 1994.

[66] K. M. Girish, B. Zhou, J-Q. Hu, "Formulation of the traffic engineering problems in MPLS based IP networks". *ISCC, Fifth IEEE Symposium on Computers and Communications*, 3--6 July, 2000. Proceedings pp. 214--219.

[67] www.ilog.com/products/oplstudio/, September 2003.

[68] IETF, Internet Engineering Task Force. RFC 2676. QoS Routing Mechanisms and OSPF Extensions. August 1999. 50 pp. http://www.ietf.org/rfc/rfc2676.txt

[69] M. J. O'Mahony, D. Simeonidou, D. K. Hunter, A. Tzanakaki, "The Application of Optical Packet Switching in Future Communication Networks". *IEEE Communications Magazine*, Vol. 39 (2001), 3, 128--135.

[70] S. N. Stepanov, V. B. Iversen, V. S. Lagutin, V. O. Kostrov, "Modelling issues of differentiated services in MPLS networks". *Seminar On Telecommunication Networks and Traffic Theory*. Saint-Petersburg, Russia, January 29 - February 1, 2002. 6 pp.

[71] J. Wroclawski, “Specification of the Controlled-Load Network Element Service”, RFC 1121, September 1997.

[72] S. Shenker, C. Partridge, R. Guerin, “Specification of Guaranteed Quality of Service”, RFC 1122, September 1997.

[73] K. Kar, M. Kodialam, T. V. Lakshman, "Minimum interference routing of bandwidth guaranteed tunnels with MPLS traffic engineering applications", IEEE Journal on Selected Areas in Communications, Issue Vol.18 Issue.12,December 2000, Page no. 2566 –2579.

[74] F. Le Faucheur, L. Wu, B. Davie, S. Davari, P. Vaananen, R. Krishnan, P. Cheval, J. Heinanen, “Multi-Protocol Label Switching (MPLS) Support of Differentiated Services”, *RFC 3270*, May 2002

[75] F. Le Faucheur, W. Lai, “Requirements for Support of Differentiated Services-aware MPLS Traffic Engineering”, RFC 3564, July 2003.

[76] W. Stallings, “High-Speed Networks”, *Prentice Hall*, *Upper Saddle River, NJ,* 1998, ISBN 0-13-525965-7

[77] J. Y. Hui, ” Switching and traffic theory for integrated broadband networks” Boston : Kluwer, 1990.

[78] M. Berger, “Multipath packet switch using packet bundling”, *Proceedings of HPSR 2002.*

[79] A. Herkersdorf, L. Heusler, E. Maehle ”Route discovery for multistage fabrics in ATM switching nodes” Performance evaluation 22 (1995).

[80] A. Varma, D. Stiliadis “Hardware Implementation of Fair Queuing Algorithms for Asynchronous Transfer Mode Networks”, IEEE Communications Magazine, December 1997.

[81] W. E. Denzel, A. P. J. Engbersen, I. Iliadis “A flexible shared-buffer switch for ATM at Gb/s rates” *Computer Networks and ISDN systems 27* (1995) 611-624.

[82] C. Minkenberg, T. Engbersen, “A combined input and output queued packet switched system based on PRIZMA switch on a chip technology” *IEEE Communications Magazine,* Dec. 2000.

[83] K. Kar, T. V. Lakshman, D. Stiliadis, and L. Tassiulas, “Reduced complexity input buffered switches,” in Proc. Hot Interconnects VIII, Stanford, CA, Aug. 2000.

[84] N. McKeown, “The iSLIP scheduling algorithm for input-queued switches”, *IEEE/ACM Transactions on Networking*, p 188 -201, Vol.7 Issue.2, 1999.

[85] R. B. Magill, C. E. Rohrs, R. L. Stevenson, "Output-queued switch emulation by fabrics with limited memory", *IEEE Journal on Selected Areas in Communications*, Volume: 21, Issue: 4, May 2003.

[86] S. T. Chuang, A. Goel, N. McKeown, B. Prabhakar, "Matching Output Queuing with a Combined Input Output Queued Switch", *IEEE Journal on Selected Areas in Communications*, vol.17, n.6, Dec.1999, pp. 1030-1039.

[87] M. Nabeshima, "Performance evaluation of a combined input- and crosspoint-queued switch", *IEICE Transactions on Communications*, Vol.E83-B Issue.3, p737-41, 2000

[88] T. Javidi, R. Magill, T. Hrabik, "A high-throughput scheduling algorithm for a buffered crossbar switch fabric", *Proceedings of IEEE ICC 2001*, p 1586-1591, vol.5.

[89] I. Radusinovic, M. Pejanovic, Z. Petrovic, "Impact of scheduling algorithms on performances of buffered crossbar switch fabrics", *Proceedings of IEEE ICC 2002*, p 2416-2420, Vol.4

[90] R. Rojas-Cessa, E. Oki, Z. Jing, H. J. Chao, "CIXB-1: combined input-one-cell-crosspoint buffered switch", *Proceedings of IEEE HPSR 2001*, p 324-329.

[91] L. Mhamdi, M. Hamdi, "MCBF: A High-Performance Scheduling Algorithm for Buffered Crossbar Switches", IEEE Communications Letters, Vol. 7, No. 9, September 2003.

[92] R. Rojas-Cessa, E. Oki, H.J. Chao, "CIXOB-k: combined input-crosspoint-output buffered packet switch", *Proceedings of IEEE GLOBECOM 2001*, p 2654-2660, Vol.4

[93] F. Abel, C. Minkenberg, R. P. Luijten, M. Gusat, I. Iliadis, "A Four-Terabit Single-Stage Packet Switch with Large Round-Trip Time Support", *10th Symposium on High Performance Interconnects HOT Interconnects*, p.5, 2002

[94] P. Gupta, N. McKeown, "Designing and implementing a fast crossbar scheduler", *IEEE Micro*, Vol.19 Issue.1, p 20-28, 1999.

[95] C. Minkenberg, "Performance of i-slip scheduling with large round-trip latency", *HPSR Workshop on High Performance Switching and Routing*, 2003

[96] Andreas Magnussen, "ATM Switching Systems", *COM, Technical University of Denmark,* Ph.D Thesis.

[97] M. Á. Ruiz-Sánches, E. W. Biersack, W.Dabbous, "Survey and Taxonomy of IP Address Lookup Algorithms", *IEEE Network*, March/April 2001

[98] P. Gupta, N. McKeown, "Algorithms for Packet Classification", *IEEE Network*, March/April 2001

[99] P. Gupta, N. McKeown, "Packet Classification on Multiple Fields", SIGCOMM, Vol.29, Issue.4, 1999

[100] V. Srinivasan, S Suri, G. Varghese, M. Waldvogel, "Fast and scalable Layer four Switching", *Proceedings of ACM Sigcomm*, 1998.

[101] M. Buddhikot, S. Suri, M. Waldvogel, "Space Dekomposition Techniques for fast layer 4 switching", *Protocols for High-Speed Networks*, USA, 1999

[102] J. Lunteren, T. Engbersen, "Fast and Scalable Packet Classification", *IEEE Journal on Selected Areas in Communication*, Vol. 21, No. 4, May 2003,

[103] www.netlogicmicro.com. Sep 2003.

[104] M. Zitterbart, "High-performance routing-table lookup", *Phil. Trans.*, Royal Soc. London A (2000) 258 p 2217-2231.

[105] R. Sedgewick, "Algorithms in C++", *Addison Wesley*, 1990

[106] S. Nilsson, G. Karlsson, "IP-address Lookup Using LC-tries", *IEEE JSAC,* June 1999, vol. 17 no.6, p 1083-92

[107] C. Labovitz, G. malan, F.Jahanian, "Internet routing instability", *ACM SIGCOMM '97*, September 1997

[108] M. Berger, "IP Lookup with fast update and low memory requirement", *Proceedings of HPSR 2003*, p

[109] M. Uga, K. Shiomoto, "A Fast and Compact Longest Prefix Lookup Method using Pointer Cache for very long Network Address", *proc. IEEE ICCCN 1999*, Boston MA, Oct 1999.

[110] Mae-West routing database, "The Internet performance Measurement and Analysis (IPMA) project", *http: //www.merit.edu/ipma/routing_table/*, 10 Oct. 2002