

Self-Management of Hybrid Optical and Packet Switching Networks

Tiago Fioreze

Graduation committee:

Chairman:	Prof. dr. ir. Anton J. Mouthaan
Promoter:	Prof. dr. ir. Boudewijn R. Haverkort
Assistant promoter:	Dr. ir. Aiko Pras

Members:

Prof. dr. Lisandro Z. Granville	Federal University of Rio Grande do Sul
Dr. hab. Olivier Festor	INRIA Nancy
Prof. dr. ir. Cees Th.A.M. de Laat	University of Amsterdam
Prof. dr. Antonio Liotta	Eindhoven University of Technology
Prof. dr. ing. Paul J.M. Havinga	University of Twente
Prof. dr. Hans van den Berg	University of Twente



CTIT Ph.D.-thesis Series No. 09-163

Centre for Telematics and Information Technology
University of Twente, P.O. Box 217, NL-7500 AE Enschede

ISSN 1381-3617

ISBN 978-90-365-2966-2

Publisher: Wöhrmann Print Service.

Cover photo credit: Rodolfo Clix.

Cover design: Tiago Fioreze.

Copyright © Tiago Fioreze 2010

SELF-MANAGEMENT OF HYBRID OPTICAL AND PACKET SWITCHING NETWORKS

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. H. Brinksma,
volgens besluit van het College voor Promoties,
in het openbaar te verdedigen
op woensdag 17 februari 2010 om 15.00 uur

door

Tiago Fioreze

geboren op 10 oktober 1979
te Tapera, Rio Grande do Sul, Brazilië

Dit proefschrift is goedgekeurd door:
Prof. dr. ir. Boudewijn R. Haverkort (promotor)
Dr. ir. Aiko Pras (assistent-promotor)

Acknowledgments

SINCE this thesis is not the fruit of a single person, I would like to express my sincere gratitude to those who, somehow or other, helped me throughout all my Ph.D. years and made this thesis possible.

First of all, I would like to thank to my supervisor, Dr. Ir. Aiko Pras, for his careful and patience guidance during all my study years. He gave me the opportunity to enroll as a Ph.D. student in the DACS group, which I am extremely grateful. I also would like to thank to Prof. Dr. Ir. Boudewijn R. Haverkort for his great advice and extreme professionalism. Not the least, I thank all DACS group members for having taken me and helped me during all these years.

I would like to thank the members of my graduation committee: Prof. dr. Lisandro Z. Granville, Dr. hab. Olivier Festor, Prof. dr. ir. Cees Th.A.M. de Laat, Prof. dr. Antonio Liotta, Prof. dr. ing. Paul J.M. Havinga, and Prof. dr. Hans van den Berg. I am really grateful for your time spent reviewing this thesis. Special thanks to Lisandro for his valuable help during the third year of my Ph.D. trajectory.

The flow analysis performed in this thesis would not have been possible without the cooperation of SURFnet, GÉANT, and the ICTS department of the University of Twente. Special thanks to Hans Trompert (SURFnet), Maurizio Molina (GÉANT), and Roel Hoek (ICTS) for their helpful contribution in the flow collection process. Last, but by no means least, I also thank my colleagues of the EMANICS project by their valuable collaboration.

I also extend my gratitude to all my friends who made my staying here in Enschede more enjoyable. Coffee breaks, parties in Macandra, Grolsch tours, social events, dinners or simply "*een biertje in de stad*", all these events would not be as much fun without you. With this respect, I would like to particularly mention Luiz Olavo, Eduardo Silva, Eduardo Zambon, Laura Daniele, Ricardo Neisse, Tom Broens, Rodrigo Pessoa, Anne Remke, Assed Jehangir, Jose Martinez, Remco van

de Meent, Wilma Hiddink, Idilio Drago, Luciana Bonino, Sharon van Sluys, Rafael Barbosa, Anna Sperotto, Giovane Moura, Desislava Dimitrova, Rick Hofstede, Marijn Jongerden, Fei Liu, Ramin Sadre, Marten van Sinderen, Luís Pires, Hailiang Mei, Ismênia Galvão, Patricia Costa, João Paulo, and so many other nice people I have met during all my Dutch years. The same gratitude applies to all UT-Kring Voetbal members, with whom I shared great football matches and tournaments.

I would like to show my special thanks to my “Honey”, Līga Vilmane, for all her patience, love and joy expressed during all these years we have been together. I specially thank you for your comprehension and support during the hardest moments of my Ph.D. trajectory. For you, my sincere and deepest gratitude.

Above all, I would like to thank my family for their immeasurable support over all these years far away from you. My greatest thanks go to my parents Arni e Sueli Fioreze and to my sister Marister Fioreze, who, despite the distance, showed me their tender heart and constant encouragement to achieve this Ph.D. degree.

Tiago Fioreze
Enschede, January 2010.

Abstract

Hybrid optical and packet switching networks are composed of multi-service hybrid devices that enable forwarding of data at multiple levels. Large IP flows at the IP level may be therefore moved to the optical level bypassing therefore the per hop routing decisions of the IP level. Such move could be beneficial since congested IP networks could be offloaded; leaving more resources for other smaller IP flows. At the same time, the flows switched at the optical level would experience better Quality of Service (QoS) thanks to larger bandwidth and negligible jitter. Moving these large flows to the optical level requires the creation of lightpaths to carry them. Currently, two approaches are used for that purpose: direct management and indirect management. With a direct approach, management messages are explicitly issued by the network manager to each managed device (*e.g.*, multi-service hybrid devices). Whereas with an indirect approach, messages are issued by the manager to one managed device that is in charge of signaling the other ones. In both approaches, the decision of which IP flows will be moved to lightpaths is although taken by network managers. As a result, only IP flows explicitly selected by such managers will take advantage of being transferred over lightpaths. However, it may be that there are also other large IP flows, not known to the manager, that could potentially profit from being moved to the optical level. The objective aimed in this Ph.D. thesis is at investigating the use of self-management principles in hybrid optical and packet switching networks in order to identify which IP flows should be moved to the optical level as well as establish and release lightpaths for such flows.

Contents

1	Introduction	1
1.1	Background	1
1.2	Related work on self-management	9
1.3	Motivation, scope, and objective	11
1.4	Research questions, and their research approaches	13
1.5	Thesis structure	15
2	Management approaches for hybrid networks	17
2.1	Conventional management approaches	17
2.2	Analysis of the conventional approaches	26
2.3	The self-management manifesto	27
2.4	Self-management of lightpaths	31
2.5	Concluding remarks	35
3	Monitoring of network traffic	37
3.1	Potential network parameters	38
3.2	Evaluation of identification parameters	40
3.3	Evaluation of behavior parameters	44
3.4	Possible techniques for monitoring IP data	54
3.5	The effects of sampling on elephant flows	62
3.6	Summary	73
4	Making autonomous decisions	75
4.1	Autonomic decision objective	76

4.2	Similarities with cache management	78
4.3	The autonomic decision process	81
4.4	Assumptions made	87
4.5	Validation of the decision policy	93
4.6	Grooming flows over lightpaths	96
4.7	Concluding remarks	100
5	The impact of self-managing lightpaths	103
5.1	The side effect of moving flows on the fly	104
5.2	Additional aspects	117
5.3	Summary	122
6	Conclusions	123
6.1	Overall conclusion	123
6.2	Future research	126
A	Statistical & mathematical background information	127
A.1	Decision trees	127
A.2	The CHAID algorithm	129
	Bibliography	131
	Glossary	145
	Acronyms	147
	Index	151
	About the author	153

Chapter 1

Introduction

This chapter starts by presenting some background information and related work, followed by the motivation and objective of this thesis. We then introduce the research questions addressed in this thesis and the respective approaches to answer them. Last, we finalize this chapter with a summarized structure of this thesis.

1.1 Background

THE Internet as we know today has proved to be a successful global network system, connecting billions of users worldwide. Notwithstanding its success, the simplicity of the Internet architecture has shown to be its “Achilles’ heel”, presenting some limitations that result in grand challenges to be addressed, such as vulnerability to attacks and scaling for more extreme dynamics [152]. In an effort to address these challenges, the networking community has been discussing the redesign of the Internet, the so called *Future Internet*. For that, two major fundamental approaches have been discussed [52] [11]: incremental approach and clean-slate approach. The former is the current approach used nowadays on the Internet and it consists of moving the Internet from one state to another through incremental patches. Whereas, the latter aims at a radical redesign of the current Internet architecture with new ideas applied from the scratch.

Although we do not know yet the details of how the *Future Internet* will look like, we can already foresee a future Internet in which optical communication infrastructures will play a major role. A tendency towards this prognosis has already been observed nowadays through the increasing change in the set of core technologies that form the Internet. Internet backbones that once relied solely on IP routing to deliver end-to-end communications are moving towards hybrid solutions that combine more than one networking technology, *i.e.*, towards hybrid networks.

In this thesis, we focus on hybrid networks that combine both IP and optical technologies. A hybrid IP and optical network is a network that can take data forwarding decisions simultaneously at both IP and optical levels [90]. These hybrid

networks are composed of intermediate multi-service hybrid devices that are both switches at the optical level and traditional routers at the IP level. In such an environment, IP flows can traverse a hybrid network through either a lightpath or a chain of routing decisions.

Concerning IP flows, we adopt the definition of an IP flow as a unidirectional sequence of packets that share the same properties [34]. On its turn, we consider a lightpath as a direct optical data connection over an optical fiber [94]. The lightpath can consist of the whole fiber, a wavelength within the fiber, or a TDM-based channel within the wavelength. Figure 1.1 depicts this lightpath hierarchy.

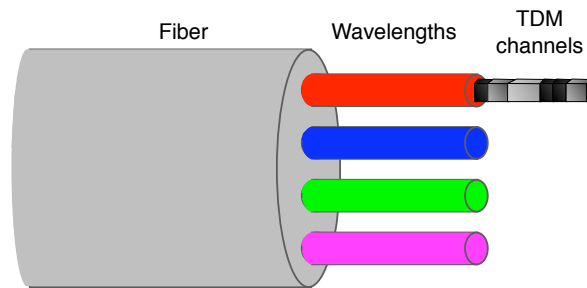


Figure 1.1: *The lightpath hierarchy.*

When IP flows are completely transported via lightpaths they bypass the per hop routing decisions of the IP level. As a result, the QoS offered by hybrid networks is considerably better when compared to traditional IP networks. Big IP flows that overload the regular IP level, for example, may be moved to the optical level where they experience better QoS (*e.g.*, negligible jitter and larger bandwidth). At the same time, the IP level is offloaded and can better serve smaller flows. Last but not least, it is also cheaper to send traffic at the optical level than at the IP level [38]. For the same traffic rate, the cost of an optical switch is $1/10^{th}$ of an Ethernet switch or $1/100^{th}$ of a conventional router.

In order to give an estimate of the amount of bandwidth that optical fiber communication makes possible nowadays, we highlight an optical transmission record that Alcatel-Lucent Bell Labs has recently set [2]. Researchers at Alcatel-Lucent Bell Labs have managed to multiplex 155 wavelengths, each one of them carrying 100 Gbps, over a 7.000 kilometers fiber (roughly the distance between Amsterdam and Minneapolis). That accounts for a transfer rate of 15.5 Terabits per second, which is equivalent to the transmission of 400 DVDs per second. If we also take into account that a single optical cable can accommodate many fibers, data transmission rates in the order of several Petabits per second can be reached.

With the increasing bandwidth demand by applications, such as high-definition television (HDTV) [8], grid computing [151], and large-scale scientific experiments (e.g., LOFAR project [39]), the importance of hybrid optical and packet switching networks is growing. Network providers, such as SURFnet [146], GÉANT [63], amongst others, are increasingly adopting hybrid networks. Collaboration among several of these providers aims at sharing optical capabilities, resulting in an interconnection of their research and education networks. By promoting this move towards hybrid networks, network providers may offer, therefore, new perspectives for services in hybrid networks.

1.1.1 SURFnet6 – an example of a hybrid network

SURFnet is the Dutch organization that develops, implements, and maintains the national research and educational network of the Netherlands. SURFnet is also responsible for managing SURFnet6, which is a hybrid optical and packet switching network composed of hybrid devices located at several cities in the Netherlands. Figure 1.2 shows the current SURFnet6 dark fiber topology. Dark fibers refer to unlit optical fibers, but available for use.



Figure 1.2: SURFnet6 dark fiber in The Netherlands.

SURFnet6 interconnects universities, research centers, polytechnics, academic hospitals, and scientific libraries in the Netherlands and also provides access for them to other networks worldwide. SURFnet6 hybrid devices are mainly optical switches that support native IPv4, IPv6, and lightpath provisioning over a single transmission infrastructure. With regard to its topology, SURFnet6 is composed of core switches, which are located in different places in Amsterdam, and by edge switches, which provide connection for access routers situated in the SURFnet6 users' domains. This enables, for instance, the transfer of huge amounts of data (e.g., 10 Gbps) coming from these domains through SURFnet6. The high capacity in terms of bandwidth existing in SURFnet6 is due to well-known multiplexing techniques and standards for connecting fiber-optic transmission systems, such as Dense Wavelength Division Multiplexing (DWDM) [178], Synchronous Digital Hierarchy (SDH) [79], and Synchronous Optical Networking (SONET) [5].

1.1.2 Multiplexing techniques and standards

Multiplexing is a process of combining multiple signals into one single signal over a shared medium. The multiplexing techniques most used in optical networks are Wavelength Division Multiplexing (WDM) [111] and Time-Division Multiplexing (TDM) [69]. The WDM technique consists of multiplexing multiple wavelengths over a single optical fiber. The amount of multiplexed wavelengths over a single fiber can divide WDM into Coarse WDM (CWDM) for fibers carrying less than 8 wavelengths and Dense WDM (DWDM) for those fibers carrying from 9 up to 160 wavelengths. On its turn, the TDM technique consists of interleaving portions of data streams in time, so that multiple data streams can be carried on a single transmission path. In optical networks terms, TDM consists of dividing a wavelength into time slots in order to send data frames. This approach forms the basis for today's standards used in digital communication, namely SDH and SONET.

SONET and SDH standards

SONET, which is standardized by the American National Standards Institute (ANSI), is a set of standards for synchronous data transmission over fiber optic networks that are often used for framing and synchronization at the physical layer. SONET is based on transmission at speeds of multiples of 51.840 Mbps. On its turn, SDH is the international version of the standard published by the International Telecommunications Union (ITU).

Table 1.1 shows the set of specifications of transmission rates in today's SDH and SONET networks. The highest rates that are commonly deployed are the OC-192 and OC-768 circuits, which operates at rates of 10 Gbit/s and 40 Gbit/s, respec-

Optical carrier level	Payload rate (Mbps)	Overhead rate (Mbps)	Total rate (Mbps)
OC-1	50.112	1.728	51.840
OC-3	148.608	6.912	155.520
OC-12	601.344	20.736	622.080
OC-24	1.202.208	41.472	1.243.680
OC-48	2.405.376	82.944	2.488.320
OC-192	9.621.504	331.776	9.953.280
OC-768	38.486.016	1.327.104	39.813.120
OC-3072	153.944.064	5.308.176	159.252.240

Table 1.1: Optical carrier specifications.

tively. Speeds beyond 40 Gbit/s are technically viable (*e.g.*, 160 Gbit/s), but have not been widely implemented yet due to the cost of high-rate transceivers [175]. When fiber exhaust is a concern, multiple SONET signals can be transported over multiple wavelengths over a single fiber by means of DWDM. Such circuits are the basis for all modern transatlantic cable systems and other long-haul circuits.

SONET and SDH are similar in their way of dividing wavelengths into time slots. That is done by using TDM to interleave the transmission of SONET and SDH frames, namely Synchronous Transport Signal (STS) and Synchronous Transport Module (STM), respectively. STS and STM frames are transmitted at every 125 μ s. However, the structure of SONET frames differs from SDH ones.

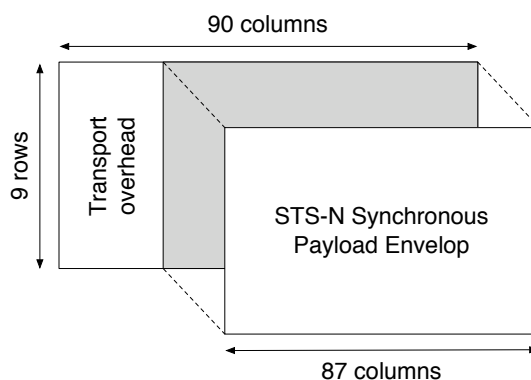


Figure 1.3: The STS-N frame structure.

The basic SONET frame (STS-1) is formed by 9 rows of 90 columns (810 bytes of data). The first 3 columns (27 bytes of overhead) contain the transport overhead for supporting features such as framing, management operations, and error monitoring. The remaining 87 columns (783 bytes of payload) form the synchronous payload envelope, which is the available capacity to transport network user data. The structure of the SONET frame is shown in the Figure 1.3.

The basic SDH frame (STM-1) is rather similar to the STS-1 frame with regard to its format, but it is three times larger, though. The STM-1 frame consists of 9 rows of 270 columns (2.430 bytes of data) that, as well as STS-1 frames, are transmitted at every $125 \mu\text{s}$. The first 9 columns (81 bytes of overhead) contain the transport overhead and the other 261 columns (2.349 bytes of payload) form the payload envelope. The SDH frame is shown in the Figure 1.4.

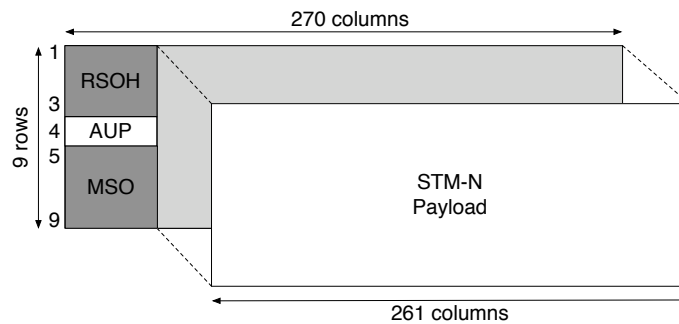


Figure 1.4: *The STM-N frame structure.*

The multiplexing of these SDH and SONET basic frames using TDM allows higher transmission speeds to be reached. For instance, if three STS-1 frames are multiplexed by interleaving each STS-1 frame (810 bytes) this will allow 3 STS-1 frames (2.430 bytes) to be sent every $125 \mu\text{s}$, having therefore a rate of 155.520 Mbps. The same explanation is valid for SDH.

1.1.3 Network management

In order to keep hybrid networks operational, a proper network management is desirable. Network management is a broad term, which can be categorized into 5 management functional areas referred by the acronym FCAPS. FCAPS stands for *Fault, Configuration, Accounting, Performance, and Security* [78].

1. *Fault management:* aims at identifying, isolating, correcting, and logging any

fault that may take place in a network.

2. *Configuration management*: aims at gathering, storing, and keeping track of configuration parameters (e.g., routing table) from network devices.
3. *Accounting management*: aims at gathering statistics (e.g., link usage) from network users to enforce usage quota as well as billing users for resources utilization.
4. *Performance management*: aims at maintaining and optimizing QoS in a network. Through the collection and analysis of network data, the network performance can be monitored and adjusted whenever required.
5. *Security management*: aims at securing a network against user misbehavior and unauthorized access. User authentication and data encryption play an important role here.

The focus of this thesis is regarding the configuration and performance functional areas. Within the performance area, we aim at the monitoring of IP flows transiting within a hybrid network. This information is then analyzed, and the configuration of the hybrid network is adjusted whenever required, which relates to the configuration aspect.

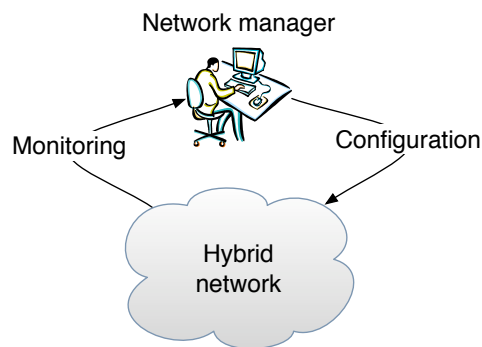


Figure 1.5: The human-in-the-loop paradigm in the management system.

Figure 1.5 depicts a traditional network management paradigm, in which a network manager regularly monitors a hybrid network. Based on his analysis of the collected data, he may decide to change the network configuration in order to adjust the network performance. It is worth highlighting that this paradigm keeps the human in the management loop. That is, most of the management decisions have

to go through the network manager. As a result, the management system does not go beyond any predetermined state or perform any unexpected action, unless explicitly triggered by the network manager. While this is not a problem by itself, it may not scale when the number of decisions to be taken by the network manager goes beyond its capacity. That could jeopardize any other management activity to be performed by the network manager.

1.1.4 Self-management

In order to move the human factor to a higher level in the management system, new research studies have been carried out. In such studies, performance and configuration aspects are performed by a self-managing system rather than by a network manager. The latter expresses *what* he expects the self-managing system to achieve, but not necessarily *how* this is to be obtained. Figure 1.6 shows a self-managing system taking the place of a network manager. The latter is moved to a higher level in the management hierarchy where he keeps the self-managing system in control, rather than the whole hybrid network.

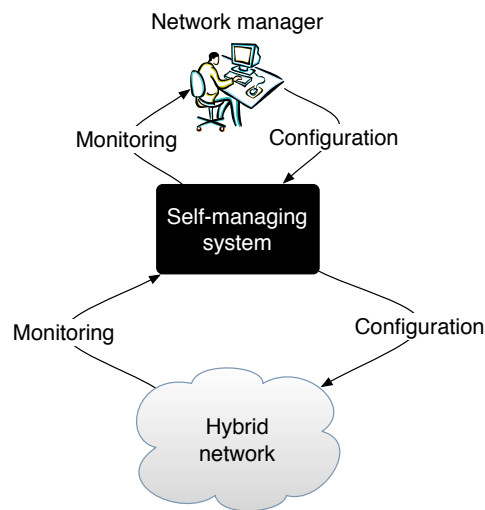


Figure 1.6: The self-managing paradigm.

It is important to say that the self-management concept is not recent. It has been out there for many years and it was started by IBM in 2001 with the release of the Autonomic Computing Initiative (ACI) manifesto [71]. In such manifesto, IBM pro-

posed an approach in which self-managed computing systems could work with a minimum of human interference. This approach is inspired from the human body's autonomic nervous system. Many actions are performed by our nervous system without any conscious recognition, such as the act of adjusting our eye's pupils depending on the amount of light or the act of sweating in order to regulate our body temperature. Below, we quote the main objective of IBM's autonomic initiative that is:

“to design and build computing systems capable of running themselves, adjusting to varying circumstances, and preparing their resources to handle most efficiently the workloads we put upon them. These autonomic systems must anticipate needs and allow users to concentrate on what they want to accomplish rather than figuring how to rig the computing systems to get them there.”

A system can be understood as a collection of computing resources bound together in order to achieve certain objectives. For example, a network router can constitute a system responsible for forwarding network traffic. When combined with other network routers, they can form a larger system, *i.e.*, a Local Area Network (LAN) network. On its turn, a LAN network combined with other LANs can form a Metropolitan Area Network (MAN), and so on. Based on the IBM autonomic principle, each system must be able to manage its own actions (*e.g.*, traffic forwarding), while collaborating with a larger, higher-level system.

The same analogy can be found in the human body. From single cells to organs and organ systems (*e.g.*, the circulatory system), each level maintains a measure of independence while contributing to a higher level of organization, culminating in the organism, *i.e.*, the human body. In most parts of our daily life, we remain unaware of our vital organs (*e.g.*, the heart) activities, since these organs (systems) take care of themselves and they only ascend to a higher level (*e.g.*, the brain) when something is wrong and they need some assistance.

More details on the ACI relation between autonomic computing and the research work of this thesis is addressed in Section 2.3.

1.2 Related work on self-management

SINCE the releasing of the ACI manifesto, several research works investigating the use of self-management capabilities have been reported. To name a few of these works, Lupu *et al.* [93] have been researching the use of self-management on healthcare practicing, in which a ubiquitous self-managed computing environment is used to monitor and report the health of patients under medical treatment. In another work, self-management is investigated to be used in situations where there is a

great risk for human beings, such as in military or disaster scenarios. Within this line of research, we point out the work by Eskindir *et al.* [7] who has been investigating the use of self-management on Unmanned Autonomous Vehicles (UAVs). UAVs are mobile robots employed for reconnaissance in dangerous areas for human beings. For example, in a war zone scenario, instead of sending human soldiers, UAVs could be sent into a certain enemy area in order to gather vital information (*e.g.*, the positioning of the enemy troops). These UAVs would self-coordinate their actions, based on a mission assigned by a higher level entity, such as an army commander.

Not much differently, self-management has also been investigated in the area of communication networks [135] [46] [81]. Much of the focus of this investigation aims at developing highly distributed algorithms, with the objective to optimize several aspects of network operability (*e.g.*, performance). This optimization is aimed through the provision of self-management capabilities to communication networks.

Within the specific context of hybrid networks, self-management has been investigated in various ways. Most research works can be found on the use of self-management in hybrid wireless and wired networks [140] [25] [50] [155]. The network management for these hybrid networks is different from conventional and infrastructure-based network management. Device heterogeneity, constant mobility, and dynamic topologies make the challenge quite hard. As a result, a number of problems arise from this new hybrid network architecture. In particular, a large number of access points or base stations in the hybrid network may not be efficiently managed and configured through a centralized management system. In such a situation, the employment of self-management principles can satisfy the autonomous behavior of these hybrid networks as well as improve the dynamic behavior of nodes within such networks.

Studies that are closely related to the research presented in this thesis, are by Sabella *et al.* [129] and Miyazawa *et al.* [103]. Sabella *et al.* focus his research in new strategies for performing dynamic routing and grooming (multiplexing) of IP flows over lightpaths in hybrid networks based on Generalized Multiprotocol Label Switching (GMPLS). Such networks are modeled as a multi-layer network consisting of an IP/MPLS layer and an optical layer. Sabella *et al.* propose a solution that adopts a dynamic routing algorithm based on the Dijkstra algorithm integrated with a method for grooming IP flows over lightpaths. It is worth highlighting that the decision making process is not the main focus of Sabella *et al.*'s research, but instead, the question how to decrease the ratio of blocked lightpath requests. More details about the grooming strategies introduced in Sabella *et al.*'s research and their relation with our research work is presented in Section 4.6.

Aligned with Sabella *et al.*'s research, Miyazawa *et al.* [103] also proposed a

multi-layer network consisting of an IP layer and a MPLS/GMPLS layer. In their research, they propose a dynamic bandwidth control management mechanism based on the volume of IP flows. In their work there is a centralized management system that observes the bandwidth of IP flows, and decides about offloading these flows based on pre-defined upper and lower threshold values. These threshold values are defined in advance by a human operator and statically stored in the configuration file of the management system. Once an IP flow has a bandwidth utilization that exceeds the pre-determined upper threshold, the management system triggers an action to create a lightpath. In contrast, when the flow decreases its bandwidth utilization below the lower threshold, the management system initiates a deletion process for deleting the established lightpath. The main shortcoming of their research is that the thresholds are statically defined and they are not adjusted depending on the current traffic. This can lead up to an unbalance between the IP and optical levels. If the upper threshold values are too restrictive, IP flows may not be offloaded over lightpaths, which may result in congestion in the IP level and underutilization of the optical level. Moreover, with a misadjusted lower threshold, a flow can be inadequately removed from the optical level back to the IP level, where it can contribute to a congestion situation.

In our research, we aim at moving IP flows to the optical level based on the flow throughput, but without any restriction imposed by the optical level (threshold values). Large flows are moved whenever there is bandwidth available at the optical level. When there is no free bandwidth, flows which are utilizing bandwidth the least at the optical level, are removed first in order to give place to larger flows. More details about our self-management approach will be given in Chapter 4.

1.3 Motivation, scope, and objective

THE **motivation** of this thesis comes from the need to provide self-management capabilities for hybrid optical and packet switching networks, as discussed in the previous subsections. As it is going to be presented in more details in Chapter 2, network management approaches currently used in these networks require human interaction to select IP flows and manage lightpaths. This interaction may be therefore slow and error-prone.

Within a Optical Circuit Switching (OCS) paradigm, lightpaths between source and destination pairs are established before data is transferred, and released after the transfer is completed. When a lightpath is requested within one single domain (intra-domain), several steps are taken (*e.g.*, phone calls and emails exchanges) between requesters and network domain administrators in order to establish the light-

path. Hence, it may take hours before a desired lightpath can be used. When requests for a connection span multiple domains (inter-domain), the lightpath provisioning may take even longer.

While the lightpath is being established, many large IP flows may be using resources at the IP level and, therefore, likely congesting the IP level. Moreover, by the time the lightpath is finally established, those large flows may no longer exist. In addition to this slowness, large IP flows eligible for lightpaths might somehow be transiting undetected by the manager's eyes. As a result, these flows would stay at the IP level, whereas they could be transmitted over lightpaths where they would perceive better QoS.

Some alternatives to speed up the establishment process of a lightpath have been proposed as it is the case of the Optical Packet Switching (OPS) and Optical Burst Switching (OBS) paradigms. Within a OPS paradigm [18], a lightpath is established by means of sending a control packet along with the data to be transferred over the chosen lightpath. If successfully performed, the lightpath setup can take the order of micro- or nanoseconds. However, the OPS paradigm requires data to be buffered while the control packet is processed at each intermediate node along the chosen lightpath. This can result in packet loss if there is no available space in buffer. Alternatively, the OBS paradigm [119] tries to overcome the need for buffering by sending the control packet and the data to be transferred more loosely coupled in time. That is done by choosing a fixed delay (offset time) that is no shorter than the maximal time need to process a control packet along the intermediate nodes. Due to this deliberate delay, the setup of a lightpath when using the OBS may take longer than the OPS in the order of milliseconds. It is worth mentioning that both paradigms (OPS and OBS) are in their experimental stage and they may still take some time to be fully deployed on hybrid optical and packet switching networks.

One can say that there is a huge gap between the OCS paradigm, and the OPS and OBS paradigms with respect to the time to set a lightpath up. Within this context, we put this thesis into perspective as depicted in Figure 1.7. We see our self-management of lightpaths in hybrid packet and optical switching networks as a proposal that fits in between this gap as well as that may be implemented in a near future optical Internet.

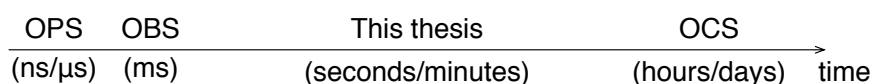


Figure 1.7: *This thesis in perspective.*

With this in mind, the **scope** of this thesis is the *monitoring* of IP flows in hybrid networks, the *decision making* with respect to moving IP flows over lightpaths, and the *configuration process* these decisions require. Figure 1.8 illustrates the scope of this thesis.

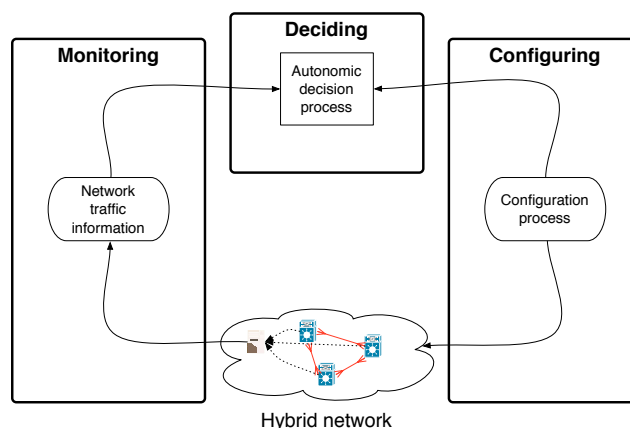


Figure 1.8: The scope of this thesis.

Within this scope, the main **objective** of the research presented in this thesis is:

to investigate the feasibility of employing self-management capabilities on hybrid optical and packet switching networks in order to autonomically move large IP flows from the IP level to the optical level, as well as, creating and releasing lightpaths to transport such flows at the optical level.

1.4 Research questions, and their research approaches

GIVEN the fact we foresee optical communication infrastructures playing a major role in the *Future Internet*, the main high level research question we pose in this thesis is as follows: “*Is the idea of self-management of hybrid optical and packet switching networks technically feasible for the future Internet?*” In order to answer this question, we refine it into the following subquestions:

1. *What is the state-of-the-art in the management of hybrid networks?*
2. *How can the monitoring of IP flows be performed?*

3. *How can autonomic decisions be made?*
4. *What are the side effects of moving IP flows to the optical level on the fly?*

Research question 1 has as main objective to investigate the possibility of employing self-management in future hybrid optical and packet switching networks. For that, we analyze the current management approaches used to establish and release lightpaths in such hybrid networks. Answering this research question is important to show the main drawbacks present in these approaches and therefore motivate the feasibility of our self-management proposal. In order to do that, we perform a study of the literature and interview professionals in the network management area.

The remainder research questions 2, 3, and 4 can be summarized into the question *How can self-management be implemented?* These questions are related to different stages of our proposed implementation. Research question 2 aims at analyzing what network information is relevant to take autonomic decisions about moving flows to the optical level. Since these decisions will be taken on the fly, an evaluation of what network parameters are relevant for that is important. For that, we study the literature and statistically evaluate these parameters. Research question 2 aims also at comparing monitoring techniques that provide means to obtain the chosen network parameters. These techniques are compared while observing their suitability to our autonomic decision process.

Following that, research question 3 focuses on how to make autonomic decisions. For that, our steps comprise checking the literature in order to see whether there is any related study. Since this is not the case, as described later in this thesis, we then introduce our approach to take decisions about moving IP flows over lightpaths. Following that, a validation is performed in order to compare our approach with today's approach to establish lightpaths as well as with the best theoretical approach. We perform this validation through the use of simulation with real network data. Lastly, we observe different strategies to accommodate elected flows over lightpaths.

Last but not least, research question 4 aims at showing some analysis about moving IP flows on the fly. When flows are moved on the fly, some performance problems with the flow throughput may occur. In a manual process, a lightpath is first established and flows are moved over it afterwards. It is known that packet loss may occur in this case when the lightpath capacity is smaller than the flow throughput. On the other hand, little is known about what happens when flows are moved on the fly through network levels. It is expected that some packets belonging to a flow would be transferred more quickly over a lightpath than the other remaining packets still being transferred at the IP level. This could cause packets belonging

to the same flow to arrive out of order at their destination (thus confusing TCP) or even being discarded. In order to observe that, we will make use of simulation tools (NS2 [112]). In addition, we also give thoughts to some additional aspects (lightpath capacity estimation, rerouting, amongst others) to be reconsidered with the advent of our self-management proposal.

Once we have all the answers for these questions, we have all the elements required to achieve our main research objective, as stated on page 13.

1.5 Thesis structure

THE remainder part of this thesis basically follows the order of the research questions posed in the previous section. The remainder of this thesis is thus organized as follows:

- **Chapter 2** (“Management approaches for hybrid networks”) presents first the current management approaches used in hybrid optical and packet switching networks. Second, it presents the main shortcomings these approaches have. Lastly, our self-management approach is introduced. Chapter 2 addresses our research question (1).
- **Chapter 3** (“Monitoring of network traffic”) presents first an evaluation of what network parameters are relevant to our autonomic decision process when deciding about the move of IP flows between IP and optical levels. Following that, monitoring techniques are compared while observing their suitability for our self-management approach. Chapter 3 addresses our research question (2).
- **Chapter 4** (“Making autonomic decisions”) introduces our autonomic decision process. Descriptions and assumptions about the decision process are provided, followed by a validation of our proposal. We also observe here different strategies to accommodate flows over lightpaths. Chapter 4 addresses our research question (3).
- **Chapter 5** (“The impact of self-managing lightpaths”) evaluates the impact on throughput performance when moving flows from the IP level to the optical level on the fly. Lastly, we also highlight some additional aspects that should be reconsidered with the advent of our self-management proposal on hybrid networks. Chapter 5 addresses our research question (4).
- Finally, we close this thesis in **Chapter 6** (“Conclusions”), where we draw our conclusions and identify possible directions for further work.

Chapter 2

Management approaches for hybrid networks

Computer networks are complex communication systems that enable interactions among users and services. Such interactions result in network traffic that should be monitored and managed to guarantee network operability. Currently, a number of network operators, such as GÉANT [63], Internet2 [75], and SURFnet [146], is moving towards hybrid optical and packet switching networks [125] [116]. In this chapter we first review conventional management approaches used in hybrid networks. Second, the main drawbacks of these approaches are exposed, which results finally in the introduction of our self-management approach. The organization of this chapter is as follows:

- *Section 2.1 presents two conventional management approaches: direct and indirect management. In this same section, we present the main technologies used in these approaches.*
- *Section 2.2 shows the main drawbacks the conventional approaches have when employed on current hybrid networks.*
- *Section 2.3 presents our understanding on what the term self-management means. Even though self-management is widely employed nowadays, little consensus exists on its real meaning.*
- *Section 2.4 presents our self-management approach to overcome the drawbacks of the conventional management approaches. We also present the main architectural components behind our self-management proposal.*
- *Section 2.5 closes this chapter by drawing some concluding remarks.*

2.1 Conventional management approaches

NETWORK vendors enable the remote management of their devices (e.g., routers) by means of management interfaces, allowing network operators to configure and monitor those devices. The Simple Network Management Protocol (SNMP)

[137] and the Transaction Language 1 (TL1) [95] are examples of well-known management technologies that have been widely investigated by the network management community. Many others management technologies, however, are available today [138].

Management interfaces provide access to manage network devices, which are conventionally managed by means of two main approaches: direct management and indirect management. With a direct approach, management messages are explicitly issued by the network manager to each managed agent (Figure 2.1). Whereas with an indirect approach, messages are issued by the manager to one managed device that is in charge of signaling the other ones. The last signaled device then notifies the manager the status of the management operation (Figure 2.2).

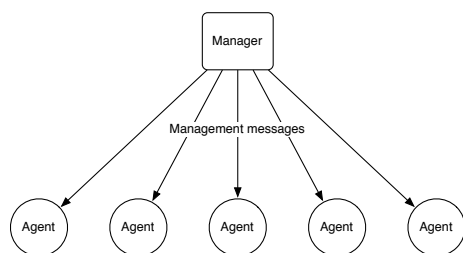


Figure 2.1: Direct management.

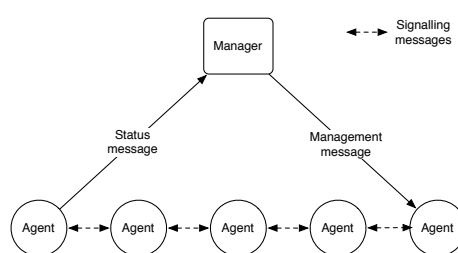


Figure 2.2: Indirect management.

Within the context of hybrid networks, direct management consists of a central manager (*e.g.*, a network operator or an automated management process) directly accessing optical devices to create and release lightpaths for selected flows at the IP level. In contrast, indirect management enables optical devices to coordinate among themselves the creation of lightpaths by exchanging signaling messages. However, the decisions on which IP flows should be moved to the optical level and which devices are involved are still taken by network operators [16].

Human factors also impact on the management of lightpaths. For example, network operators of SURFnet report, when informally interviewed, that it may take hours (intra-domain) or even days (inter-domain) before a lightpath is established by network operators when using a direct management approach. In such long periods, several big IP flows could have been transported via lightpaths, but due to the decision delay they remain being routed at the IP level.

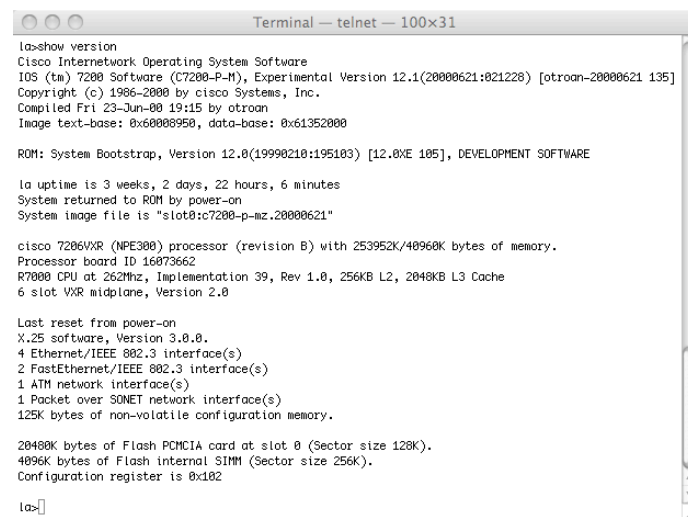
More details about the management technologies used in the direct and indirect approaches are presented in the upcoming Subsections 2.1.1 and 2.1.2.

2.1.1 Direct management approach

In the direct approach, the management of network devices (*e.g.*, a hybrid switch) is performed through management interfaces and protocols like Command Line Interface (CLI), TL1 [95], SNMP [137], and Web-Based Enterprise Management (WBEM) [138]. In the next subsections we will briefly discuss these management technologies. It is important to note that further management technologies exist, such as Common Management Information Protocol (CMIP) [164], NETCONF [48], amongst others, but we concentrate our description on CLI, TL1, SNMP, and WBEM because of their employment on hybrid networks.

Command Line Interface

Command Line Interface is a mechanism that enables users to interact with a device's operating system. The interaction consists of a user issuing commands for which he receives a response back from the managed system, to then enter another command, and so forth, characterizing CLI as a task-oriented configuration solution. CLI is commonly the primary user interface used for configuring, monitoring, and maintaining most network devices, which are usually accessed using protocols, such as TELNET or SSH. One example of CLI is Cisco IOS CLI [29]. Figure 2.3 shows the output of a `show version` command, which can be issued by using Cisco IOS CLI.

The image shows a terminal window titled "Terminal — telnet — 100x31". The output of the "show version" command is as follows:

```
la>show version
Cisco Internetwork Operating System Software
IOS (tm) 7200 Software (C7200-P-M), Experimental Version 12.1(20000621:021228) [otroan-20000621 135]
Copyright (c) 1986-2000 by cisco Systems, Inc.
Compiled Fri 23-Jun-00 19:15 by otroan
Image text-base: 0x60000950, data-base: 0x61352000

ROM: System Bootstrap, Version 12.0(19990210:195103) [12.0XE 105], DEVELOPMENT SOFTWARE

la uptime is 3 weeks, 2 days, 22 hours, 6 minutes
System returned to ROM by power-on
System image file is "slot0:c7200-p-mz.20000621"

cisco 7206VXR (NPE300) processor (revision B) with 253952K/40960K bytes of memory.
Processor board ID 16073662
R7000 CPU at 262Mhz, Implementation 39, Rev 1.0, 256KB L2, 2048KB L3 Cache
6 slot VXR midplane, Version 2.0

Last reset from power-on
X.25 software, Version 3.0.0.
4 Ethernet/IEEE 802.3 interface(s)
2 FastEthernet/IEEE 802.3 interface(s)
1 ATM network interface(s)
1 Packet over SONET network interface(s)
125K bytes of non-volatile configuration memory.

20480K bytes of Flash PCMCIA card at slot 0 (Sector size 128K).
4096K bytes of Flash internal SIMM (Sector size 256K).
Configuration register is 0x102

la>
```

Figure 2.3: Output of a `show version` command on a Cisco router.

Transaction Language 1

Transaction Language 1 is a management protocol originally conceived for telecommunication environments used to manage optical and broadband access infrastructures. TL1 is a human readable language developed by Telcordia Technologies that allows management stations to communicate with devices of different vendors, removing the need to support specific vendor interfaces. The management stations usually exchange TL1 messages with network elements through TCP connections. The messages supported by TL1 are the following:

1. *Input message*: is a command sent from a management station to a network device in order for the latter to perform some requested action.
2. *Acknowledgment message*: is a brief output message generated in response to an input message. An acknowledgment is generally later followed by an output response message to the originally issued command. Acknowledgments are also issued if the normal response (output response message) to an input message cannot be transmitted within 2 seconds of its receipt.
3. *Output message*: is the response to an input message.
4. *Autonomous message*: is an asynchronous message sent by the network device to the management station. The message is normally triggered by events or alarmed conditions on the network device.

It is worth mentioning that the syntax of the TL1 messages follows a fixed structure while the commands themselves are extensible.

Simple Network Management Protocol

The Simple Network Management Protocol is defined by the Internet Engineering Task Force (IETF) [74] as an application layer protocol used for monitoring and configuration of network devices. However, in practice, SNMP is mostly used for monitoring and hardly for configuration [138].

A SNMP-based managed scenario consists of three main components: 1) managed device, 2) agent, and 3) Network Management System (NMS). A managed device is any network device (*e.g.*, switches and routers) that contains a SNMP agent, which collects and stores management information in a Management Information Base (MIB) [126]. MIB is a set of management information defined in modules that are written following the Structure of Management Information (SMI) [100]. The management information is made available by managed devices via SNMP so that the NMS can deal with them. Finally, a NMS runs applications that monitor and

control managed devices. Figure 2.4 shows the main SNMP components and their relationships.

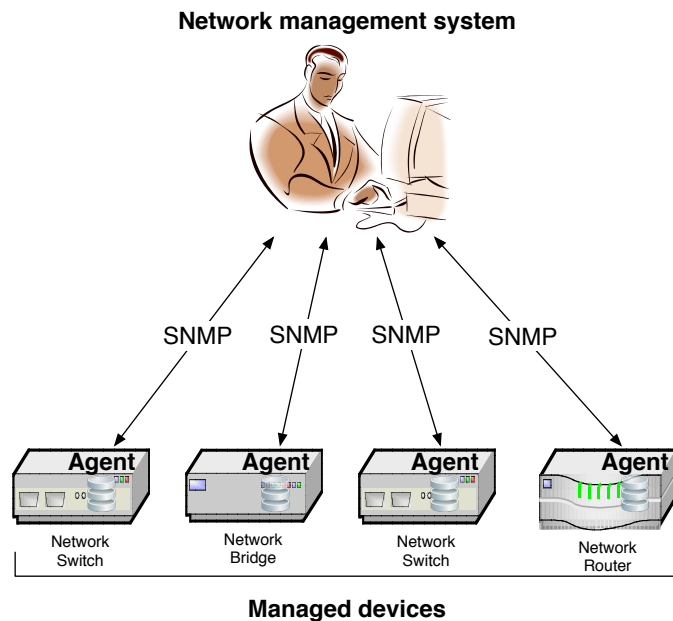


Figure 2.4: *SNMP main components and their relation.*

In comparison with TL1 (in which the messages are exchanged in plain ASCII text), SNMP messages are encoded following the Basic Encode Rules (BER) [77] into Abstract Syntax Notation One (ASN.1) [76]. As a result of that, SNMP reply messages have to be decoded first in the network management system and then interpreted by the network operator.

Web-Based Enterprise Management

Web-Based Enterprise Management is a set of management and Internet standard technologies developed by the Distributed Management Task Force (DMTF) [43] in order to unify the management of enterprise computing environments. WBEM is based on the following open standards:

1. *Common Information Model (CIM)* standards [42]: provides a common format, language, and methodology for collecting and describing management data.

2. *CIM-XML* [41]: uses Extensible Markup Language (XML) over HTTP to exchange CIM information. It is worth mentioning that having XML as a data representation method has the advantage of being human readable.
3. *CIM Operations over HTTP* [44]: defines a mapping of CIM operations onto HTTP that allows implementations of CIM to interoperate in an open and standardized manner.
4. *Web Services for Management* [45]: is a SOAP-based protocol that provides a common way for systems to access and exchange management information.

WBEM has been adopted by several corporations such as Apple, Microsoft, and Hewlett-Packard, amongst others. Corporations such as these have incorporated WBEM into their operating systems mostly to support remote management.

Example of use of the direct management approach

By using one or more of the aforementioned technologies, network managers can directly manage devices on hybrid networks. In order for network managers to create lightpaths, they directly configure each device along the path by passing to them the required connection parameters.

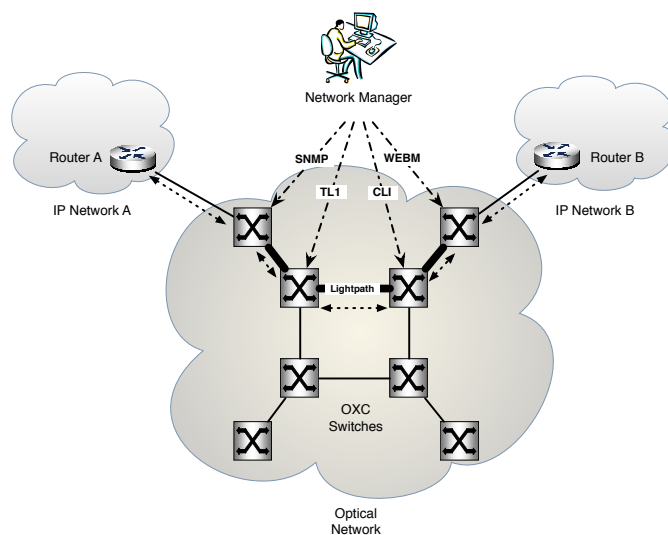


Figure 2.5: Example of a lightpath establishment through the direct management approach.

Figure 2.5 shows a scenario in which the direct management approach is used to configure Optical Cross-Connected (OXC) switches [109]. Following the definition of direct management, the network manager has to set up each OXC switch along the chosen route. The manager may use a single management technology (*e.g.*, SNMP) or a combination of them, since optical switches from different vendors may employ different management technologies. Once the setup is finished, the traffic is transferred over the established lightpath.

2.1.2 Indirect management approach

In the indirect approach, the management of network devices is performed through the exchange of signalling messages. As shown in Figure 2.2, in the indirect approach, network devices forward messages from device to device in order to perform certain task (*e.g.*, the setup of a connection). This is different from the direct approach in which a single manager individually contacts each device for the same task. Well-known technologies that employ the use of signalling are Multiprotocol Label Switching (MPLS) [127], used in packet switching network, and its extension GMPLS [96], used in hybrid networks.

Multiprotocol Label Switching

Multiprotocol Label Switching is a technology that allows a packet-switched network (IP network) to operate as a circuit-switched network. With regard to its architecture, MPLS converges connection-oriented forwarding techniques and the Internet's routing protocols to one single architecture [6]. With regard to the OSI model, the MPLS architecture is mostly considered to be situated between the Layer 2 (data link layer) and Layer 3 (network layer).

As depicted in Figure 2.6, MPLS works by adding labels (MPLS headers) to conventional IP network packets. These labels are assigned to IP packets when they enter an MPLS network through Label Edge Routers (LERs). Once inside the MPLS network, the packet's IP headers are not analyzed anymore by MPLS routers located in the core of the MPLS network, called Label Switch Routers (LSRs). Rather, labels are used as an index into a table that specifies the next hop and a new label. The old label is replaced with a new one, and then the packet is forwarded to its next hop, enabling the creation of Label Switch Paths (LSPs) along the MPLS network. Label switching is faster than conventional IP routing because the label lookup requires only one access to the table, in contrast to a traditional routing table access that might require thousands of lookups [17].

One important feature of MPLS is in connection with traffic engineering. MPLS routers can create LSPs taking into account network traffic load and available band-

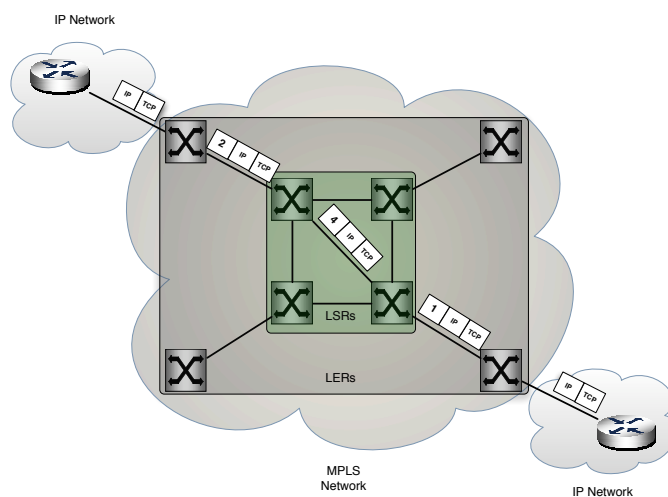


Figure 2.6: MPLS forwarding scheme.

width. This gives to network operators the ability to control traffic loads in different parts of a MPLS network, to optimize resource usage, and to route traffic along certain paths. There are two protocols for managing LSPs: Constraint-Routing Label Distribution Protocol (CR-LDP) [4] and Resource Reservation Protocol for Traffic Engineering (RSVP-TE) [51]. It is worth highlighting although that the former has been deprecated. The IETF MPLS working group [72] has decided to focus their efforts purely on the latter.

The MPLS architecture has an important drawback. It cannot be applied in hybrid optical and packet switching networks since it was originally defined to be applied in packet-switching networks and does not convey sufficient information for hybrid networks. Therefore, some modifications and the addition of new features are required to adapt MPLS to the peculiarities of the today's hybrid networks.

Generalized Multiprotocol Label Switching

GMPLS extends the characteristics of MPLS by supporting three different types of switching, besides the traditional Packet Switching Capable (PSC) type of switching:

- *Fiber-Switch Capable (FSC)* consists in executing the transmission of data according to the position of the actual physical port of the optical fiber through which data is transmitted.

- *Lambda Switch Capable (LSC)* consists in executing the transmission of data according to the lambda (wavelength) inside the optical fiber through which data is transmitted.
- *TDM*: consists in executing the transmission of data according to the time slots inside a wavelength of the optical fiber through which data is transmitted.

GMPLS defines a hierarchy of LSPs. At the bottom of the hierarchy are LSPs established by using the PSC type of switching. Followed in ascending order are LSPs established by using TDM, LSC, and FSC. This hierarchy is similar to MPLS support for label stacking, in which many smaller LSPs can be aggregated into one larger LSP. Unlike MPLS, GMPLS no longer carries labels in the data, but they are defined in the GMPLS-enabled optical switches. Conversely, regarding the configuration process of LSPs, GMPLS works similarly to MPLS by using signaling messages.

With regard to the way GMPLS can be employed in hybrid networks, GMPLS can support two operational models [10]: peer model (Figure 2.7) and overlay model (Figure 2.8). These operational modes influence the way users (*e.g.*, a network operator in an adjacent IP network) of a GMPLS-enabled hybrid network request the establishment of LSPs.

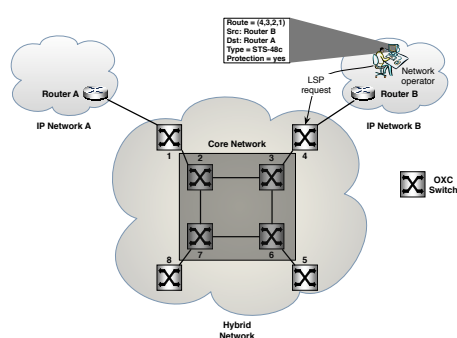


Figure 2.7: Peer model.

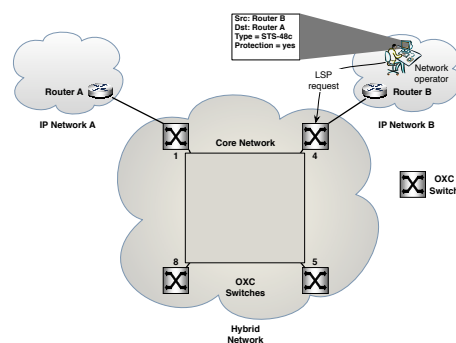


Figure 2.8: Overlay model.

Peer model: In the peer model, the complete topology of the hybrid network is known to all network devices, *i.e.*, all devices in the hybrid network share the same network topology information. The peer model is suitable whenever the transfer of full routing is required. Moreover, if there is no concern regarding policy and security at the network interconnection boundaries, users at different administrative domains are able to see the entire hybrid network topology

as well as to choose a desired LSP. Figure 2.7 shows one example in which a network operator within network B sends connection parameters to its adjacent OXC switch in order to create a LSP. Once the adjacent optical switch retrieves this information, it starts the process of establishing the desired LSP by interacting with other switches along the path. In case it is not possible to establish the LSP, an error message is sent back to the user.

Overlay model: Unlike the peer model, in the overlay model the hybrid network topology is not exposed to the edge devices or to any users in different administrative domains. The edge devices are although revealed to users at different administrative domains. This model also adopts separate routing domains. The overlay model is generally employed where specific policies are defined as a means to allow a specific domain not to disclose its topology. Since the topology of the core network is hidden, users are not able to choose their desired connection path. Therefore, to create a LSP, users just send a request for a LSP towards the destination; the OXC on the edge of the hybrid network then determines the best path (by using routing protocols, such as Open Shortest Path First (OSPF) [110] or Intermediate System-to-Intermediate System (IS-IS) [113]) to the far OXC where the destination router is connected. Figure 2.8 shows one example in which a network operator within the network B provides most of the connection parameters shown in the Figure 2.7, except for the desired route.

2.2 Analysis of the conventional approaches

THE main advantage of direct management is its simplicity. The whole management process is centralized, which allows a better control of the managed network. As results, the configuration of switches can be performed in parallel and troubleshooting can be more precise and fast. However, the direct management approach has an important drawback concerning scalability, which is a classical problem in any centralized solution [102]: when the number of managed devices increases above the number the management system is able to cope with, the management activities performed by the management system begin to deteriorate. In such overloaded situation, the direct management may be too slow to react to changes in the network traffic, and thus reduce network performance.

The indirect management approach, in comparison with the direct approach, is more scalable. This comes from the fact that fewer network devices need to be directly managed while the remaining ones are indirectly managed through signaling. This results in a certain autonomy to the managed network, since some network de-

vices can have themselves the freedom of taking decisions (*e.g.*, choosing a LSP path across the managed network). However, this autonomy must still be explicitly triggered by the users of the network. In addition to that, these users and network managers must still provide configuration parameters for the network. Compared to the direct management approach, these configuration parameters are only provided to some devices and these devices signal the others.

The main drawback of the indirect approach, however, is that the configuration of the network devices is sequential and not parallel as with the direct approach. This sequentiality may be slow if there are several devices to be configured in order to set up a LSP path. Moreover, this configuration process can be even slower when the LSP is computed on the fly by the use of signaling messages. For instance, if there is one device along the chosen LSP path that cannot attend a certain requirement, the signaling protocols have to rollback all the previous configured devices and decide for another LSP path. In such a situation, the configuration process can be significantly slower depending on the amount of managed devices.

2.3 The self-management manifesto

CONVENTIONAL management approaches have certain shortcomings, such as lengthy configuration process and heavy dependence on human intervention to perform certain tasks (see Section 2.2). In the specific case of hybrid networks, these approaches depend on the intervention of network managers to select and move IP flows to the optical level and establish/release lightpaths. This intervention can therefore take a considerable amount of time to be performed.

In order to overcome this dependency on human intervention, a new management approach, named self-management, has been widely researched in the network management community [46] [81] [117] [26] [40]. The term self-management means the act of computer systems managing their own operation without (or with very little) human intervention, as defined by IBM in 2001 within the IBM ACI [71]. IBM divided self-management into 4 aspects (nonetheless other subdivisions exist [132]), commonly referred as *self-**, as follows:

Self-configuration: consists of an automated configuration process of components and systems based on high-levels policies. For example, when a new device is incorporated into a computer network, this device is expected to automatically configure itself and at the same time the rest of the network seamlessly adjust itself to take this new device in.

Self-optimization: means that components and systems are supposed to continuously improve their own performance. One example of this aspect is the auto-

matic update process most operating systems provide to their users. Instead of requiring the computer users to manually seek for updates, the operating system does that automatically, aiming at keeping the operating system in an optimal shape.

Self-healing: consists of the capability of a system to automatically detect, diagnose, and repair problems found at certain components. As an example, a computer could self-heal every time a virus would strike the system, by automatically patching the damaged files.

Self-protection: is seen as a system automatically defending itself against malicious attacks or failures. Systems are also supposed to early detect an incoming attack or failure. A computer system could, for instance, prevent the infection by a certain email virus through analysis of email attachments.

Despite the fact that the self-management manifesto was initially proposed by scientists and industry experts in the IBM Research headquarters, several other institutions are expanding its idea, such as Cornell University [157] and Columbia University [114]. It is worth of mentioning that not only American institutions are focusing on using self-management, but also European projects such as Autonomic Internet (AUTOI) [12] and Exposing the Features in IP version Six protocols that can be exploited/extended for the purposes of designing/building Autonomic Networks and Services (EFIPSANS) [92] projects.

2.3.1 Definitions of self-management

Although the term self-management has been widely considered in the community, there is no universal consensus on what self-management actually means [156] [134], which leads to different definitions for the term self-management. Some of the most known definitions for self-management are as follows:

- *Autonomic management:* is the most common synonym used to refer to the term self-management. That comes from the fact IBM considers self-management as the essence for autonomic computing systems [86]. As a result, the terms self-management and autonomic management are interchangeably used to mean the same. By analyzing the keywords attached to papers submitted via the Journal and Event Management System (JEMS) [133], we checked the amount of papers that were submitted to the most important network management conferences (*e.g.*, IM, NOMS, MANWEEK). The result is that 80% of the papers are submitted with the keywords as self-*, whereas 20% are registered as autonomic. This leads to a conclusion that even if they are constantly

used as synonyms, the term self-management is the most referred and used by the network management community.

- *Automatic management*: is commonly confused with autonomic (and thus with self-management). Even though their meaning are similar, there is however a subtle difference between them. According to the New Oxford American Dictionary [101], automatic means the act of “*working by itself with little or no direct human control*”. Whereas, autonomic means “*acting involuntarily or unconsciously*”. Within the network management context, automatic management could refer as the act of managed devices automatically following explicit policies defined by a network operator. On its turn, autonomic management could refer as a specialized automatic process in the sense that the process is instructed to perform actions based on certain policies too, **but** with the capability of self-learning new actions.
- *Autonomous management* [28]: is another definition used sometimes to refer to self-management. Autonomous means that a process can operate independently from any human intervention. This would require an autonomous system to be highly intelligent to cope with management tasks. Moreover, an autonomous management system does not necessarily need any management interface since it runs without outside control. However, this lack of external control (*e.g.*, a network operator) results, according to some, in a contradiction [118]. If an autonomous “management” system includes enough intelligence in order for the system to govern its own behavior (*i.e.*, its own management), one can assume that there is no need whatsoever of managing such a system, which somehow invalidates the use of the term management to address this kind of management approach.

It is worth saying that the foregoing differentiation among the self-management definitions is not a common view in the community. On the contrary, this differentiation solely destines for being a reference to be used throughout this thesis. Moreover, we see these definitions as following an evolution in the network management approaches as well as having different degrees of autonomy (Figure 2.9).

The simplest management approach is the conventional management approach, as presented in Section 2.1. In the conventional management approach, the network management system is manually managed by network operators. There is no intelligence whatsoever and no (or very little) automation in the execution of management tasks. A next step in the evolution of management approaches is the automation of management tasks. In this case, the management system automatically performs explicit tasks defined by network managers, but nothing beyond the scope of the

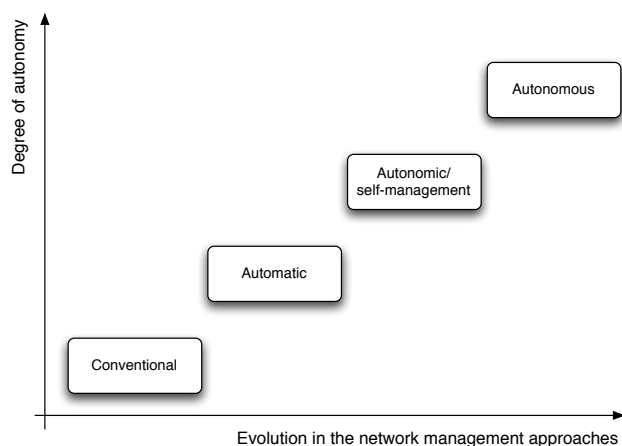


Figure 2.9: Evolution in the network management approaches vs. their degree of autonomy.

defined rules. Following to automatic management, autonomic management (or self-management) also performs these tasks, but it is capable of learning new rules by itself. The last step in the evolution process and the most complex one is the autonomous management. At this level, the management system is fully capable of deciding by itself the rules to follow. There is therefore no dependence on human intervention. The management system is intelligent enough to decide its own rules and following them according to its judgement.

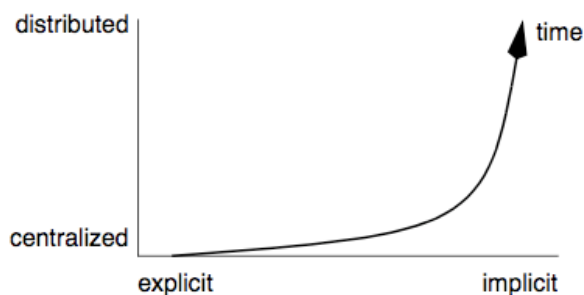


Figure 2.10: "From explicit & centralized to implicit & distributed management" [118].

A similar vision (Figure 2.10) in the way network management is performed is presented in [118]. According to [118], a centralized management approach is ini-

tially chosen as a way for network operators to ensure that the management system will behave as expected. For that, a closer look at the behavior of management system is needed, which brings a centralized approach as a natural initial choice. This initial choice can be related to the conventional management approach described in Section 2.1, in which network managers are responsible for the management tasks. However, as the management tasks are becoming clearer, some steps are automated step by step, which results in little need for human intervention. To avoid problems related to centralization (single point of failure, possible performance bottleneck), the next step is to distribute these automated management tasks more and more. There is therefore a shift from centralized and explicit management towards distributed and implicit management approaches, such as the aforementioned automatic, autonomic, and autonomous approaches.

2.4 Self-management of lightpaths

WITHIN the context of this thesis, self-management aims at autonomically: 1) detecting flows at the IP level eligible to the optical level as well as 2) establishing/releasing lightpaths for those flows. Network operators would only be required to initially configure the self-management process with decision policies. After this initial setup, the self-management process autonomically runs by itself. It is important to highlight that our self-management approach is more focused on the self-configuration aspect of the IBM *self-**, although self-optimization could also be pointed out as another considered aspect due to the load balancing between the IP and optical levels.

At its current stage, our self-management approach is designed to work in a centralized way rather than in a distributed way. The reason for that comes from the fact that we want to be sure that our approach behaves the way we expect. This close observation is presumed to provide insightful information for a move towards to a distributed self-management approach at a later stage. We followed, therefore, the trend in the development of new management systems as described in [118].

Decision policies defined by network operators should fulfill a desired objective, which is expected to be achieved by using the self-management approach. The main objective of our self-management approach is to offload as much traffic as possible from the IP level to the optical level. For that, our self-management approach aims at selecting to the optical level flows that are few in amount, but represent most of the traffic, namely the elephant flows. Figure 2.11 depicts how our proposal for a self-management of lightpaths in hybrid networks looks like.

In Figure 2.11, IP routers located in the IP domain B are exporting network

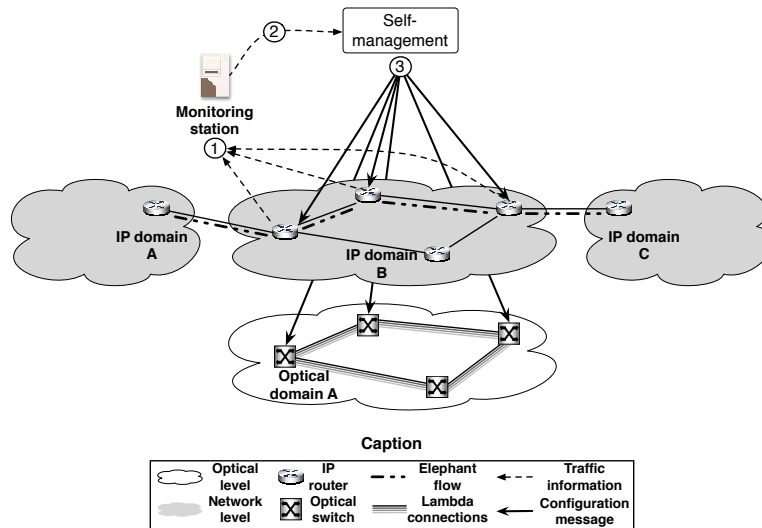


Figure 2.11: Self-management of lightpaths in hybrid optical and packet networks.

traffic information to a monitoring station (step 1). Network information contains flow information, such as source & destination IP addresses, protocol, flow volume, amongst others. This information is then forwarded to our self-management module (step 2). Based on the information received, decisions are made by the module taking into account whether an elephant flow is eligible or no longer eligible for a lightpath at the optical level. If the decision is in favor of creating a lightpath (*i.e.*, the elephant flow is eligible to be moved to the optical level), the self-management module configures the IP routers in the IP domain B and the optical switches in the optical domain A (step 3). The routers are informed that the elephant flow is offloaded to the optical level. On their turn, the optical switches are configured to establish a lightpath for the offloaded elephant flow. From that point on, the elephant flow is switched at the optical level bypassing thus the network level in the IP domain B.

Our self-management of lightpaths can be divided into two architectural groups: functional and physical architectures [56]. The functional architecture describes the functional blocks of our self-management approach as well as their respective functionality. On its turn, the physical architecture presents where the functional blocks are physically located.

2.4.1 Functional architecture

Our functional architecture presents the functional blocks of the self-management module as well as their interactions. Our architecture deals with the network interface and Internet layers of the TCP/IP network architecture (Figure 2.12).

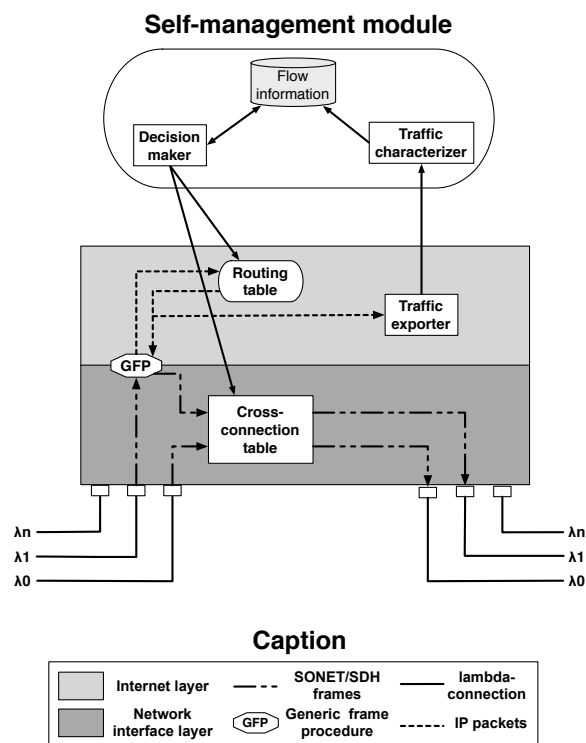


Figure 2.12: Self-management functional elements.

The self-management functional block is composed of three main elements: traffic characterizer, decision maker, and flow information cache. The traffic characterizer is in charge of collecting network information exported by a traffic exporter and characterizing it. The characterized information is then stored in the flow information cache. On its turn, the decision maker analyzes this information and, based on the decision policies defined by network operators, decides what IP flows should be moved from the IP level to the optical level and vice-versa.

If a decision to move IP flows to the optical level is taken, then the decision maker establishes a lightpath. The decision maker performs that by adjusting the routing and cross-connection tables of all nodes along the chosen path. On the other

hand, if a decision to move IP flows back to the IP level is taken, the decision maker tears down the established lightpath. For that, the decision maker reconfigures the routing and cross-connection tables of the involved nodes in order to release the lightpath. More details about our self-management module and its elements will be described later in Section 4.3.

2.4.2 Physical architecture

The physical architecture consists of the physical location of the functional blocks. In our physical architecture, the functional blocks are located at two different physical locations: in the multi-service hybrid devices and in an external management device (Figure 2.13).

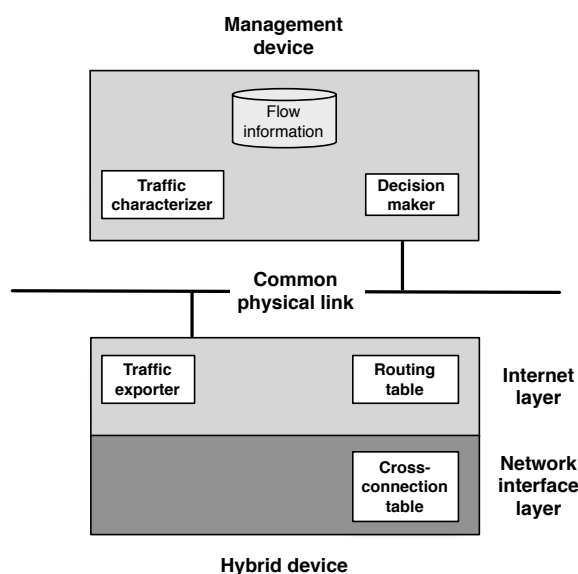


Figure 2.13: Physical architecture.

The traffic exporter and the routing & cross-connection tables are located inside the hybrid devices. On their turn, the self-management elements are located in one external management device. This management device is in charge of managing more than one hybrid device (also known as 1:N management relationship), which is widely used in different management areas [136] [145] [65].

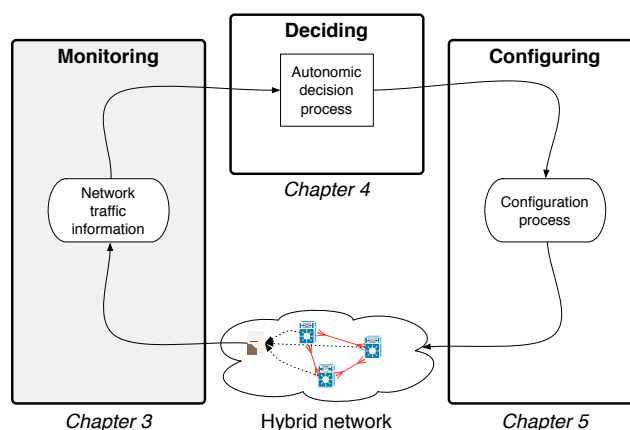
2.5 Concluding remarks

THIS chapter reviewed the conventional approaches to manage lambda connections in hybrid optical and packet switching networks. In practice, two approaches are currently used: a direct management approach, which is based on SNMP or CLI manager-agent interactions, and an indirect management approach, which has GMPLS as its main technology. Both approaches, however, require human interaction in order to select eligible elephant flows to the optical level as well as to create and release lightpaths. As discussed in Section 2.2, conventional approaches are therefore prone to be slow when managing lightpaths. In order to overcome that, we introduced in this chapter the use of self-management to automatically detect elephant flows and manage lightpaths for those flows. Along with our self-management proposal, we also introduced its functional and physical architectures. The functional architecture defines which functional blocks and their interactions are needed to perform self-management in a logical and comprehensible way. On its turn, the physical architecture defines the physical locations of these functional blocks.

Chapter 3

Monitoring of network traffic

Network monitoring techniques provide vital information for network operators to take management decisions. A proper monitoring process is thus essential for a good network operation. Within the context of this thesis, monitoring techniques provide information about IP flows (e.g., source IP address, source TCP/UDP port, amongst others) to our autonomic decision process. On its turn, the decision process analyses this information and takes decisions about which IP flows should be moved to/from the optical level, as depicted in the figure below. There exist several techniques for network monitoring, such as packet-based, SNMP-based, and flow-based. There are also some variants in the way these techniques are employed, such as the use of sampling to reduce the amount of monitored data. In this chapter, we focus on techniques for network monitoring that are relevant for our self-management approach as well as on the potential flow information they can provide to our autonomic decision process.



The organization of the present chapter is as follows:

- Section 3.1 identifies the network parameters that may be relevant for our autonomic decision process.

- Sections 3.2 and 3.3 evaluate these parameters. Section 3.2 evaluates parameters that are used to identify flows, whereas Section 3.3 evaluates parameters that provide behavioral flow information.
- Section 3.4 presents possible techniques to collect IP flow information. It is also discussed in this section which technique is the most relevant for our self-management approach.
- Section 3.5 analyzes the effects of using sampling, which is widely employed in large networks, on the relevant parameters.
- Section 3.6 concludes this chapter.

3.1 Potential network parameters

NETWORK parameters provide valuable information about the status of network traffic (*e.g.*, traffic volume) and devices (*e.g.*, routers). With regard to our autonomic decision process, network parameters are used to predict the behavior of flows transiting a network. Our objective in this section is to define a list of parameters that we believe to be beneficial for our autonomic decision process to identify large IP flows.

In order to define a set of relevant parameters for our autonomic decision process, a literature study has been carried out first. Network parameters have been taken from the following sources: (a) MIB modules IP-MIB [128], TCP-MIB [124], UDP-MIB [53], Interfaces Group MIB [99], RMON-II [161], and SMON [165]; (b) the information model for the IP Flow Information eXport (IPFIX) protocol [120]; and (c) IPv4 packet structure [172], TCP header structure [176], and UDP header structure [177]. Since these sources also deal with information not related to flows (*e.g.*, MAC to IP address translation in IP-MIB), we have selected the subset of information that we first believed to have potential to predict the volume of large IP flows. The outcome of this study is the classification of network parameters divided into two main groups: *flow identification parameters* and *flow behavior parameters*.

Flow identification parameters define a sequence of packets that share the same fields. Since the number of fields a flow can have is rather extensive [120] [33], for the analysis later in this chapter, we limited this number as follows:

1. *IP addresses*: identify network interfaces (connected to a certain network). IP addresses are important because some devices can generate more traffic than others. For example, a file server is expected to generate more traffic than an ordinary desktop.

2. *TCP/UDP port numbers*: represent the communication end points that network applications use to exchange data via transport protocols (*i.e.*, TCP and UDP). Following the same reasoning above, some applications can generate more traffic than others. For example, a Telnet session is expected to generate less data than a FTP session.
3. *Network segments*: are portions of a computer network, which can vary in scale from small networks (*i.e.*, LAN) up to larger ones (*i.e.*, Wide Area Network (WAN)). Depending on the size that a network may have, a network segment can include smaller network segments. For example, the University of Twente (UT) network belongs to the Dutch national research network, SURFnet6 [147]. Similar reasoning as before, we believe a network segment can generate more traffic than others. The following network parameters can be used to identify network segments:
 - *Subnet*: Continuous bits in an IP address prefix used to identify a subnet;
 - *Autonomous System*: A collection of IP routing prefixes.

Flow behavior parameters characterize the behavior of flows. The term behavior refers to the activity of a flow, which can still be in progress (partial activity) or be completed (total activity). The potential flow behavior parameters that may be relevant to our research are the following:

1. The number of *bytes* transferred thus far is the most important parameter to be considered, since our autonomic decision process aims at offloading high volume flows. We consider therefore the number of bytes as the target parameter.
2. The number of *packets* transferred thus far may give a good indicative about the size of a flow.
3. Literature [181] [20] [144] [24] has shown that some large flows may also be long in *duration*, normally presenting a heavy-tail distribution. Duration can therefore be a potential input parameter for our autonomic decision approach.
4. *Throughput* is the average rate of a communication. It is usually expressed in bytes per time unit (*e.g.*, Bytes per second (Bps)), but it can also be measured in packets per time unit (*e.g.*, Packets per second (Pps)). Both Bps and Pps will be considered for the analysis in the remainder of this chapter.

In order for our self-management approach to take the right decisions about moving flows between IP and optical levels, the selection of appropriated network parameters is important. For that, we evaluate the aforementioned identification and behavior parameters while observing the size of the flows, *i.e.*, their number of bytes. This evaluation is the main focus of the following Sections 3.2 and 3.3. Section 3.2 presents the evaluation of parameters that are used to define flows (*identification parameters*). On its turn, Section 3.3 shows the evaluation of parameters that provide information about the activity of flows (*behavior parameters*).

3.2 Evaluation of identification parameters

IN this section, our goal is to investigate the effect that different flow definitions may have on the flow size [58]. Our approach to investigate that consists in: 1) collecting flow information from a real network, and 2) varying the definition of the collected flows while observing flow size. The following subsections present how network data was collected from SURFnet6 (our testbed) as well as how our analysis was performed.

3.2.1 Measurement setup

For our analysis we measured traffic on the SURFnet6 network for two weeks (July 20, 2006 - August 3, 2006). The core routers — NetFlow-enabled — of SURFnet6 exported information about all network traffic within SURFnet6, at that time, in no more than 5 minutes interval. Since SURFnet6 is a high-speed network (10 Gbps links and up), technological issues, such as limited processing resources, prohibited exporting NetFlow records for all packets therefore. A packet sampling ratio of 1 out of 100 packets was used to reduce the amount of NetFlow records.

In order to facilitate our measurements, SURFnet routed NetFlow data from its core routers to a NetFlow collector located at the UT. At the UT domain, a machine was setup to dump the incoming NetFlow stream into so-called *pcap* files, using the *tcpdump* tool [149]. We decided to use *tcpdump* instead of a real NetFlow collector software: this way, we could store the NetFlow stream in its “raw state”. This allowed us to replay our analysis without the need for SURFnet to retransmit the data.

The total amount of data that was received from SURFnet in this two weeks period was about 81 GB. This includes all UDP, IP, and pcap overhead (e.g., packet timestamps). The total amount of NetFlow reported bytes was 4.0 TB. However, since SURFnet uses 1:100 sampling with NetFlow, this accounts for some 0.40 PB of the actual network traffic.

3.2.2 Analysis setup

The IETF has been working on the standardization of flow-based measurements. One of the results of this standardization process is the specification of IPFIX standard [121]. According to IPFIX a flow is defined as having some common properties (flow fields). The most traditional definition for a flow is the 5-tuple definition. This definition identifies a flow as a set of IP packets that share the same values for the following 5 fields: source IP address, destination IP address, source port, destination port, and protocol. However, different definitions for flows do exist, by choosing coarser or finer flow definitions [57] [59]. Within this context, our analysis setup consisted of using different flow fields than the traditional 5-tuple to define IP flows. That resulted in different levels of granularity for a flow as depicted in Figure 3.1 and to be discussed next.

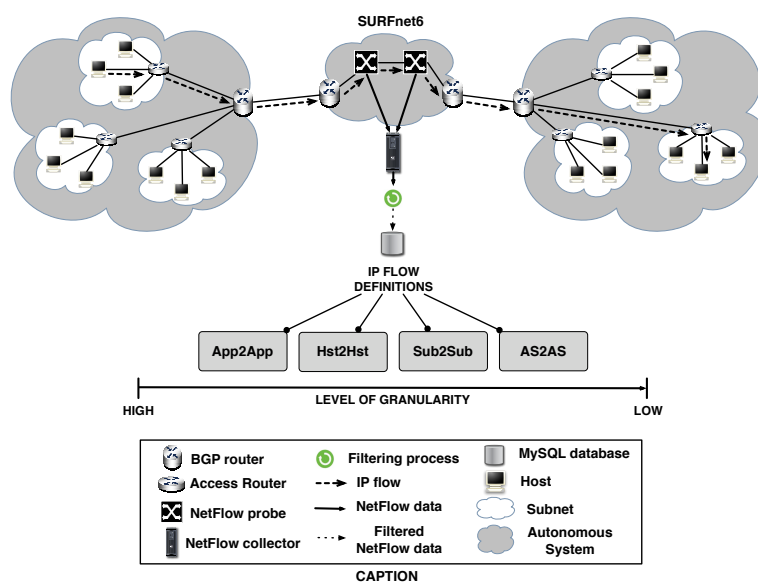


Figure 3.1: The different definitions for a flow.

The higher the granularity of a flow definition is, the more restrictive the flow definition will be when grouping packets into flow sets. Our flow definitions go from a high level of granularity to a low level of granularity as follows:

1. *App2App*: the traditional 5-tuple flow definition;
2. *Hst2Hst*: group of packets with the same source and destination IP addresses;

3. *Sub2Sub*: group of packets matching the most significant bits of the source and destination IP addresses;
4. *AS2AS*: group of packets with the same source and destination autonomous systems.

Given the above mentioned definitions for a flow, we observe how their usage affect flow size. We only observed flows that were considered to be big enough to be moved to the optical level, viz., above or equal to 50.112 Mbit/s (*i.e.*, the OC-1 payload rate in SONET networks [5]). Since SURFnet exported flow information in 5 minutes interval, we deliberately selected flows which minimum size was greater or equal to 14.68 Gbits ($50.112 \text{ Mbps} \times 300 \text{ seconds} = 14.68 \text{ Gbits}$).

3.2.3 Results

The following results only consider flows that satisfied our threshold value above mentioned.

Percentage of the total traffic

Figure 3.2 shows the percentage of the total traffic (4.0 TB) that each flow definition represents.

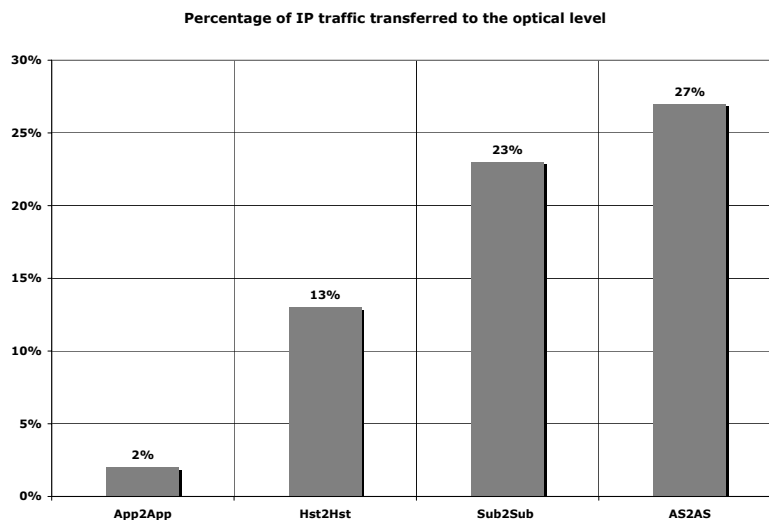


Figure 3.2: Percentage of the total traffic per flow definition.

App2App is the flow definition that least represents the IP traffic moved to the optical level (2% of the total traffic). The reason for that is that *App2App* is too restrictive to group packets into flows, which results in flows with small size. There is although a significative improvement in the percentage of IP traffic moved to the optical level when not considering port numbers. That is shown by the *Hst2Hst* flow definition, which moved, in our analysis, almost 7 times more (13%) IP traffic to the optical level than the *App2App* flow definition. Likewise, *Sub2Sub* and *AS2AS* flow definitions moved, respectively, almost 12 and 14 times more. It can be concluded therefore that the lower granular a flow definition is, the more packets will be grouped into flows, and, thus, the more traffic is selected to be moved to the optical level.

Average flow size

Figure 3.3 complements Figure 3.2 by showing the average size of flows using the flow definitions considered in our analysis. It is worth mentioning that we used a confidence interval of 95% to calculate the confidence limits of our average values.

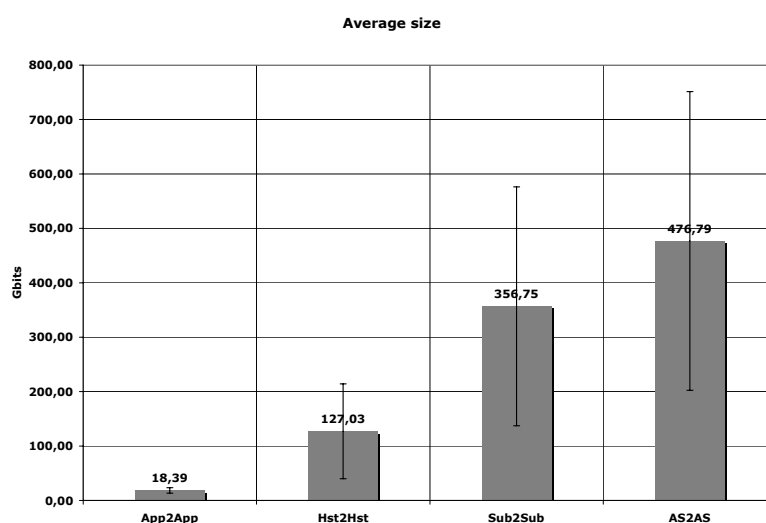


Figure 3.3: Average flow size with a confidence interval of 95%.

The average flow size of *App2App* flows was 18.39 Gbits, which was slightly above our criteria, *i.e.*, flow minimum size greater or equal to 14.68 Gbits. The remaining flow definitions presented a much higher average flow size than *App2App*

flows, but with a considerable variance in the flow size. For example, *AS2AS* flow definition had big flows, such as above 700 Gbits, but also smaller ones, such as 250 Gbits. This can be explained by the fact that when flow definitions with lower level of granularity are used, different kinds of traffic (*e.g.*, VoIP and P2P traffic) may be mixed, presenting therefore different sizes. By the same token, when more restrictive flow definitions are used, similar kinds of traffic are grouped, presenting thus smaller variance in size.

3.2.4 Concluding remarks

Based on the analysis presented in this section, we can conclude that the use of different definitions for a flow have significant influence on the flow size. The average size of a flow increases when lower granularity to define flows are used (Figure 3.3). In average, the size of flows varied, in our analysis, from small sizes (18 Gbits), in the case of *App2App* flows, to big sizes (1.7 Tbits), in the case of *AS2AS* flows. Lower granular flow definitions also resulted in more IP traffic being moved to the optical level, as presented in Figure 3.2.

Even though the obtained results were somehow already expected, we did not know beforehand any quantification for the observed variation in flow size and the percentage of offloaded traffic. The main contribution of our analysis was thus to show how much the size of the flows eligible for lightpaths vary according to the flow definition used. It is worth mentioning that this variance is also in accordance with the self-similarity studies found in the literature [36] [153] [23].

Within the context of our self-management approach, lower granular flow definitions contribute better to offload more IP traffic to the optical level, being therefore the most relevant flow definitions to our autonomic decision process. Our autonomic decision process is designed although to work independently of the flow definition used, since networks operators may employ different flow definitions on their managed networks. However, for further analysis in this thesis, we stick with the traditional 5-tuple flow definition. That is due to practical reasons, since our measurements are performed on networks that use such flow definition.

3.3 Evaluation of behavior parameters

IN this section, the potential behavior parameters introduced in Section 3.1 will be evaluated. Our goal here is to observe which set of these parameters are relevant estimators to predict flow size. For that, we introduce a statistical methodology to validate our goal [55]. The following subsections present how network data was collected to our experiment as well as how our statistical methodology is comprised.

3.3.1 Measurement setup

To avoid any disturbance of sampling, we decided to consider only NetFlow data with a sampling ratio of 1:1 (1 packet out of one) from the UT network. NetFlow records over an entire week from Sep 12, 2008 to Sep 19, 2008 were collected (our population). However, due to the large amount of data, we chose a single business day (our sample) in order to speed up our analysis. NetFlow records over the entire day Thursday Sep 18, 2008 were considered in the evaluation of our behavior parameters. The records were captured in *nfcapd* format and then stored into a MySQL database. The amount of collected NetFlow records stored in our MySQL accounted for Sep 18 was 30.51 GB. These records were then combined in order to reflect the real behavior of the flows along the considered day. The MySQL table storing the combined NetFlow records into flows accounted for Sep 18 was 26.08 GB.

3.3.2 Statistical analysis

Our statistical methodology comprised in applying descriptive analyses to evaluate how the flow behavior parameters number of packets, duration, Bps, and Pps influence flow size. For that, we start with basic *summary statistics* [174] like mean, median, amongst others, followed then by more advanced statistical analyses, namely *conditional probability* [105], *correlation* [169], and *classification tree* [168], respectively.

Summary statistics

The initial number of flows considered in our statistical analysis was 378.363.608, which represented a total traffic of 18.11 TB on Sept 18, 2008. We started our statistical analysis by defining the set of flows we are focused on. This set of flows is based on the target our self-management of lightpaths aims at, *i.e.*, the *elephant flows*. Contrary to the analysis presented in Section 3.2, we do not restrict the elephant flows to the minimum size of 14.68 Gbits. We aim here at observing these flows on its normal behavior, *i.e.*, without any restriction imposed by the optical level.

We focus our analysis therefore on flows that have the following characteristics: (a) few in number, (b) persistent in time, and (c) represent most of the traffic. Out of the total number of collected flows (378.363.608), a very small percentage of them (0.82%) accounted for the biggest percentage (97%) of the total traffic. This percentage of flows (3.092.885 in numbers), from now on referred as big flows, matches the characteristics (a) and (c) previously mentioned.

Our next step was to check, out of the big flows, the ones persistent in time. Persistent means that they do not have a short duration, as it is the case of bursty flows [87]. Table 3.1 shows some summary statistics about the duration of our big flows. It

Flows	3.092.885
Mean	274 sec
Median	19 sec
Minimum	0 sec
Maximum	86.507 sec
Percentiles (25%)	12 sec
Percentiles (50%)	19 sec
Percentiles (75%)	57 sec

Table 3.1: Summary statistics on the duration of our big flows.

also shows that most of these flows (75%) have a considerable short duration, *i.e.*, a duration shorter than 57 seconds. This allows us to conclude that most of the flows are short-lived. Interestingly, some flows lasted during the entire day (86.400 seconds). Few of them lasted even more than that due to the active timeout value of 120 seconds configured in the UT NetFlow-enabled router. That would allow flows to have therefore a maximum reported duration of 86.520 seconds.

Conditional probability

Summary statistics provide a good overview about the overall duration of flows, but do not say much about how persistent these flows are over time. Our objective here is to know how long a flow should be active before we can predict, with reasonable certainty, that that flow will be persistent in time. In order to have a better insight about the persistence of these flows we used conditional probability [105]. Conditional probability is the probability of some event A happening given the occurrence of some other event B ($P(A | B)$). In our analysis, we are interested in knowing the probability of flows being persistent in time. For that, we observe the conditional probability of flow duration as follows. Given that the duration (D) of a flow has already lasted at least a certain amount of time B ($D \geq B$), what is the probability this flow will last for at least another 60 seconds ($D \geq B + 60$)?

Table 3.2 shows the conditional probability for the duration (in seconds) of our big flows. It shows that there is a small probability (24%) that a flow will last at least another minute, given the fact that it has just started. However, there is a considerable improvement (67%) in this probability when flows have elapsed at least one minute. This probability gets more stable the longer a flow has already elapsed. This allows us to conclude that the longer a flow has already elapsed, the smaller the

$P(D \geq B + 60 \mid D \geq B)$	Percentage
$P(D \geq 60 \mid D \geq 0)$	24%
$P(D \geq 120 \mid D \geq 60)$	67%
$P(D \geq 180 \mid D \geq 120)$	78%
$P(D \geq 240 \mid D \geq 180)$	83%
$P(D \geq 300 \mid D \geq 240)$	86%
$P(D \geq 360 \mid D \geq 300)$	89%
$P(D \geq 420 \mid D \geq 360)$	90%
$P(D \geq 480 \mid D \geq 420)$	91%
$P(D \geq 540 \mid D \geq 480)$	92%
$P(D \geq 600 \mid D \geq 540)$	92%

Table 3.2: Conditional probability for the duration of our big flows.

probability of ending in the next time period; viz., the duration of flows presents a heavy tail behavior. It is worth saying that taking advantage of heavy tails can contribute to the design of computer systems, as already concluded by Crovella [35].

Based on Table 3.2, we choose an elapsed duration of 5 minutes to define a flow as being persistent in time. The reason for that comes from the fact that the percentage of flows lasting for at least another minute gets relatively stable (around 90%) when flows reach a minimum duration of 5 minutes. It is relevant to say that, in our analysis, flows with duration below 5 minutes did not represent a considerable amount of traffic (26% of the total traffic), whereas flows with duration above or equal to 5 minutes represented 74% of the total traffic.

One may rise the question whether 5 minutes is the duration that should always be considered to define a flow as being persistent. We believe the answer is no, cause IP flows can vary their behavior depending on several factors, such as time of the day, kind of network, network users, amongst others. It is interesting to highlight that we introduce here, although, a generic approach to find out flow persistence.

Correlation techniques

Based on the outcome of the conditional probability applied on our traces, another filtering was done in the 3.092.885 flows in order to remove flows with a duration shorter than 5 minutes. This resulted in the selection of 283.783 flows (0.07% of the total number of collected flows) having duration above or equal to 5 minutes. These

are thus the set of flows of which metrics throughput (Pps and Bps), packets, and duration are observed in relation to flow size (number of bytes).

Once defined the set of flows, we first observe how those metrics relate with flow size by drawing correlation charts (Subsection 3.3.3). Correlation charts provide us a visual impression of what the correlation is, but they do not express the quantification of how strong or weak the correlation is. To obtain quantitative correlation values regarding the considered metrics, we used Pearson's correlation method [115]. Pearson's correlation computes the pairwise associations for a set of variables and displays the results in a matrix. This is useful for determining the strength and direction of the association between two metrics.

In our context, there is an unmistakable intuition that Bps and duration have the strongest correlation to tell about the size of flows. We use Pearson's correlation, however, to see if this fairly obvious correlation may contain some other unsuspected correlations. Applying therefore correlation analysis on our data can lead to a better understanding (Subsection 3.3.4).

Classification tree

Even though Pearson's correlation is a good method to quantify the relationship between two metrics, it does not consider the combination of more than two of them. It could be therefore that an interaction of more than two metrics could give a better refinement about the best predictors for flow size. In order to find that out, we used a classification tree technique, called CHi-squared Automatic Interaction Detector (CHAID) method, which is a kind of decision tree widely used in data mining areas. CHAID divides a data set into exclusive and comprehensive partitions that differently relate with an observed dependent variable [84]. These partitions are defined by a tree structure and they are classified in descendent order of independent variables, called *predictors*. For each partition of predictors, CHAID assigns a probability of response. All probabilities are subsequently used to rank the partitions with the strongest relation with the dependent variable. It is worth mentioning that the partitions of each predictor are merged if they are not significantly (significance level of 0.05%) different in regard to the dependent variable. More details about decision trees and the CHAID algorithm can be found in the Appendixes A.1 and A.2, respectively. In the case of our analysis, CHAID calculates which independent metrics – duration, packets, Pps, and Bps (the predictors) – have the strongest relation with flow size (the dependent variable). The outcome of our analysis by using CHAID is presented in Subsection 3.3.5.

The potential flow behavior parameters have been statistically evaluated by observing how they one by one contribute to identify flows that generate large amounts

of data. The following subsections present first the results regarding our correlation analysis, followed then by the CHAID classification tree.

3.3.3 Correlation charts

Figure 3.4 shows the relation of packets, duration, Pps, and Bps with the number of bytes (flow size), respectively. All figures present both axes with logarithmic scale. The figures show that there is a linear relation between packets, Bps, and Pps with bytes, except for the case of duration. Figure 3.4(b) shows that a flow can have a long duration and a small amount of bytes, a short duration and a large amount of bytes, and all in between. Meanwhile, the metrics packets, Bps, and Pps walk along with bytes. The bigger those metrics are, the bigger the flow size is expected to be.

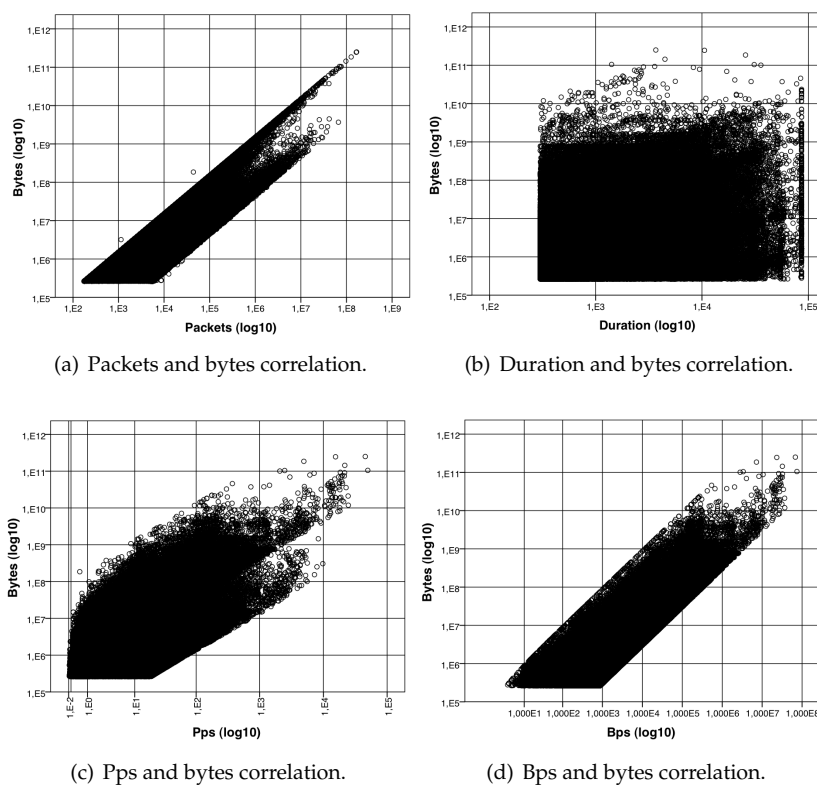


Figure 3.4: Correlation between considered behavior parameters and flow size.

It is worth mentioning that there is a certain degree of linearity among those metrics. Figure 3.4(a) shows a strong linear relationship between the number of packets and flow size. Figures 3.4(c) and 3.4(d), in turn, show a slightly higher variability in the linearity when compared to packets. Bps and Pps linearities with flow size are influenced by flow duration, which presents a strong variability. For example, we have seen cases in which a lot of packets were generated in a short duration (high Pps throughput), but also a small amount of packets generated in a long duration (low Pps throughput).

3.3.4 Pearson's r correlation

Correlation charts can give some insights on the linearity between two metrics, but they do not precisely quantify it. In order to quantify the correlation between two metrics, we used Pearson's r correlation, which is defined as the sum of the products of the standard scores of the two measures divided by the degrees of freedom:

$$r = \frac{1}{n-1} \sum_{i=1}^n \left(\frac{X_i - \bar{X}}{s_x} \right) \left(\frac{Y_i - \bar{Y}}{s_y} \right)$$

where $\left(\frac{X_i - \bar{X}}{s_x} \right)$, \bar{X} , s_x , and n are the standard score, sample mean, sample standard deviation, and number of samples, respectively [107]. Table 3.3 shows the r correlation among our considered metrics.

	Packets	Bps	Pps	Duration
Pearson's r correlation for bytes	0.927	0.671	0.642	0.058

Table 3.3: Pearson's correlation bytes pairwise with other considered metrics.

A correlation of 0 (zero) means that there is no linear relationship between two metrics. Whereas, a correlation of 1 means that there is a strong positive linear relationship between two metrics. As Table 3.3 shows, there is a strong linear relationship between packets (0.927), Bps (0.671), and Pps (0.642), with bytes. In contrast, duration does not present a strong linear relationship (0.058). This means that duration should not be exclusively focused on when trying to predict a flow size.

3.3.5 CHi-squared Automatic Interaction Detector

Since Pearson's correlation only shows the relation between two metrics, we used CHAID in order to see the relation of more than two metrics with the flow size.

For that, we first deliberately divided flow size into 4 categories according to their number of bytes.

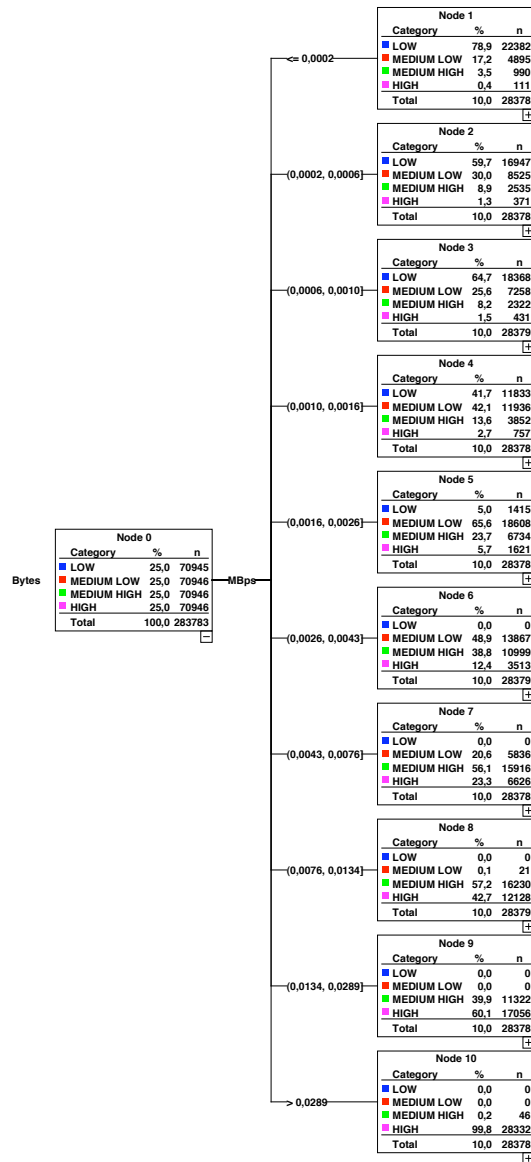


Figure 3.5: 1st depth of the CHAID classification tree.

The 25% biggest flows are referred as *HIGH* volume, whereas the 25% smallest flows are named *LOW* volume. The medium flow sizes (*i.e.*, the remaining 50%) are then divided into the 25% biggest medium flows, referred as *MEDIUM HIGH* volume, whereas the 25% smallest medium flows are named *MEDIUM LOW* volume. Once the flow volume is categorized, we observe how the considered parameters relate to these 4 categories. It is worth mentioning the number of categories can be arbitrarily chosen. Figure 3.5 partially shows the result of our CHAID classification tree. Only *node 0*, and *node 1* throughout *node 10* are shown for the sake of space. Bytes is the dependent variable and it is represented by the *node 0* in the CHAID tree. *Node 0* contains the 4 aforementioned categories for the flow size. The CHAID method then starts dividing the predictors (*i.e.*, Bps, Pps, duration, and packets) into partitions (nodes) and cross-tabulating them against the dependent variable (*node 0*). The predictor that presents the smallest level of significance (*i.e.*, the most statistically significant relationship with the dependent variable) is placed at the first depth of the CHAID classification tree along with its partitions. CHAID chooses Bps (represented in MBps in Figure 3.5) as the best predictor for flow size as it is ranked right to *node 0*. From *node 1* to *node 10* it is possible to see how each group of flows, classified by their Bps throughput, influences flow size. Flows with small Bps throughput tend to be small in flow size (*e.g.*, *node 1*). On the other extreme, flows with big Bps throughput tend to generate large flows (*e.g.*, *node 10*).

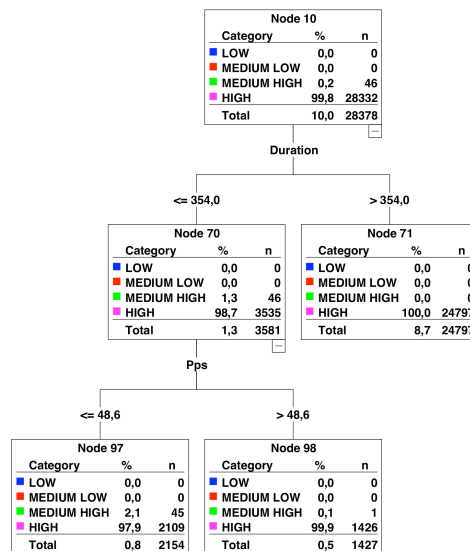


Figure 3.6: Disposition of duration and Pps in the CHAID classification tree.

After the CHAID method selects the first level predictor and its best merged partitions, it begins to place other predictors beneath the initial predictor. The CHAID method continues this procedure until further sub-divisions cannot be performed. That is shown in Figure 3.6, in which we only expanded *node 10* to show how the other metrics are disposed in the CHAID classification tree. The second best predictor pointed out by the CHAID method is flow duration (represented in seconds in Figure 3.6). Our partially displayed CHAID classification tree shows that flows with high Bps and long durations are those that have the higher flow size (*node 71*). Finally, the third best predictor is Pps, being disposed in the last level (depth) of the CHAID classification tree. The number of packets did not show a significant prediction in our model, and it was ignored by the CHAID method.

Observed	Predicted				Accuracy
	LOW	MEDIUM LOW	MEDIUM HIGH	HIGH	
LOW	65.499	5.446	0	0	92.3%
MEDIUM LOW	6.381	60.334	4.231	0	85.0%
MEDIUM HIGH	521	3.016	62.835	4.574	88.6%
HIGH	63	48	7.257	63.578	89.6%
Percentage	25.5%	24.3%	26.2%	24.0%	88.9%

Table 3.4: The accuracy of our CHAID classification model.

CHAID also provides a measure of confidence that the classification tree is correct (Table 3.4). Table 3.4 shows the accuracy of our classification tree for the 4 observed flow volume categories. For example, out of 70.946 flows categorized as *HIGH* volume, 63.578 of them were correctly predicted as such (accuracy of 89.6%). Meanwhile, 63, 48, and 7.257 flows were incorrectly predicted as *LOW* volume, *MEDIUM LOW* volume, and *MEDIUM HIGH* volume, respectively. An inaccuracy of 10.4%. In general, our classification tree correctly classifies about 89% of the flows while misclassifying a flow size only in about 11% of the cases. That high accuracy allows us to assume that our CHAID classification tree correctly selects, for the most part, the proper predictors for the flow size.

3.3.6 Summary and conclusions

The behavior parameters were evaluated with relation to flow size by using descriptive statistical methods. We started defining the set of flows we were interested on, *i.e.*, the elephant flows. For that, we first employed summary statistics, which provided an overview on the duration of flows (Table 3.1). Summary statistics do not

say much although about how persistent flows are. We then used conditional probability in order to know more about the persistence of flows over time. We found out that flows with a minimum duration of 5 minutes have 89% of chance of continuing running for at least the next minute (Table 3.2). Based on that, we deliberately assumed 5 minutes to define a flow as being persistent. We highlight here although that our main contribution is not the value of 5 minutes itself, but the approach we used to reach it.

Our next statistical step was to find out correlation among the considered behavior parameters. Even though initially some metrics such as duration and Bps were intuitively expected to influence flow size, little was known about how strong this influence could be. Moreover, we were not aware if there were any other unsuspected parameters (*e.g.*, Pps and packets) that could have significant influence to flow size. We first then used correlation charts (Subsection 3.3.3) to get some knowledge about the correlation amongst the considered metrics. Our correlation charts (Figure 3.4) provided some good indications about the correlations, but it did not quantify how strong they were. To solve this uncertainty, we used Pearson's r correlation (Subsection 3.3.4). Pearson's correlation showed that packets ($r = 0.927$), Bps ($r = 0.671$), and Pps ($r = 0.642$) have a strong linear relationship with flow size, while duration ($r = 0.058$) has not. Since all parameters have certain influence on flow size, they should not be used alone, but in groups. We used finally the CHAID technique to analyze that (Subsection 3.3.5). As evaluated by CHAID, Bps and duration are the best predictors for flow size, followed by Pps (as a refinement parameter). Even though packets parameter was considered by Pearson's correlation as the parameter with the strongest linear relationship with flow size, it was dismissed by CHAID.

In conclusion, CHAID statistically confirms the intuition that Bps and duration are the metrics to be considered when observing flow size. In order of importance, Bps, duration, and Pps (as an optional refinement) are the best predictors. These metrics have more impact on the flow size than the number of packets, which was ignored. Our self-management of hybrid networks should therefore take them into account when taking decisions on moving flows to the optical level.

3.4 Possible techniques for monitoring IP data

NETWORK monitoring techniques provide important information on network traffic transiting in a network. Within the context of this thesis, monitoring techniques play an important role by supplying information on IP data to our automatic decision process. In this section, we aim at selecting a monitoring technique to be used with our self-management approach. For that, we first start by defining

the requirements monitoring techniques should satisfy in order to be considered relevant to our self-management approach. We then select the set of techniques to be evaluated based on the level of detail they provide for IP data, namely, *packet-based techniques*, *SNMP-based techniques*, and *flow-based techniques*. Finally, these techniques are evaluated while observing their relevance to our self-management approach.

3.4.1 Requirements

The considered monitoring techniques are evaluated based on a set of requirements, which are mostly based on the evaluations performed in Sections 3.2 and 3.3 and stated as follows:

1. *The monitoring technique has to provide means to differentiate IP packets.* As concluded in Subsection 3.2.4, IP packets can be grouped into flows through different definitions. Such definitions can go from highly granular to slightly granular, which results in small or big flows. Providing therefore means for our self-management approach to differently group IP packets into flows can contribute to offload large IP flows to the optical level.
2. *The monitoring technique has to provide means to calculate the behavior of IP flows.* As concluded in Subsection 3.3.6, Bps and duration (and optionally Pps) are the best predictors for flow size. Monitoring techniques should therefore provide the following information to calculate those predictors: duration, number of bytes, and number of packets (optional).
3. *The monitoring technique has to be feasibly employed on high-speed networks.* Since we focus our self-management approach on high-speed networks, the monitoring technique should be able to cope with large amounts of network data. Packet loss, for example, during the measurement process is undesirable, since network information can be inaccurately reported.

Even if other requirements could be listed here, these are the most important ones. The next subsections present details of the considered monitoring techniques.

3.4.2 Packet-based techniques

Packet-based techniques can be subdivided into those that collect complete IP packets (*i.e.*, header + payload) or those that collect only some part (*e.g.*, the header) of them. The header contains addressing and control fields, while the payload carries

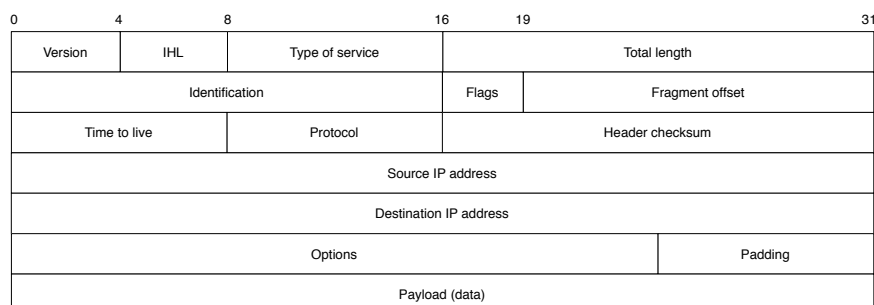


Figure 3.7: IP packet format.

the actual data (transport headers plus application data) to be sent over the network. Figure 3.7 shows the format of an IP packet.

A tool commonly used to collect packet information is *tcpdump* [149]. *Tcpdump* can act as a packet sniffer that runs from the command line interface. It allows the user to intercept and display packets being transmitted over a network of interest.

The collection of full packets is useful when detailed network analysis, such as those used to detect certain Denial of Service (DoS) attacks are in place [106]. In this case, not only the header is considered, but also the payload itself. The analysis of the payload content comes handy when there is a need to look for specific (malicious) contents, such as buffer overflow codes trying to exploit some vulnerability at a certain network server. By analyzing the payload of IP packets, one could detect a possible attack attempt.

As an alternative to collect full packets, network operators commonly choose to collect only the packet headers. In this situation, information about the payload of a packet is not considered. Collecting only packet headers is useful when the information about the payload is irrelevant, such as in cases of network troubleshooting or network traffic optimization (*e.g.*, load balance).

As an example, an average of 25 billions packets transited the UT network during working days in September 2008. In order to fully collect these packets (considering an average packet size of 1500 octets), approximately 34 TB of storage space would have been needed per working day. In comparison, approximately 1.4 TB of storage space would have been needed per working day, if only the packet headers (*i.e.*, nearly the first 60 octets of each packet) would have been collected.

The main advantage of using packet-based techniques is the finest level of detail for network traffic (*i.e.*, individual IP packets) that is obtained in real time. The level of detail combined with real time information result in a very accurate calculation of the throughput. However, such accuracy comes at a cost: the high volume of mea-

surement data. Dealing with such large amount of data can be very expensive. The collection process may require the acquisition of expensive monitoring stations with large storage capacity and high processing power. The acquisition cost can become worse when the collection of packet traces is employed on high-speed networks. In such networks, not only storage capacity and processing power are required, but also expensive network cards. These cards may be required to handle data transfer rates at gigabit scale. The non-employment of such high-speed cards may result in potential packet loss, disrupting therefore the collection process [88]. Based on the aforementioned reasons, packet-based techniques are frequently refrained from being used on high-speed networks.

3.4.3 SNMP-based techniques

The Simple Network Management Protocol can also be used to monitor IP traffic. SNMPv1 and SNMPv2 (historic versions) are the versions mostly used by network operators, whereas SNMPv3 (the standard version) is rarely employed [139]. Examples of MIBs used for monitoring IP traffic are MIB modules IP-MIB [128], TCP-MIB [124], UDP-MIB [53], and Interfaces Group MIB [99].

Flow information can be collected as well via Remote Monitoring (RMON) [160], RMON-II [161], and Meter MIB [19]. These MIBs are either available on routers and switches or they are installed on dedicated instruments, called probes. Probes are in charge of monitoring data packets crossing the network at certain key points (*e.g.*, a WAN interface on a router). The use of probes aims at providing real-time traffic information while not introducing additional load on the routers.

The data collection on such MIBs commonly employ the pull mode. By using such a model, network devices running SNMP agents are regularly requested through SNMP messages to send network data. This regularity is commonly referred as polling interval. A common polling interval used is 5 minutes, but it can be adjusted to minor or major orders.

The main advantage of using SNMP-based techniques is regarding the amount of measurement data. Compared with packet-based techniques, SNMP-based techniques require less monitoring data. Comparing with the same example presented in Subsection 3.4.2, approximately 742 MB of storage space would have been needed to collect SNMP messages per working day in September 2008. This required storage space is based on a polling frequency of 90 SNMP *Get-next* messages per second and with a message size of 100 bytes. The polling frequency and message size values were derived from the analysis performed by Schönwälder *et. al* [139].

However, this low volume of measurement data comes at the cost of a coarse-grained level of detail for network traffic. SNMP-based techniques provides highly

aggregated information about the measured network traffic [62]. For example, while packet-based techniques may provide information about individual packets passing through a monitored router interface, SNMP-based techniques would provide a simpler counter of how many packets have passed. Moreover, this low volume of measurement data may weigh against information gain, such as, for instance, the lack of information about specific application traffic [143].

The main disadvantage although is regarding the use of pull mode. When the pull mode is employed, information about flows are only sent when a request occurs. In a situation where memory shortage on SNMP agents happens, flow information can be discarded to give place to new information [182]. On high-speed networks, where the volume of flows is very high, the pull mode may not scale, resulting in flow information being discarded due to memory shortage. As a consequence, the activity of IP flows may be inaccurately reported.

3.4.4 Flow-based techniques

As an in-between technique for packet-based and SNMP techniques, flow-based techniques consist of collecting data about a set of packets (*i.e.*, flows) rather than individual packets. There are some technologies to collect flow information from managed networks. The most well-known and widely used is NetFlow [31], which is largely employed to collect flow information from Cisco routers. Although initially developed by Cisco, today several network devices of other vendors also employ NetFlow. In addition, NetFlow has been strongly influencing the definition of the IPFIX standard.

Cisco NetFlow

Cisco NetFlow is an embedded instrumentation software that is used to characterize network information. NetFlow can be used by network operators to analyze network usage or network security anomalies, to name a few of purposes. NetFlow characterizes flows by inspecting¹ packets when those are forwarded by a NetFlow-enabled router. Each packet is examined for a common set of IP packet fields. These fields are used to determine if a packet is unique or similar to others. In other words, these packets are used to define a flow of packets that share the same fields. Figure 3.8 depicts the process of collecting flow information with NetFlow.

All packets that share the same fields are grouped into a flow and then packets and bytes are tallied. This information is then stored in a cache on a NetFlow-

¹Packet inspection, employed in flow-based measurement, inspects network packets seen on the wire to update the appropriate fields of flow records. Inspected packets are not stored in the router's internal cache.

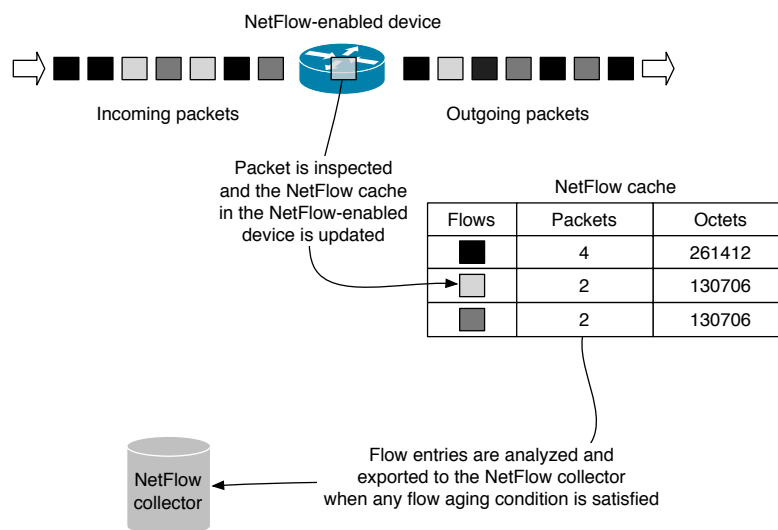


Figure 3.8: Collecting flow information with NetFlow.

enabled device. There are primarily two methods to access the flow information stored in the NetFlow cache. The first method consists of using CLI in order to issue commands to visualize flow information. The second method consists of configuring the NetFlow-enabled device to export flow information into a NetFlow collector (push model). In this case, the following steps are taken:

1. NetFlow is configured within a network device in order to inspect flow of packets and store information into the NetFlow cache.
2. NetFlow is configured to send flow entries in the NetFlow cache to a NetFlow collector.
3. A process constantly analyzes the cache searching for flows that have terminated or satisfied a condition to be exported.
4. Approximately 30 to 50 flow entries are bundled together and exported to the NetFlow collector.

The conditions for the NetFlow process to export flows are the following:

- A flow is inactive longer than an *inactive timeout*, i.e., the flow record was not updated in cache due to no new packets received for that flow;

- A flow is active longer than an *active timeout*, *i.e.*, the flow record has been constantly updated in cache for a period of time longer than the active timeout;
- The FIN and RST TCP flags of an observed packet indicate that the flow finished;
- The local NetFlow cache is full and needs to be flushed.

By default, NetFlow routers are configured with an inactive timeout of 15 seconds and an active timeout of 30 minutes. Other timeout values can although be employed, such as those described in Table 3.5.

Network	Inactive timeout	Active timeout
UT	30	120
SURFnet	30	300
GÉANT	60	300
<i>Default</i>	<i>15</i>	<i>1.800</i>

Table 3.5: *Inactive and active timeouts (in seconds).*

The default cache size is 64K flow cache entries, with each cache entry 64 bytes long. It is important to notice that, given the existence of the active timeout, long lived flows (*e.g.*, elephant flows) will be reported by multiple (yet complementary) flow records. For example, if the active timeout is set to 30 minutes and a flow lasts 120 minutes, this flow will generate 4 NetFlow records, each one reporting a duration of approximately 30 minutes. In order to calculate the real duration of this flow, the 4 NetFlow records have to be combined.

The usage of flow-based techniques considerably reduces the amount of collected data. Compared with the same example presented in Subsection 3.4.2, approximately 22 GB of storage space would have been needed to collect flow information from the UT network during working days in September 2008. This is nearly 1500 times less storage space than collecting the whole packets and nearly 65 times less than collecting only the packet headers. On the other hand, it is 30 times more than when using SNMP-based techniques.

The main advantage of flow-based techniques is that they require lower measurement volume than packet-based techniques. On the other hand, the amount of information is just a summary rather than full information about single packets. When compared to SNMP-based techniques, flow-based techniques generate

more measurement data, but provide a greater level of details for network traffic. More importantly, flow-based techniques, such as NetFlow, employ the push model, which exports flow information automatically without the need to wait for a request. By using the push model, one can prevent the overloading of resources (*i.e.*, memory) by simply exporting the information as soon as certain thresholds are about to be exceeded (*e.g.*, lack of memory space). Finally, a wide-range of routers and switches on today's high-speed networks are NetFlow-enabled. That results in a more cost-effective alternative since no extra hardware/software is needed to monitor IP flows.

The main disadvantage of using flow-based techniques is the loss of per-packet information. In situation where there is a need to know information about individual packets, then flow-based techniques should not be used. Flow-based techniques can also require high computation and memory requirement for the flow cache when the traffic volume is considerable high. However, in order to reduce this overhead, aggregation and sampling techniques can be employed.

3.4.5 Concluding remarks

Based on characteristics of each monitoring technique presented before, we can conclude the following. Packet-based techniques provide the finest level of detail on IP traffic and allows the calculation of its behavior with great accuracy. As a result, packet-based techniques satisfy our first two requirements. However, packet-based techniques are rarely employed on high-speed networks due to the cost of coping with high volume of measurement data. Packet-based techniques fail therefore to satisfy our third requirement.

On its turn, SNMP-based techniques generate far less measurement data than packet-based techniques at the cost of a coarser level of detail on IP traffic. Even with such coarse level of detail, SNMP-based techniques still satisfy our first and second requirements, since the amount of provided information still allow our self-management approach to differentiate IP packets, even if the amount of detail is not as many as with packet-based techniques. However, SNMP-based techniques use the pull model to provide data information. That means that information about flows are only sent when a request occurs. Since high-speed networks may generate a large amount of monitoring data, the pull mode may not scale well due to memory shortage, leading thus to flow information loss. Due to that SNMP-based techniques fail to completely satisfy our third requirement.

Finally, flow-based techniques can be considered as an in-between technique in terms of measurement data and level of detail. Flow-based techniques, on the contrary to SNMP-based techniques, use the push model. The push model prevents

flow information to be lost by exporting it before a memory shortage occurs. In addition, flow-based techniques, such as NetFlow, are available on today's routers and switches, being therefore widely employed in practice. Since flow-based techniques satisfy all our three requirements, we consider it as the most appropriated technique for our self-management approach.

Although at a first sight, flow-based techniques solve the scalability problem of the packet-based techniques, in reality flow-based techniques may also suffer the effects of larger networks. In continental wide networks, for example, the huge amount of traffic may generate an amount of flow records prohibitive to be used in actual scenarios. In this situation, ordinary flow recording is unfeasible; instead, traffic sampling takes place. Traffic sampling is widely employed in the measurement processes of high-scale networks in order to further decrease the amount of processed data and therefore reduce the consumption of storage and processing power. Due to its importance, more details about the use of sampling and its effects on the measurement of network traffic are presented next.

3.5 The effects of sampling on elephant flows

IN order to reduce storage space and processing power that the inspection of every single packet may rise, packet sampling is commonly employed on high-speed networks. As a result of sampling, the collected network traffic is an estimation rather than the real traffic. Our goal in this section is to observe whether the use of sampling on elephant flows may inaccurately report our metrics of interest, namely number of bytes, number of packets, and duration [54]. These metrics are needed to calculate Bps and Pps throughputs. In order to achieve our goal, we first start by presenting some sampling methods commonly used as well as related work. We then introduce our approach to observe the effects of sampling on our metrics of interest.

3.5.1 Sampling methods & Related work

Sampling methods are categorized between systematic sampling and random sampling [183]. The former selects the beginning of the sampling process and the duration of the selection intervals based on a deterministic function. For example, routers that inspect every n th packet, perform a systematic count-based sampling of 1 packet out of n ($1:n$). On the other hand, random sampling selects the beginning of the sampling process and the duration of the selection intervals according to a random process. In contrast to systematic sampling, random sampling requires the generation of random numbers. For example, routers can inspect every n th packet

out of N . In this case, n would be a random generated number in the range $[1, N]$. Figure 3.9 depicts one example of how the two sampling categories act.

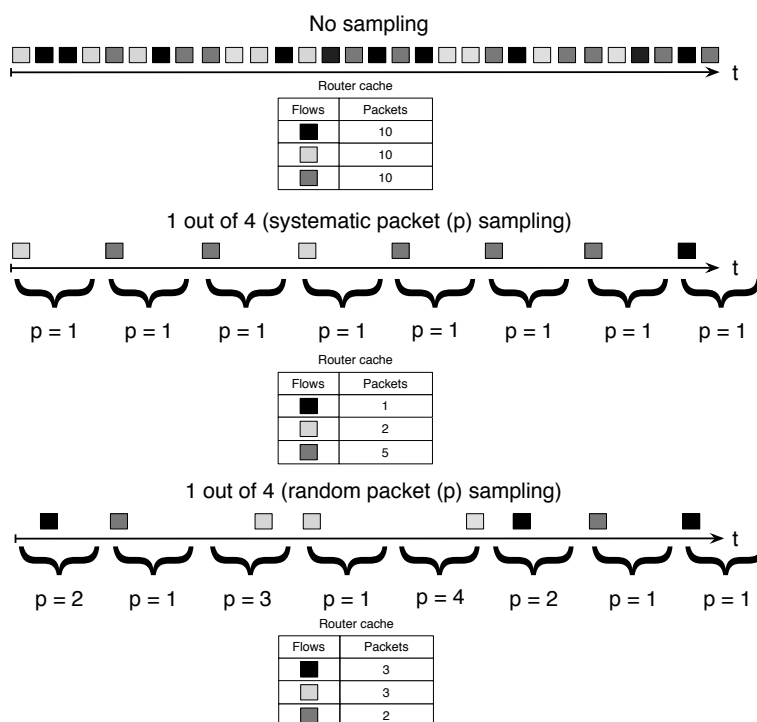


Figure 3.9: Example of no packet sampling, systematic packet sampling, and random packet sampling.

Observations on the effects of sampling in the monitoring of network traffic is considerably found in the literature [108] [123] [104] [184] [47]. Most of these works are highly detailed studies of the effects of sampling on some flow metrics, such as packet distribution. In their greater part, whole flow populations are considered instead of focusing on specific sets of a population (*e.g.*, elephant flows). We believe that by focusing on a specific set, more insightful results can be achieved on the observed set. To the best of our knowledge, few of them are known from the literature about the effects of sampling on the specific set of elephant flows while observing the flow metrics: number of bytes, number of packets, and duration.

Moreover, most of these works miss one important dimension of analysis. In their majority, they present isolated studies, in which simulation tools or controlled test beds are used to reproduce a network being measured using packet sampling.

We believe that these methods, even if suitable for their analysis, may not consider unexpected external factors that may occur in real networks, such as background traffic. Observed metrics can behave differently when competing with other traffic [159]. Even if considered, background traffic may not be 100% modeled when using simulation tools or controlled test beds.

We believe that background traffic can play an important role by significantly influencing the way packets are sampled, and therefore distorting the collected information about our considered flow metrics. Based on that, we believe that analyzing the effects of sampling using data collected from real networks would provide more realistic and relevant conclusions. Within this context, the specific research question that motivates our investigation is: *Can information on elephant flows, obtained from real networks, be considered accurate when sampling is used?* Answering this question is relevant because it can better help to understand the effects of sampling on our metrics of interest as well as warn us of any misleading information sampling can result to our autonomic decision process.

3.5.2 Our analysis approach

In order to answer our research question, we have collected and analyzed flows from three different real networks that employ different systematic sampling ratios: the UT network, where no sampling is in fact employed (1:1); SURFnet (SN), where a sampling ratio of 1 out of 100 packets is used (1:100); and finally on GÉANT (G), which employs a sampling ratio of 1 out of 1000 (1:1000).

From all collected flows, only those that transited in the three considered networks, and had therefore the chance of being sampled in these networks, were considered for our analysis. Another filter was used over the considered data in order to select only elephant flows. Figure 3.10 depicts our approach.

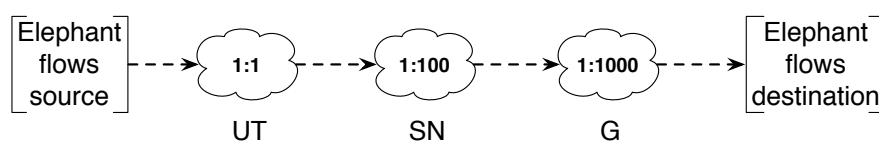


Figure 3.10: Our approach to observe the effects of sampling on our metrics of interest.

In order to collect NetFlow data from the considered networks, NetFlow-enabled routers in the UT, SURFnet, and GÉANT networks have been configured to export NetFlow records to flow collectors hosted in the UT network. When we collected NetFlow traces from these networks, UT and GÉANT routers used NetFlow version

5 whereas SURFnet routers used NetFlow version 9. Traces from the three networks have been collected over a period of two days (August 1, 2007 - August 2, 2007).

3.5.3 Combining flow records

Since NetFlow reports flow metrics in parts (Subsection 3.4.4), one needs to combine the NetFlow records in order to compute the original flow duration, number of packets, and number of bytes. In order to combine NetFlow records, there is a need to determine the gap that separates two consecutive NetFlow records of the same flow. Two consecutive flow records are grouped into the same flow if the gap between the end of the previous NetFlow record and the start of the next NetFlow record is smaller or equal to *gap timeout*. Otherwise the next NetFlow record starts a new flow (Figure 3.11).

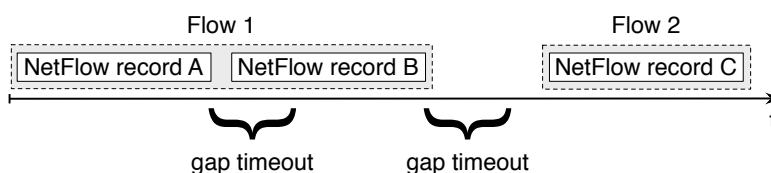


Figure 3.11: Example of combining NetFlow records into flows.

We have deliberately chosen *gap timeout* as 30 seconds, which is a common value for the TCP `TIME-WAIT`² state. We then decided that all NetFlow records of the same flow which gap was smaller or equal to 30 seconds are grouped into the same flow. All our analyses were made then over the combined flows rather than using the original NetFlow records. The next subsection shows the results of our analysis considering elephant flows.

3.5.4 Trace Analysis

This subsection presents the results of our analysis considering the number of bytes, number of packets, and duration. These metrics are observed by comparing their expected theoretical values after re-normalization with their obtained values in practice when sampling is employed. The traces collected from UT were considered as basis for this comparison. Table 3.6 shows how the expected values for each parameter are calculated for each network.

²When a TCP connection is closed, the socket pair associated with the connection is placed into a state known as `TIME-WAIT`, which prevents other connections from using the same 5-tuple (source/destination IP addresses, source/destination TCP ports, and protocol) for a period of time.

Network	# Bytes	# Packets	Duration
UT	B	P	D
SURFnet	B/100	P/100	D
GÉANT	B/1000	P/1000	D

Table 3.6: Expected values in theory for the SURFnet and GÉANT networks.

Bytes (B) and packets (P) observed in UT are expected to be reported in SURFnet and GÉANT as the original B and P divided by the sampling ratio employed, *i.e.*, 100 and 1000, respectively. In the case of duration (D), regardless the number of octets and packets seen in SURFnet and GÉANT, the same duration D is expected in both networks because the sampling ratio should not affect it. For each metric, an average deviation is computed by calculating how distant the obtained values in practice deviate from their expected value in theory.

Number of bytes

The first metric considered in our analysis is the number of bytes. We compare this metric by its number expected in theory with the number obtained in practice.

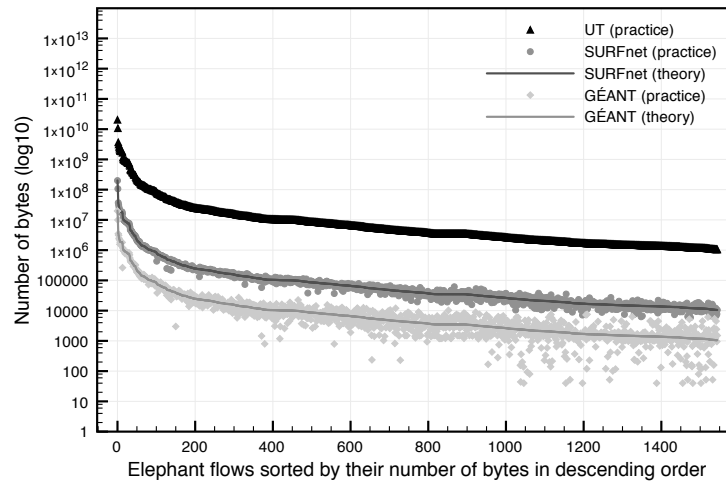


Figure 3.12: Byte distribution in theory and in practice per organizational network.

Figure 3.12 shows the expected value in theory for SURFnet and GÉANT bytes and the values that have been obtained in practice. The x-axis shows the elephant flows ordered by their number of bytes in descending order, being therefore elephant flows with bigger sizes leftmost and smaller ones rightmost. The y-axis represents their number of bytes in logarithmic scale.

As observed in Figure 3.12, the bigger the elephant flows are, the closer their obtained values in practice are to their expected values in theory. In contrast, as the size of elephant flows decreases, more imprecise are their reported values. Generally speaking, the number of bytes of elephant flows measured via flow sampling is close to their expected theoretical value. The reason is that elephant flows generate a large amount of packets, which increases the chances of their packets to be sampled and, as a result, increases the chances of flow bytes being accurately reported.

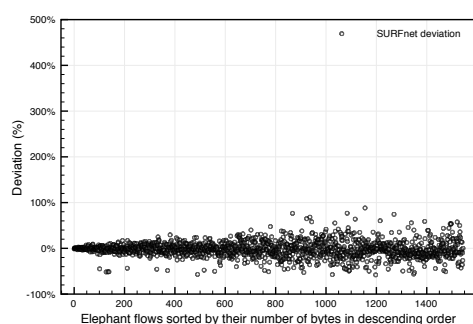


Figure 3.13: Deviation percentage of SURFnet elephant flows when compared with its theoretical value.

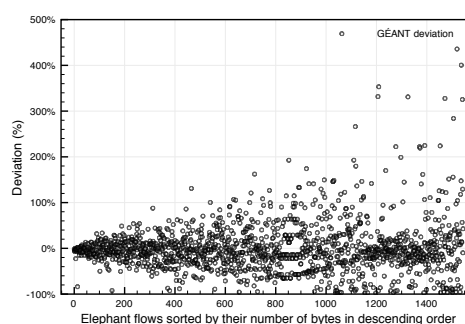


Figure 3.14: Deviation percentage of GÉANT elephant flows when compared with its theoretical value.

Figures 3.13 and 3.14 show the deviation percentage of the SURFnet and GÉANT flows when compared to their theoretical value. The average deviation found in our analysis was -0.55% in SURFnet and -5.47% in GÉANT. The average deviation observed in the GÉANT flows is 10 times bigger than the SURFnet one. We assume that the reason for that is due to the sampling ratio used in these two networks: GÉANT has a sampling ratio 10 times bigger than SURFnet.

Number of packets

The number of packets were also compared by observing their expected number in theory with the number obtained in practice.

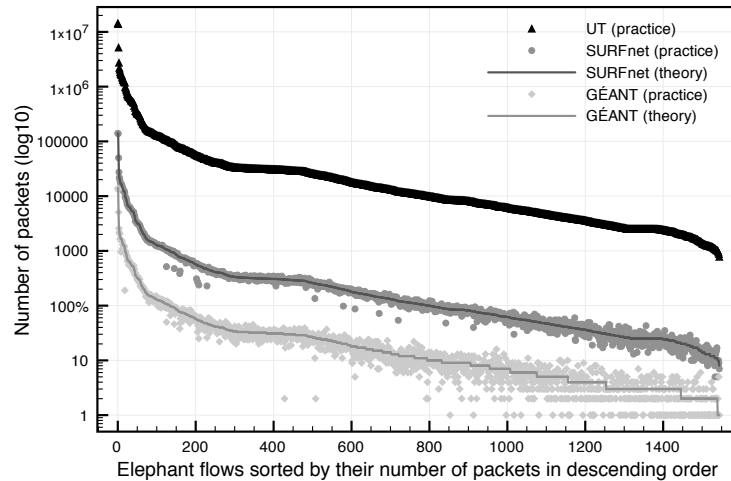


Figure 3.15: Packet distribution in theory and in practice per organizational network.

Figure 3.15 shows the number of packets measured at the UT network, the expected values in theory for SURFnet and GÉANT packets, and their values obtained in practice. The x-axis shows the flows ordered by the number of packets in descending order, being therefore the flows with the bigger amount of packets leftmost and the smaller ones rightmost. The y-axis (logarithmic scale) shows the number of packets. Moreover, Figure 3.15 shows that the bigger the amount of packets of elephant flows are, the closer the obtained values for the elephant flows packets are to their expected value in theory (as also observed in the previous metric). The reason for this similarity follows the same previous explanation: the larger the amount of packets generated, the higher the chances are that these packets are sampled.

Figures 3.16 and 3.17 show the deviation percentage of the SURFnet and GÉANT flows when comparing with their expected value in theory. The average deviation found in our analysis is -0.62% in SURFnet, while in GÉANT it is -6.20% . The same assumption previously stated when explaining the average deviation for the number of bytes also applies here.

Partial conclusion is that both metrics present a certain deviation from the reality, but their deviation is relatively small in terms of percentage. SURFnet, which has a sampling ratio smaller than GÉANT is more precise, whereas GÉANT presents more deviation from the expected value.

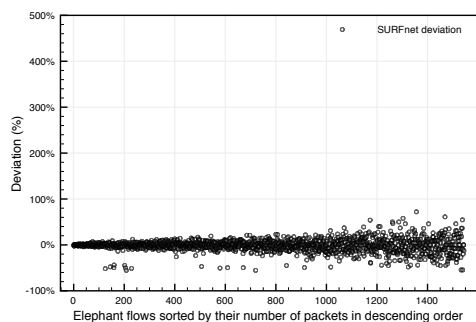


Figure 3.16: Deviation percentage of SURFnet elephant flows when compared with its theoretical value.

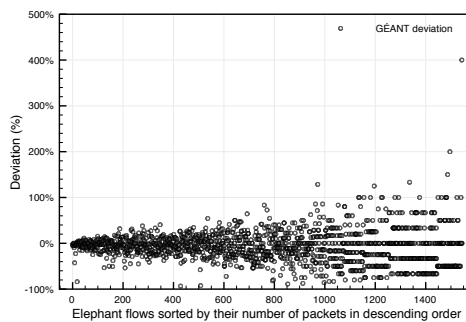


Figure 3.17: Deviation percentage of GÉANT elephant flows when compared with its theoretical value.

Flow duration

We present here the duration of the elephant flows as observed in the three considered networks. The flow duration is shown as total duration, which is calculated from the timestamp of the last seen packet belonging to the flow minus the timestamp of the first packet seen.

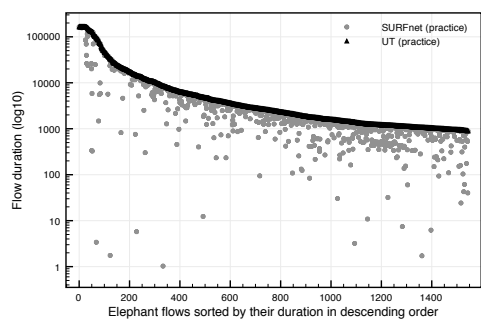


Figure 3.18: Flow duration distribution observed at the SURFnet network.

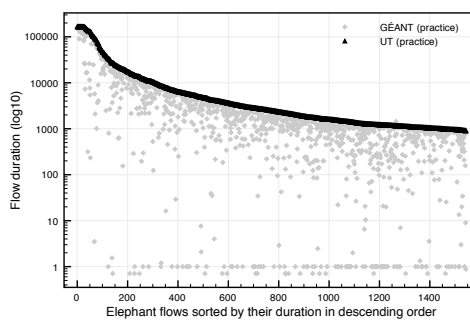


Figure 3.19: Flow duration distribution observed at the GÉANT network.

Figures 3.18 and 3.19 show the obtained duration (in seconds) for SURFnet and GÉANT networks, respectively. In theory, they should have the same duration as UT, which, however, is not the case in practice. The x-axis shows the flows ordered by their duration in descending order, being therefore the flows with the longer

duration leftmost and the shorter ones rightmost. The y-axis (logarithmic scale) shows the duration of every single elephant flow.

SURFnet reports, on average, -15.15% of difference to the UT duration, whereas in GÉANT the difference is on average -31.15%. These percentages are computed by averaging the deviation percentage of the reported flow durations in the SURFnet and GÉANT networks when compared with their real duration (*i.e.*, the flow duration reported by UT). The conclusion is that the larger the sampling ratio, the larger the number of misreported flows (in terms of duration) will be. Moreover, we can also conclude that flow duration is more sensitive to missing packets than the packets and octets metrics. We believe that this sensitivity may occur when a flow does not keep a constant and high packet rate. Oscillations in the flow rate may decrease therefore the chances of packets being sampled (Figure 3.20) and, as a result, the flow duration is reported in short and small parts rather than in continuous ones. This situation gets worse when high sampling ratios (*e.g.*, 1:1000) are employed, as observed in Figure 3.19.

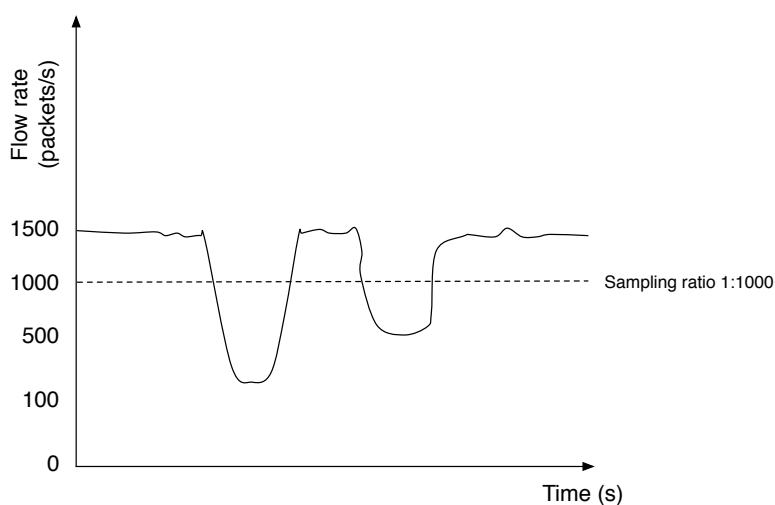


Figure 3.20: Example of an inconstant flow rate vs. a high sampling ratio.

Partial conclusion concerning flow duration is that it brings some risk for our self-management approach. The reason for this concern comes from the fact that if the flow duration is inaccurate, so are the calculations of flow Bps (number of bytes/duration) and flow Pps (number of packets/duration). Alternatives shall therefore be considered in order to overcome this inaccuracy.

Flows with duration equal to zero

Our study also generated some interesting side effects, such as the observation that there was a large amount of flows reported with duration equal to zero in the three considered networks (Figure 3.21).

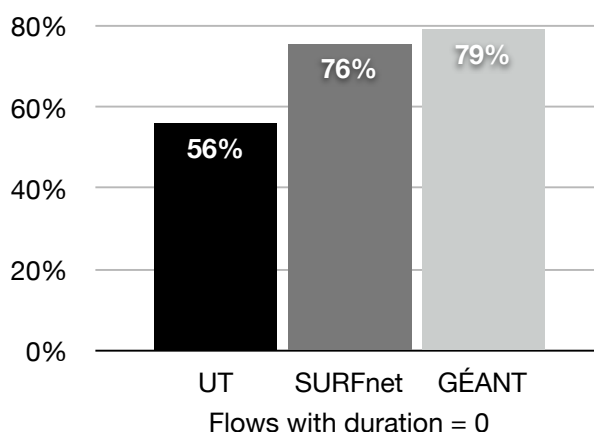


Figure 3.21: Percentage of total flows reported with duration equal to zero.

UT does not use sampling and it had 56% of the flows reported with duration equal to zero. Meanwhile, SURFnet (sampling ratio 1:100) and GÉANT (sampling ratio 1:1000) had 76% and 79% of the flows reported with duration equal to zero, respectively. The two main reasons we see for this observation are:

- *The existence of applications that regularly send control messages:* we have found that there is a group of applications that make NetFlow to generate flows with a single packet (*i.e.*, flows with duration zero). We have seen applications such as the well-known Domain Name System (DNS) and Network Time Protocol (NTP), but newer ones like Gnutella. These applications regularly send control messages either for synchronization (*e.g.*, NTP) or for cache updates (*e.g.*, DNS TTL caching). If this interval is bigger than the NetFlow inactive timeout, this will result in thousands of flows with duration equal to zero.
- *The usage of sampling:* sampling increases the probability of not inspecting a packet belonging to an existing flow in cache. As a result of that, the number of

flows with a single packet is larger in those networks where higher sampling ratios are used.

3.5.5 Concluding remarks

The research question raised in this section was *Can information on elephant flows, obtained from real networks, be considered accurate when sampling is used?* Our conclusion is that the observed metrics number of bytes and packets are reliably reported when sampling is used, but flow duration is considerably affected. Our analyses show that, on average, the number of bytes and packets were respectively reported as -0.55% and -0.62% less than their real value when a sampling factor of 1:100 (SURFnet) is used. Similarly, when a sampling factor of 1:1000 (GÉANT) is used, the average bytes and packets deviation from the real value was -5.47% and -6.20%, respectively. This leads us to conclude that the bigger the sampling ratio is, the less precise the number of bytes and packets will be reported.

Not only bytes and packets were affected, but also flow duration. Actually, flow duration showed in our analysis to be very sensitive to sampling. Several analyzed samples showed us a reported duration smaller than in reality. This imprecision in reporting the expected duration increases when the sampling ratio increases. Our analyses showed that on average the flow duration had a deviation of -15.15% and -31.15% from the expected value (*i.e.*, UT flow duration), when 1:100 and 1:1000 sampling ratios were used. The reason for that comes from the fact that duration is highly affected when the flow packet rate is not constant and mostly below the sampling ratio. Since the chances of a packet being sampled decrease in that case, it increases the probability that packets will not be tallied to count for the flow duration. However, the opposite of that shows that the bigger the number of packets belonging to a flow is, more reliable is its duration, because the bigger are the chances the packet will be sampled and, therefore, tallied.

In order to avoid the effects of sampling on flow parameters found on our research, Cisco experts informally recommended us the use of a flexible sampling approach through the usage of Flexible NetFlow [32]. Flexible NetFlow has the key ability to focus and monitor specific network behavior. Instead of performing 1 out of n sampling with the same n for all traffic, Cisco Flexible Netflow enables a network operator to define flow priorities on the router. This results in the segregation of traffic into classes and specification of different sampling ratios across classes. For instance, NetFlow-based routers running Flexible NetFlow can be instructed to focus on providing full information (*i.e.*, no sampling employed) about elephant flows while sampling the other smaller flows. The information provided about the elephant flows would therefore not suffer the effects of sampling, as just presented

in this section. Since Flexible NetFlow can provide more reliable results about flows, it is therefore the most suitable technique to be employed in our self-management approach.

3.6 Summary

THIS chapter presented techniques for the monitoring of computer networks that are relevant for our self-management approach. It also presented and discussed network parameters that are potentially suitable for our autonomic decision process when deciding the destiny of IP flows.

We started by presenting a study about potential network parameters to be considered by our autonomic decision process. As presented in Section 3.1, two main groups can be pointed out: 1) parameters to identify flows, which we called *flow identifier parameters*; and 2) parameters to characterize the behavior of flows, called *flow behavior parameters*.

As presented in Section 3.2, flows can be identified by different flow definitions. The definition of flows can go from highly granular definitions to slightly granular definitions. The level of restrictiveness to group packets into flows follows the same pattern, *i.e.*, highly granular definitions for a flow are more restrictive to group packets into flows, whereas slightly granular definitions group more packets into flows. As a consequence of this level of restrictiveness, flows identified with high level of granularity tend to be smaller in size than flows identified with lower levels of granularity. It is worth mentioning although that highly granular flow definitions mix different kinds of traffic, presenting therefore a considerable variance in flow size. On the other hand, highly granular definitions are more restrictive, resulting in similar kinds of traffic with smaller variance in flow size.

Following the analysis of network parameters, Section 3.3 discussed what behavior parameters are more relevant to predict flow volume. Through our statistical analysis, the parameters Bps, duration, and optionally Pps have more influence in the prediction of flow volume. These metrics have more impact on the flow volume than others, being therefore highly relevant for our autonomic decision process to take decisions on moving flows between network levels.

With regard to monitoring techniques, Section 3.4 presented three network monitoring techniques: *packet-based* (Subsection 3.4.2), *SNMP-based* (Subsection 3.4.3), and *flow-based* (Subsection 3.4.4). Packet-based techniques present the finest level of detail about network traffic, but it is the one that demands more processing and storage space. On the other hand, SNMP-based techniques present a low volume of measurement data, but with the coarsest level of detail. As an in-between technique,

flow based-techniques generates a medium amount of measurement data while presenting a considerable amount of network traffic details.

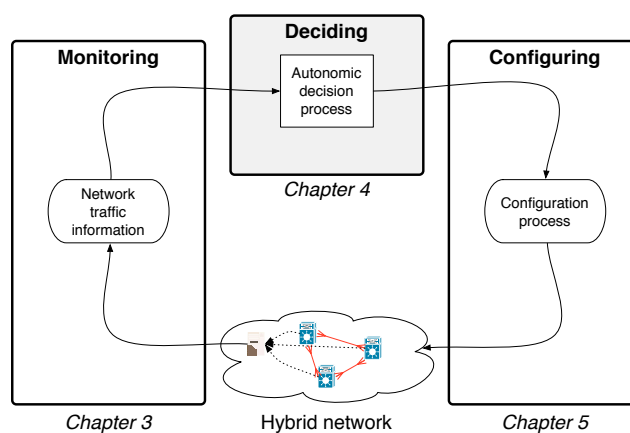
Since our self-management approach focuses on high-speed optical networks, the volume of data is considerable high in those networks, being packet-based techniques an unfeasible choice to be used in our approach. SNMP-based techniques employ the pull model, which send information about flows only when a request occurs. This model scales well as long as the amount of measurement data is not high. When that occurs, memory shortage can make SNMP agents to discard flow information, being therefore not completely satisfactory to be used on high-speed networks. Finally, flow-based techniques, employ the push model, which exports flow information automatically without the need to wait for a request. This model avoids memory shortage by exporting flow information rather than discarding it.

Our self-management approach considers therefore the use of flow-based techniques, in which we gave more special attention to the usage of sampling. The use of sampling considerable reduces the amount of processed and collected network data. As presented in Section 3.5, its use results in some side effects on network parameters such as distorted number of bytes, packets, and the total duration of a flow. The latter is the one that suffers the effects of sampling the most. One alternative to overcome these undesirable effects is the use of Flexible NetFlow, which does not perform sampling on some specified group of flows (*e.g.*, elephant flows).

Chapter 4

Making autonomic decisions

In Chapter 3 we presented monitoring techniques to obtain network information from a managed network. Also presented was the set of metrics that has most impact on predicting the volume of flows, being therefore relevant for our autonomic decision process. In this chapter, we focus on how autonomic decisions can be taken to move large flows to/from the optical level. For that, we first define the main objective of our decision process. Then, we perform a literature study in order to check whether there is any suitable approach that can contribute to reach such objective. Due to the lack of any, we propose a new one, which is explained in detail and validated at the end of this chapter.



The organization of the present chapter is as follows:

- *Section 4.1 presents what the main objective of our autonomic decision process is.*
- *Section 4.2 discusses the main characteristics and algorithms used in caching theory. The reason for this discussion is the fact that our decision process has some similarity to general caching strategies.*

- Section 4.3 introduces our autonomic decision policy to automatically move flows between the optical and IP levels as well as the algorithm used in our autonomic decision process.
- Section 4.4 presents some assumptions behind our decision policy.
- Section 4.5 presents our validation to the proposed decision policy. This section shows how much the aimed objective has been reached when autonomically moving flows to the optical level.
- Section 4.6 shows an evaluation of which grooming (multiplexing) strategy is the most appropriate to offload more data from the IP level over lightpaths at the optical level.
- Section 4.7 finalizes this chapter by presenting some concluding remarks.

4.1 Autonomic decision objective

ADVANCES in data transmission in hybrid networks have enabled data forwarding decisions to be taken at multiple levels in the protocol stack (*e.g.*, at network and optical levels). With this new network feature, elephant flows can be moved from the network level and switched completely at the optical level over lightpaths. Moving elephant flows from the IP level to the optical level presents the following advantages: (a) elephant flows over lightpaths experience faster and more reliable transmissions with optical switching than with traditional IP routing; (b) remaining smaller flows at the IP level also experience better services because the network layer is less congested after offloading elephant flows. Last, (c) it is cheaper to send traffic at the optical level than at the IP level [38].

In order to achieve the aforementioned advantages, a proper strategy for flow placement in the IP and optical levels have to be adopted. Several alternatives can be employed. One could suggest to offload only flows belonging to specialized applications, in which decisions to offload flows would be taken given certain port numbers. Even though this may partially work, it could happen that there is a non-specialized application generating more traffic than a specialized one, but the former would not be moved because it is not considered specialized (because it does not use a separate port number). Another alternative would be to prioritize flows based on the value of their Type of Service (ToS) field. This alternative is commonly used in networks that employ Differentiated Services (DiffServ) [170] to regulate quality of service levels. In such networks, some flows are chosen over others depending on the value set on their ToS field. This alternative seems, however, to be not considered in practice very much. Studies [91] [60] show that approximately

90% of the IP traffic crossing the Internet do not have the ToS field set, which indicates its lack of usability in practice. One could also suggest to prioritize only flows with TCP as the transport protocol, since TCP is the major transport protocol in use in most IP networks, representing approximately 90% of all traffic across the public Internet currently [49] [87]. This alternative has the highest chance of selecting a large flow when prioritizing only TCP over other transport protocols, but has the same shortcoming of prioritizing specialized applications. If there is a larger flow using another transport protocol (*e.g.*, UDP), this flow will not be selected due to the protocol restriction.

Our alternative, on the other hand, aims at prioritizing flows by merit (behavior) rather than by characteristics (*i.e.*, port numbers, ToS, and so on). As stated in Subsection 3.3.2, we focus on elephant flows due to their behavior (few in amount, but most of the traffic). Their merit is measured based on the amount of traffic they are expected to generate. We only know although how large an elephant flow really was, once it ends its activity. However, decisions about moving elephant flows to the optical level are made while they are still in progress. A decision cannot be taken too soon because there is a high probability that a flow is going to last short (Subsection 3.3.2). In contrast, a decision cannot be postponed too long either because the elephant flow will be consuming resources at the IP level while the decision is being taken. Moreover, there is a slight chance that the elephant flow may end when the decision is finally made. Our alternative should thus also consider the chances of an elephant flow continue its activity (*i.e.*, its survival probability) while taking decisions. Within this context, our autonomic decision process has as main objective:

“to autonomically move the biggest elephant flows with high survival probability from the IP level to the optical level.”

This objective is the essence of our autonomic decision process when deciding which flows should be moved to/from the optical level. The main decisions taken by our autonomic decision process are whether a flow should 1) be moved from the IP level to the optical level (the act of promoting it), or 2) be moved back to the IP level (the act of demoting it). When a flow is promoted, the optical level is consulted to check whether there are resources (*i.e.*, lightpaths). If so, the flow is moved. If there are no resources available, the decision process must decide which flows at the optical level should be demoted (*i.e.*, moved back to the IP level). Some similarity with the acts of promoting and demoting large flows is found in caching theory, in which a process must decide which cache entry should be removed from a cache upon inserting a new one. This similarity is addressed in the next section.

4.2 Similarities with cache management

THE main objective of cache management is to decrease the latency in the access of certain information [66]. For example, a Web page has certain latency to be accessed from a Web server on the Internet. Without caching, any new access to the same Web page will basically have the same latency. However, if caching is employed, the Web page is locally stored and its subsequent accesses are faster. Caching is currently applied in several areas, such as memory caching [64] [141], Web caching [162] [122], and disk caching [148]. Our autonomic decision process can be related to cache management theory: the optical level could be considered the cache while flows could be considered as the cache entries.

Cache management is performed by caching algorithms. Caching algorithms strive to leave in cache information that will be more likely used in the near future (*i.e.*, that will more likely have a cache hit). In acting so, the most efficient caching algorithm would be the one that always select the information that will be needed for the shortest time in the future (*i.e.*, the algorithm has a high hit rate). To reach the best hit rate as possible, caching algorithms use different strategies. Some of the most known algorithms and their respective strategies will be presented below:

Belady's algorithm: states that certain information should be replaced if this information will take place farthest in the future. For example, if a certain cache is full and there is information that will not be needed for the next 10 seconds, but there is information that is required within the next second, the former will be swapped over the latter. Even if this algorithm perfectly works in theory, it cannot be used in practice since it is generally impossible to predict how far in the future information will be needed. Belady's algorithm [14] is thus used to evaluate caching algorithms strategies since it provides the theoretical optimal result [167].

Other caching algorithms try to overcome this lack of information about the future based on present (or recent past) information. One of the most well-known algorithms is *First In First Out (FIFO)*. FIFO is a simple caching algorithm, which demands the least complexity to be implemented. It is based on the time an item has spent in cache. The first item to be added to the cache will also be the first item to be removed. Since this algorithm does not employ any further strategy to keep likely future items in cache, it is barely used, being mostly considered for historical reasons only.

More refined caching algorithms have been developed by taking into account some specialized strategies when deciding for replacing items in cache. These strategies can be classified as frequency-based, recency-based, and size-based strategies

[27] [68]. Some examples of caching algorithms that comply with these strategies are described as follows:

Least Recently Used (LRU): LRU discards the cache item that has not been used for the longest time, which characterizes LRU as an aging algorithm. The main problem with this algorithm is its arduous cache management in order to keep the items organized. Every time a cache item is used, the age of all other cache items changes. When an item in the cache is accessed, it will be moved from its place to the end of a list. When a new item in cache is accessed, it will be put in the end of the list and the oldest item is taken out of the cache.

Most Recently Used (MRU): The MRU algorithm has been developed by Shaul *et al.* [37]. MRU discards, in contrast to LRU, the most recently used items first. MRU algorithms are mostly useful in situations where the older an item is, the more likely it is to be accessed.

Largest File First (LFF): this algorithm removes first the largest file in cache. This algorithm has been developed by Williams *et al.* [1] and it is applied in Web proxy caches.

The afore presented algorithms follow, each one, exclusively one strategy, but hybrid caching algorithms also exist, in which caching strategies are combined.

Segmented LRU (SLRU): This algorithm has been developed by Karedla *et al.* [83] to improve disk caching performance through the use of recency-based and frequency-based strategies. The cache is divided in protected and probationary segments. When a hit occurs in a probationary segment, the hit cache item is moved to the protected segment. If the latter does not have enough space, then the least valuable object is moved out, and inserted as the most valuable object into the probationary segment. If there is not enough space in the probationary segment, then the least valuable object is removed from it.

Greedy Dual-Size (GDS): This algorithm has been developed by Cao and Irani [21]. GDS assigns to each cache entry a value of benefit. Cache entries are removed from the cache in the order of the smallest value of benefit to the biggest one. When an entry is removed, its value is subtracted from all other entries values in the cache. On the other hand, if an entry is hit, its value of benefit is increased to its original value.

Each one of these cache algorithms is more suitable for a specific situation. For example, for random access patterns and repeated scans over large data sets, MRU algorithm performs better than LRU due to its tendency to retain older data. In spite

of that, a common characteristic of the investigated caching algorithms is that they focus on reducing latency to obtain certain information. For that, all the investigated caching algorithms aim at increasing hit rate.

It is well-known that some properties of computer systems and networks present certain frequency distribution, in which a particular set of objects (*e.g.*, network flows) tends to occur more frequently than another set. As a matter of example of a frequency distribution, Figure 4.1 shows the flow size distribution measured at the UT network over the day Sep 18, 2008. It is seen that small flows occur more frequently than big flows.

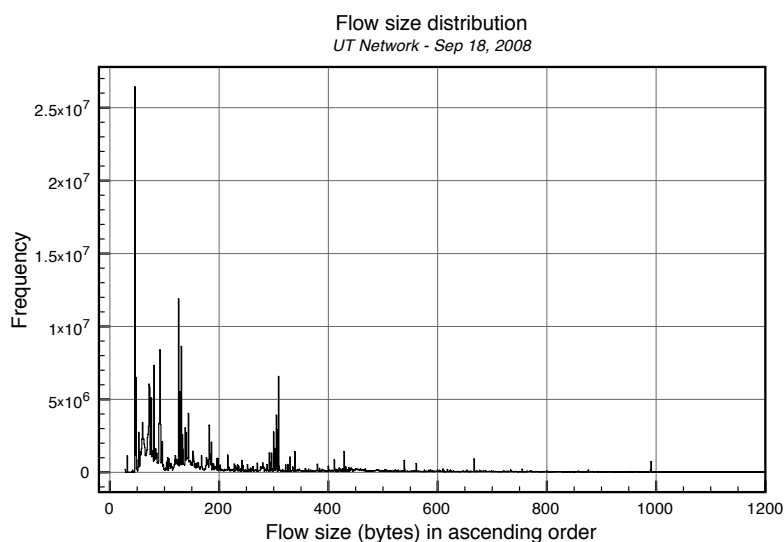


Figure 4.1: Flow size distribution at the UT network.

Caching algorithms use this frequency distribution property in order to improve hit rate. Caching algorithms focus on small entries rather than on big ones. Small entries occur more often than big ones, which increases the hit rate. Another reason for caching algorithms to choose small entries over big ones is that large entries take more space in cache, leaving out several other smaller entries. By leaving out smaller entries, the hit rate decreases, which results in an unsatisfactory cache management.

On the contrary, our autonomic decision process focuses on selecting big flows (large entries) over smaller flows to offload them to the optical level. Big flows are few in number, but represent most of the network traffic. Focusing therefore on big flows is more suitable for our decision process in order to satisfy our autonomic decision objective. As a result of this difference in focus, none of the aforementioned

caching algorithms are applicable for our autonomic decision process. **Caching algorithms focus on small entries (the most often), whereas our autonomic decision process focuses on big ones (the least often).** In face of that, the developing of a new algorithm is needed in our approach, which it will be shown in the next section.

4.3 The autonomic decision process

THIS section will present details concerning the decision of moving flows to/from the optical level. We first introduce the modules involved in the autonomic decision process, followed by our method to weigh flows. Next, our decision policy is introduced, and, finally, the section ends with the presentation of our decision algorithm.

4.3.1 The autonomic decision process modules

Our autonomic decision process is composed by three modules as seen in Figure 4.2.

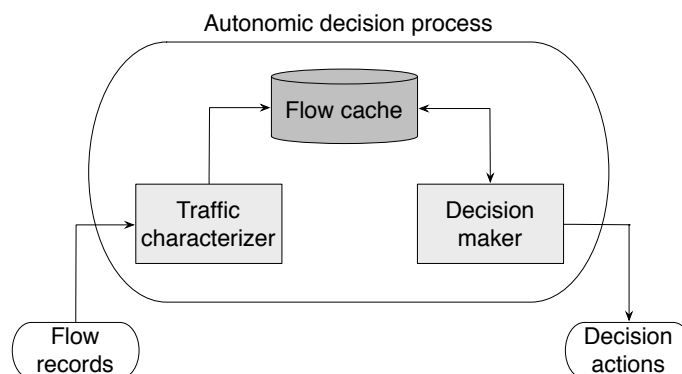
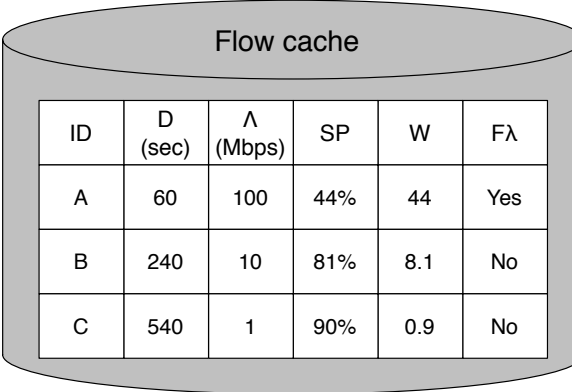


Figure 4.2: The autonomic decision process modules.

The *traffic characterizer* module is in charge of receiving flow information through flow records and characterizing it. The characterization process consists of maintaining a cache, called *flow cache*, with information and statistics on flows. This cache is then used by the *decision maker* module that, based on a decision policy (Subsection 4.3.3), decides the destiny of a flow. That is, it decides whether a flow is promoted to the optical level or demoted back to the optical level.

The *flow cache* contains the following information: flow identification (ID), flow duration thus far (D), current flow throughput (Λ), flow survival probability (SP),



Flow cache

ID	D (sec)	Λ (Mbps)	SP	W	F λ
A	60	100	44%	44	Yes
B	240	10	81%	8.1	No
C	540	1	90%	0.9	No

Figure 4.3: The flow information cache.

flow weight (W), and a flag field informing whether a flow is at the optical level or not ($F\lambda$). Figure 4.3 shows how the *flow cache* looks like.

Flow entries are kept in cache in descend order by their weight. A flow entry is removed from the cache if one of the following conditions happens:

1. **The cache is full:** In such a situation, the flow with the lowest weight is removed first.
2. **A flow ends its activity:** This is characterized by the TCP flags (FIN or RST) indicating the flow termination.
3. **A flow is inactive:** It characterizes as a gap between two consecutive flow records (Subsection 3.5.3).

Since most of large flows have a constant throughput (Subsection 4.4.1), we expect that these flows do not present any gap or, if so, a short one due to the exporting process of flow records (*e.g.*, NetFlow cache management). In order to see how short the gap of large flows is expected to be, we observed 10.000 large flows over an entire day at the UT network. The flow gap statistics presented by these flows are described in Table 4.1.

Table 4.1 shows that the observed flows had a considerable mean gap of 17 seconds. It is seen that there was a considerable gap range between 0 and 14.061 seconds, which implies a substantial variance. However, most of the analyzed flows (75%) had a gap smaller than 4,44 seconds. In order to overcome the misleading mean gap (due to the huge variance found), we believe the median value of 0,25 seconds is a more realistic value for a flow gap.

Flows	10.000
Mean	17 sec
Median	0,25 sec
Minimum	0 sec
Maximum	14.061 sec
Percentiles (25%)	0,05 sec
Percentiles (50%)	0,25 sec
Percentiles (75%)	4,44 sec

Table 4.1: Flow gap statistics (Sep 18, 2008)

4.3.2 About weighting flows

Discussions about flow identification parameters and flow behavior parameters were already presented in Sections 3.2 and 3.3, respectively. On its turn, discussions on flow survivability were presented in Subsection 3.3.2. In this subsection, we give more attention on how the weight of flows is calculated.

In order to make the best prediction as possible about moving flows, our autonomic decision process attributes weights to flows based on their current throughput and their survival probability. The weight of a flow (W_f) is represented as the following mathematical representation:

$$W_f = \Lambda_f \times P(D_f \geq \Delta t + t \mid D_f \geq t) \quad (4.1)$$

where Δt is the active flow timeout used in the monitoring process; D_f is the duration of a flow thus far, being defined by the point in time a flow started (or restarted in the case of flow inactivity) until the moment the flow is evaluated; Λ_f is the current flow throughput, being determined by the ratio between the flow volume and its duration as reported in the latest flow record. Finally, $P(D_f \geq \Delta t + t \mid D_f \geq t)$, is the survival probability (SP) of the evaluated flow. It is based on the flow duration distribution and it is calculated by means of conditional probability (Subsection 3.3.2). It is worth saying that the conditional probability is constantly updated in order to reflect traffic changes. In addition, flow entries in *flow cache* are weighted every time our autonomic decision process is fed with network information (flow records).

As a matter of example, let us imagine that flow A has an elapsed duration of 180 seconds and the latest flow record of this flow reported a number of bits of 71.000.000 and a duration of 45 seconds. In addition, $\Delta t = 60 \text{ sec}$. The weight of this flow is calculated as follows:

$$W_{flow\ A} = \Lambda_{flow\ A} \times P(D_{flow\ A} \geq \Delta t + t \mid D_{flow\ A} \geq t)$$

$$\begin{aligned}
W_{flow\ A} &= 71.000.000/45 \times P(D_{flow\ A} \geq 60 + 180 \mid D_{flow\ A} \geq 180) \\
W_{flow\ A} &= 1.5\ Mbps \times P(D_{flow\ A} \geq 240 \mid D_{flow\ A} \geq 180) \\
W_{flow\ A} &= 1.5\ Mbps \times 0.83 = 1.24
\end{aligned}$$

The weight of a flow considers some assumptions, which the proof of correctness will be shown later in this chapter (Section 4.4). When attributing weights to flows, we assume that large flows are expected to have a constant throughput over time, not presenting a considerable variability in the transmission rate. Moreover, we assume that there is a decreasing flow termination rate over time, assuming thus a heavy-tailed distribution.

4.3.3 The decision policy

A decision policy is used by our autonomous decision process to decide the destiny of a flow, *i.e.*, its promotion to the optical level or its demotion back to the IP level. In a perfect scenario, the decision policy should always keep at the optical level flows that are more likely to generate most of the traffic in the future (fulfilling therefore the objective of our self-management approach). Using caching theory as an analogy, our decision process aims at keeping at the optical level (cache) the flows (cache information) with the highest probability of generating a lot of data in the future. However, similar to Belady's algorithm (Section 4.2), no information about the future is known in advance. The decision policy should, therefore, use information currently available in order to predict a possible outcome in the future. In order to make the best prediction as possible, our decision policy aims at:

“prioritizing flows with higher weights over flows with smaller weights.”

The reason for that comes from the fact that flows with a high throughput and a high survival probability (high weight) are expected to generate more traffic than flows with low throughput or low survivability (small weight). For example, if there are two flows with the same throughput and one of them has a higher chance of continuing (*i.e.*, a high survival probability), this flow would be chosen over the one with smallest survival probability.

4.3.4 Our autonomous decision algorithm

Given the decision policy and our method to weigh flows, we show now how the *decision maker* module decides the destiny of a flow. As previously mentioned, the *decision maker* takes decisions every time *flow cache* is updated, that is, every time the *traffic characterizer* module receives flow information and updates *flow cache*.

When an update occurs, the *decision maker* performs different actions given the availability of lightpaths at the optical level. **If the number of free lightpaths ($f\lambda$) is bigger than zero**, then the *decision maker* module promotes the $f\lambda$ flows at the IP level with the highest weight and change their flags ($F\lambda$) into "Yes", informing thus that from now on, the IP flows are at the optical level.

Then, when there is no more free lightpaths at the optical level, the *decision maker* module seeks for the flow with the lowest weight at the optical level. It then compares this flow with the flow at the IP level with the highest weight. If the flow at the optical level has a weight that is equal or higher than the one at the IP level, then the *decision maker* does nothing. If the weight is smaller, then the *decision maker* demotes the flow at the optical level back to the IP level and change its flag to "No". Following that, it promotes the flow at the IP level to the optical level and change its flag to "Yes". It continues performing the actions of promoting and demoting flows until there are no flows at the IP level with a higher weight than the flow with the lowest weight at the optical level. The behavior of our *decision maker* module is depicted in Algorithm 1.

Algorithm 1 *The autonomic decision process algorithm*

Require: $f\lambda$ = Total number of free lightpaths

```

1: if flow cache is updated then
2:   if  $f\lambda > 0$  then
3:     repeat
4:       Promote( $ID_{Highest\ weight}$ )
5:     until  $f\lambda = 0$ 
6:   end if
7:   if  $f\lambda = 0$  then
8:     repeat
9:        $(ID_{Max}, W_{Max}) \leftarrow seek\_max(W_{IP})$  (Highest flow at the IP level)
10:       $(ID_{Min}, W_{Min}) \leftarrow seek\_min(W_{\lambda})$  (Lowest flow at the optical level)
11:      if  $W_{Max} > W_{Min}$  then
12:        Demote( $ID_{Min}$ )
13:        Promote( $ID_{Max}$ )
14:      end if
15:      until  $W_{Max} > W_{Min}$ 
16:   end if
17: end if

```

Once network information is fed and cache is updated (step 1), our algorithm observes the number of lightpaths available at the optical level. If there are available

lightpaths at the optical level (step 2), the flows at the IP level with the highest weight are moved (promoted) from the IP level to the optical level (step 4) until there is no more lightpaths available (step 5). If there are no available lightpaths at the optical level, the algorithm selects the remaining highest weighted flows at the IP level (step 9) and also the lowest weighted flows at the optical level (step 10). If there are flows at the IP level with bigger weight than flows at the optical level, our algorithm will demote the lowest ranked flow at the optical level (step 12) and promote the highest ranked flow at the IP level (step 13). The algorithm stops when there are no flows at the IP level which weight is bigger than any flow transmitted at the optical level (step 15). Two examples will be given for the sake of a better understanding about our algorithm.

Example 1

In this first example, let us imagine that there is only one lightpath available at the optical level ($f\lambda = 1$) and there are three flows at the IP level competing for it: flow A, B, and C. Flow A has an elapsed duration of 540 seconds with a current throughput of 1 Mbps. Flow B has an elapsed duration of 240 seconds with a current throughput of 10 Mbps. Finally, flow C has an elapsed duration of 60 seconds with a current throughput of 100 Mbps. Let us also assume that the active timeout is 60 seconds ($\Delta t = 60 \text{ sec}$). In this case, the survival probability is calculated by taking into account the probability these 3 flows will continue running for at least another minute. According to our conditional probability studies (Table 3.2), flow C would have a survival probability of running one more minute of 67%, followed by 86% of flow B, and 92% of flow A. One could naturally choose the safest option, flow A, since it has more chance of continuing until the next evaluation. However, flow A has the smallest throughput if compared with other concurrent flows. Our algorithm would therefore attribute their weight as follows:

$$\begin{aligned} W_{\text{flow A}} &= 1 \text{ Mbps} \times 0.92 = 0.92 \\ W_{\text{flow B}} &= 10 \text{ Mbps} \times 0.86 = 8.6 \\ W_{\text{flow C}} &= 100 \text{ Mbps} \times 0.67 = 67 \end{aligned}$$

Based on our decision policy, our algorithm picks then flow C, since it has the highest weight. Even though the risk of flow C ending is bigger than A and B, its throughput is considerable high, being therefore worth of taking the risk.

Example 2

As another example, there are no free lightpaths available ($f\lambda = 0$) at the optical level, but there is a flow at the IP level (flow B) with an elapsed duration of 60 sec-

onds and a throughput of 100 Mbps. At the optical level, flow A has the lowest current throughput of 10 Mbps, but with an elapsed duration of 480 seconds. The question that one may arise is “Should flow B be promoted to the optical level or not?” Based on our conditional probability studies, a flow with a duration of 60 seconds has 67% of chance of continuing for one more minute. Whereas a flow with minimum duration of 480 seconds has 92% of chance of surviving. Flow A seems to be the safest choice since it has more chances of continuing sending data than flow B. But flow B carries much more data than flow A at that moment. So, the algorithm calculates their weights:

$$W_{flow\ A} = 10\ Mbps \times 0.92 = 9.2$$

$$W_{flow\ B} = 100\ Mbps \times 0.67 = 67$$

Flow B has a higher weight than flow A. Even though the risk of flow B ending is bigger than A, its expected volume is worth of taking the risk. Flow A would be demoted back to the IP level and flow B would be promoted to the optical level.

This section presented an overview of our algorithm and its policy to classify flows. However, more complex studies were carried out by us in order to come up with the ideas presented in this section. These more detailed studies are discussed in the next section.

4.4 Assumptions made

IN order to properly weigh flows, our decision policy makes some assumptions. These assumptions form the basis for our decision policy to take the best decision as possible to promote or demote a flow. The assumptions are stated as follows:

Assumption 1: *the throughput of large flows is relatively constant during its lifetime.*

Assumption 2: *the longer the flow has lived, the higher its survival probability is.*

Assumption 3: *the shorter the active timeout is, the more accurate the flow survivability prediction is.*

The next subsections discuss the validity of each of these assumptions.

4.4.1 Assumption 1: constant throughput

This assumption has been investigated by Van de Meent [154] and Sadre *et al.* [131]. Van de Meent randomly observed some large flows from a trace and concluded

that even though flows present different flow rates when compared to one another, individually flow rates remain relatively constant.

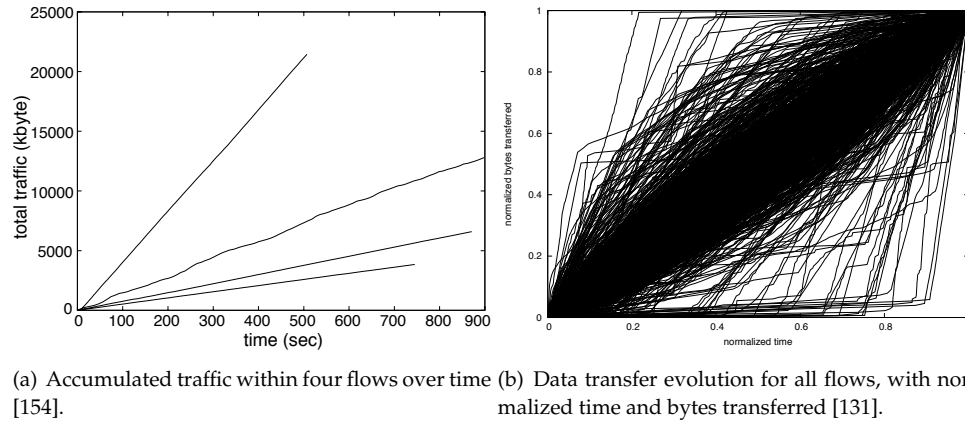


Figure 4.4: Flow throughput over its lifetime.

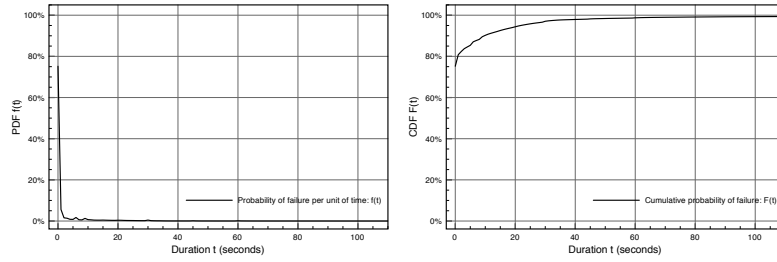
Figure 4.4(a) shows the selection of the four largest flows out of approximately 60.000 flows. The slopes of the lines indicate the rate of the flows, which present a relatively constant rate (given the fairly straight lines). On its turn, Sadre *et al.* observed 2.751 large flows, which accounted for 39.1% (0.83 TB) of the overall UT traffic over the period of one day. He concluded that most of the observed flows presented a constant throughput during their lifetime, even though some of them present some variability (Figure 4.4(b)).

These studies allow us to assume therefore that the flow throughput is expected not to vary too much until the next flow evaluation is performed, *i.e.*, until the next arrival of network information into our autonomic decision process.

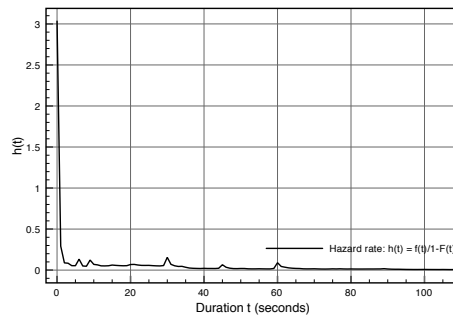
4.4.2 Assumption 2: survival probability

In order to prove assumption number 2, we observed the duration of flows over a whole day in the UT network. Figure 4.5 shows our analysis about the termination of flows, *i.e.*, when flows end their activity. Even though we observe the duration of flows during a whole day (86.400 seconds), we delimited Figures 4.5(a), 4.5(b), and 4.5(c) to 110 seconds only, because there is no significant variability in the flow duration above that time. Figure 4.5(a) shows the flow failure distribution per flow duration t ($f(t)$). It shows that most of the flows (around 75%) end up their activity at their very beginning, *i.e.*, at the time zero ($t = 0$). That means that a flow has the

highest tendency to end when it has just started. Figure 4.5(b) represents the cumulative probability of flow failure ($F(t)$). The figure shows that after approximately 60 seconds of activity, the probability of a flow to end up its activity decreases.



(a) Probability of flow failure per unit of time. (b) Cumulative probability of flow failure.



(c) The flow failure rate.

Figure 4.5: Flow termination analysis.

In order to check if the flows fails at a constant rate (*i.e.*, there is an exponential decay), we also calculated the hazard rate:

$$h(t) = \frac{f(t)}{1 - F(t)} \quad (4.2)$$

Figure 4.5(c) shows that the flow failure rate is not constant. The highest peak rate is at the time zero, when the hazard rate is equal to three ($h(0) = 3$). After that, the failure rate stays between 0.01 and 0.3 with some small peaks at irregular times. **The conclusion learned from this analysis is that the longer a flow has already lived, the higher its survival probability is.**

In another analysis, we checked if there is any distribution model that properly describes the flow survivability distribution. For that, we compared the flow duration distribution calculated from the UT network traces with three classical survivability models: Exponential, Weibull, and Pareto.

Figure 4.6 shows the UT cumulative duration distribution when compared with the considered model distributions. The x -axis is in logarithmic scale. It is worth mentioning that the parameters of the considered distributions were estimated by using Maple [97]. The UT flow duration distribution was used as an input for the parameters estimation.

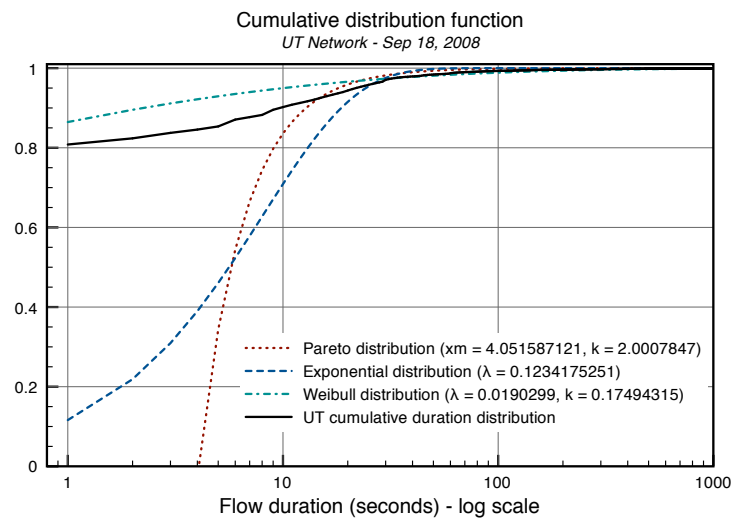


Figure 4.6: Cumulative duration distribution considering Weibull, Exponential, and Pareto distributions.

None of the considered distributions perfectly matches the UT distribution, but the flow survivability can be reasonably modeled by the Weibull model distribution rather than by Pareto and Exponential. That is an interesting finding since Weibull is a common heavy-tailed distribution [171]. As a result of that, it is safe to assume that the frequency of flows with long duration does not decrease at a proportional rate, which would be the case in an exponential distribution. On the contrary, the failure rate decreases over time, which can be confirmed by the estimated Weibull shape parameter $k = 0.17$. A Weibull shape parameter $k < 1$ means a decrease in the failure rate [179]. In the specific case of our study, it can be understood that most of the flows end up early and the failure rate decreases over time as these early

flows finish their activities. **This understanding reinforces our assumption that the longer the duration of a flow is, the bigger the chances are of this flow to survive.**

As an attempt to better fit the UT flow duration distribution, we employed the Expectation-Maximization (EM)-based fitting technique introduced by Sadre *et.al.* [130]. The EM-based fitting technique employs direct fitting of Hyper-Exponential Distributions (HEDs) to a measured data in an iterative fashion. Compared to classical heavy-tailed distributions such as Weibull and Pareto that only have two free parameters to fit the data, the typically used HEDs have more than ten. As a result of that, HEDs distributions tend to fit a set of measurement values better than classical heavy-tailed distributions.

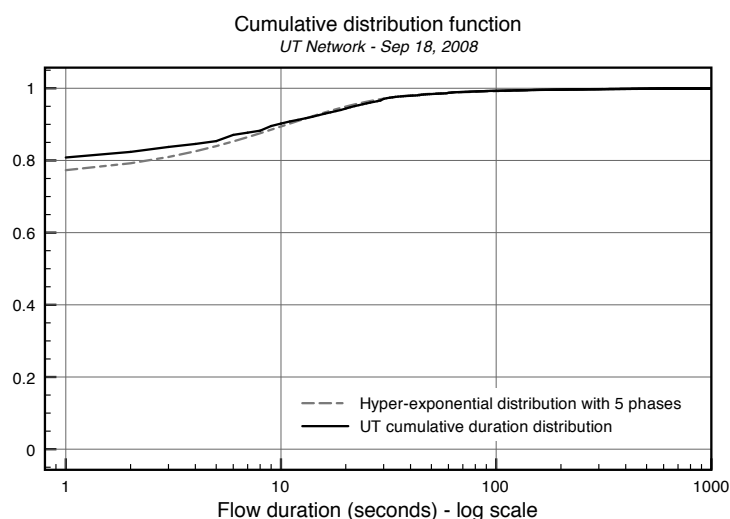


Figure 4.7: Cumulative duration distribution considering a Hyper-exponential distribution with 5 phases.

Figure 4.7 depicts the use of a HED with 5 phases¹ to fit the UT cumulative duration distribution. This figure shows that a HED with 5 phases better fits our data when compared with classical distributions (Figure 4.6).

¹The number of phases refers to the amount of weighted sums of exponential distributions employed in the fitting process.

4.4.3 Assumption 3: accuracy in predicting flow survivability

When predicting the survival probability of flows, we consider the active timeout used in the monitoring process (Δt). The active timeout defines how long the next record of a long living flow will take to be sent to our *decision maker* module. Our *decision maker* considers Δt therefore to calculate the survival probability of flows. We focus this subsection on the influence that different Δt s values have when making predictions about flow survivability.

For that, we carried our study through the use of conditional probability (Subsection 3.3.2). We observe the probability that a flow will continue running until its next flow record is received by the *decision maker* module, given the fact that this flows has already lasted a certain time. For this observation, we use different Δt lengths.

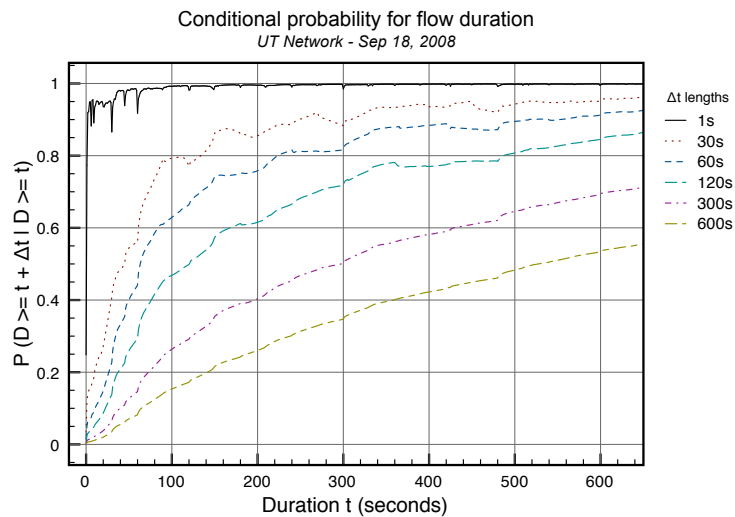


Figure 4.8: Conditional probability of the duration of flows.

Figure 4.8 shows the result of our analysis. The x -axis shows the flow duration at certain time t while the y -axis shows the probability (in percentage scale) of a flow surviving for at least until the next arrival of a new flow record ($\Delta t + t$), given the fact that the flow has lasted at least t . Following the conditional probability representation, it should be read as $P(D_f \geq \Delta t + t \mid D_f \geq t)$. Figure 4.8 shows, for instance, that there is a probability of approximately 80% of a flow lasting another 60 seconds (Δt length = 60s), giving the fact that this flow has already lasted at least 300

seconds. On the other hand, the probability that a flow will last for at least another 300 seconds given the fact that it has already lasted 300 seconds is approximately 50%. **This allows us to say that the flow survivability prediction gets less precise, the bigger the active timeout (Δt) is.**

The main conclusion is that a short Δt is better because it provides better accuracy on the prediction of the flow survivability. However, a short Δt does place a heavy burden to the measurement system due to over-processing and resources consumption. At the other end, a long Δt should be avoided either because it provides both inaccurate (*e.g.*, distorted flow throughput) and unreliable (*e.g.*, survival probability gets imprecise) information about flows. We leave for the network operator to decide what the best trade-off is for his network.

Having presented the complete reasoning (assumptions) behind our decision policy as well as details of our autonomic decision algorithm, the next section shows how we validated our proposal.

4.5 Validation of the decision policy

IN order to validate our autonomic decision process, we perform one simulation with the use of real network data. Network data was collected from the UT network by using NetFlow. At the time of the collection period, UT NetFlow-based routers exported information about long lasting flows every 120 seconds (*i.e.*, $\Delta t = 120 \text{ sec}$). For our validation, we compare our approach with two other approaches to manage lightpaths, being summarized as follows:

1. *Our approach*: takes decisions about moving flows to/from the optical level based on their current information, but no precise knowledge about their future (our weighting method).
2. *Theoretical maximum approach*: is based on the Belady's algorithm concept, in which the best result is theoretically known due to experimentation. By using this approach, information regarding the past and the future of a flow is known at the decision making point. This approach allows us to know what the best theoretical decision would be when moving flows over lightpaths.
3. *Today's approach*: is the approach currently used in most networks. A lightpath is dedicatedly established beforehand and the flows are moved over it afterwards. In most of the cases, the lightpath is torn down after the flow has long being gone (or with long gaps in the activity), which leaves the lightpath idle, whereas it could be used by other flows.

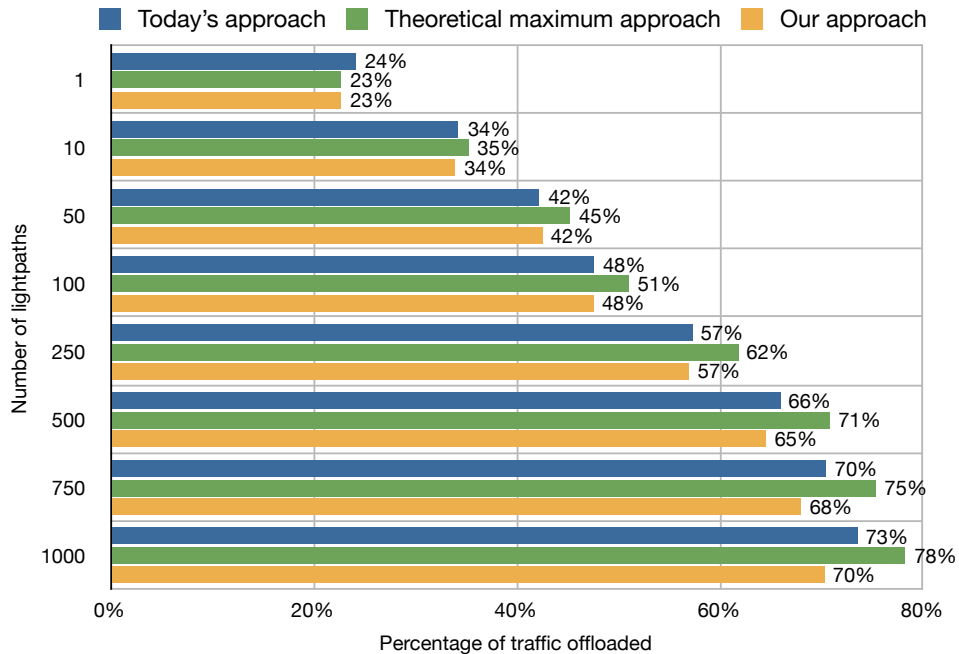


Figure 4.9: Percentage of offloaded traffic per lightpath by considering the three evaluated approaches.

We also vary the number of available lightpaths as $\{1, 10, 50, 100, 250, 500, 750, 1000\}$, even though, the actual number of lambdas (wavelengths) per fiber is not so high (*e.g.*, today's DWDM systems support up to 160 wavelengths per fiber) [13] [70]. For each number of available lightpaths, we observe the amount of traffic offloaded by using each approach. Moreover, we only use 30 min of network activity in order to speed up the analysis and evaluation.

Figure 4.9 shows the percentage of traffic that would have been offloaded per lightpath when the three approaches are considered. It also shows that the more lightpaths are made available, more traffic is offloaded. Moreover, it is seen that the three approaches have similar performance when offloading flows, being the theoretical maximum approach considerably better than the others, the higher the number of lightpaths are. The exception for that is when only one lightpath is available. In this case, the today's approach is better. The reason for that is due to the idleness state that our approach and the theoretical maximum approach stay while waiting for network information (that just came after 120 seconds in the case of our

experimentation). However, as an overall performance, the theoretical maximum approach has the best performance. The reason for that comes from the fact that, at each decision point, the theoretical maximum approach knows which flows will have the highest volume until the next decision is taken due to its knowledge about the future. However, our approach performs quite well, even though it lacks future information about the flows.

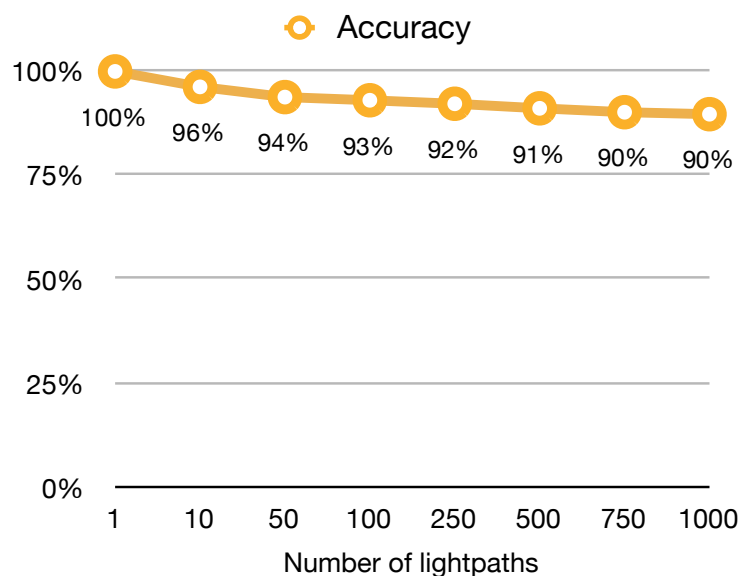


Figure 4.10: Accuracy of our proposal when compared with best theoretical result.

Figure 4.10 shows the accuracy of our approach when selecting flows to be offloaded over a certain number of lightpaths (x -axis). The accuracy is calculated by dividing the amount of traffic offloaded with our approach by the amount of traffic offloaded with the theoretical maximum approach (*i.e.*, the best theoretical result). It is seen that our approach has a good accuracy when compared with the theoretical maximum approach, being able to properly detect flows that represent most of the traffic. The main reason for that is due to the nature of large flows. When there is no disturbance in their transmission (*e.g.*, packet loss), large flows will most likely **continuously** transmit data at a slightly **constant** rate until their termination. As explained in Section 4.4, our approach takes these assumptions into consideration to weigh flows and, as a result, tends to choose the best flows.

4.6 Grooming flows over lightpaths

ONCE flows have been elected and selected to be offloaded to the optical level, they should be properly accommodated over lightpaths. In order to efficiently make use of the capacity that lightpaths provide, traffic grooming is a technique that can be employed. Traffic grooming is the process of multiplexing many flows into a single lightpath [22]. For instance, in a hybrid network using both TDM and WDM multiplexing, two flows which are destined to the same end-point can be placed on the same wavelength (lightpath). It is worth highlighting that the objective of grooming is to minimize costs in the network while still guaranteeing QoS [163].

Traffic grooming strategies can be aligned within the scope of operations research [173]. Operations research is a branch of mathematics that incorporate diverse broad areas, such as agricultural planing, biotechnology, military operations, amongst others. The main objective of operations research is optimization, more specifically the act of performing the best under certain circumstances.

This general concept of operations research can also be applied in network traffic optimization, which we focus on the research work developed by Sabella *et al.* [129]. In such work, the authors investigated two grooming strategies to accommodate IP flows over lightpaths, namely *spreading* and *packing*. The former consists of evenly distributing flows over lightpaths, whereas the latter consists of assigning a flow to the mostly loaded lightpath first. By doing so, there is a greater chance of finding available lightpaths for IP flows demanding large bandwidth. The authors evaluated these grooming strategies in terms of blocking probability, *i.e.*, the percentage of flows that are successfully accommodated over lightpaths. Their conclusion is that by using the *packing* strategy, there is a greater reduction in the percentage of blocked bandwidth requests when compared with the *spreading* strategy.

Different from Sabella *et al.*'s research, we do not observe the percentage of blocking requests per grooming strategy, but the percentage of IP traffic that can be offloaded from the IP level to the optical level by means of accommodating elephant flows over lightpaths. We suspect that when using the *packing strategy*, the chance of accommodating large elephant flows is higher than when using the *spreading strategy*. In order to find out whether our assumption is correct, we perform simulations while considering the following strategies to offload elephant flows over lightpaths:

1. **Dedicated:** elephant flows are exclusively allocated to dedicated lightpaths.
2. **Spreading:** elephant flows are groomed over the least loaded lightpath first.
3. **Packing:** elephant flows are groomed over the mostly loaded lightpath first.

Our goal here is to observe what percentage of IP traffic can be offloaded from

the IP level to the optical level when using the aforementioned strategies. In order to achieve our goal, we conduct our investigation through simulation using the ns-2 network simulator [112] as described in the next subsection.

4.6.1 Simulation setup

In this subsection, we present further details about our simulation, more specifically the network topology used. Figure 4.11 shows the topology used in our simulations.

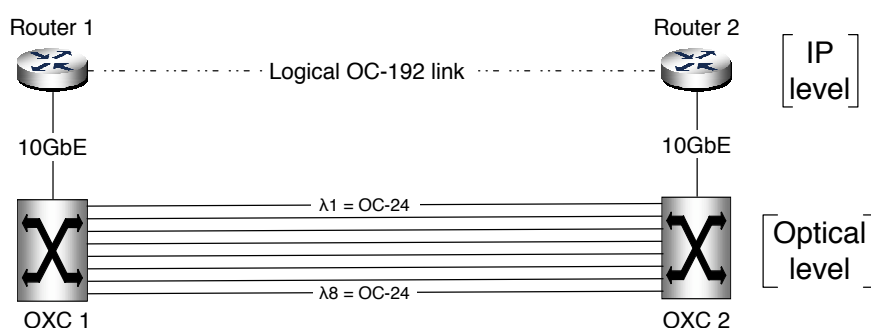


Figure 4.11: The topology used in our simulation.

The topology used in our simulation comprises of Router 1 and Router 2 being logically connected via one OC-192 link (9.952 Gbps), but physically connected via 8 x OC-24 links (1.244 Gbps). On its turn, the simulation consists of Router 1 transmitting flows unidirectionally to Router 2. The flow rates vary from 1 Mbps up to the total bandwidth available on an OC-24 link. Moreover, we limited the total aggregated flow rate over time to be smaller than 9.952 Gbps in order to avoid packet loss. Lastly, the simulation runtime is 200.000 seconds (roughly 2 days). During the simulation runtime, we assume that the start time interval of flows follows a negative exponential distribution, while the duration of a flow is assumed to follow a Weibull heavy-tailed distribution (Subsection 4.4.2).

The decision to offload a flow from the IP level to the optical level is taken every second. Similar to our weighting method, the criterion to offload flows is based on their throughput. Flows with higher throughput are offloaded first. It is worth saying, although, that we do not consider flow survivability in order to ease our simulation. Once a flow is allocated to a lightpath, it stays over this lightpath until the flow terminates. Flows that are not accommodated over a lightpath (due to the lack of available bandwidth, for instance) are kept sending data at the IP level until there is bandwidth available at the optical level.

4.6.2 Performance analysis

Figure 4.12 shows the traffic offload percentage per offloading strategy. The x -axis shows the total simulation period represented in seconds. On its turn, the y -axis represents the percentage of offloaded traffic.

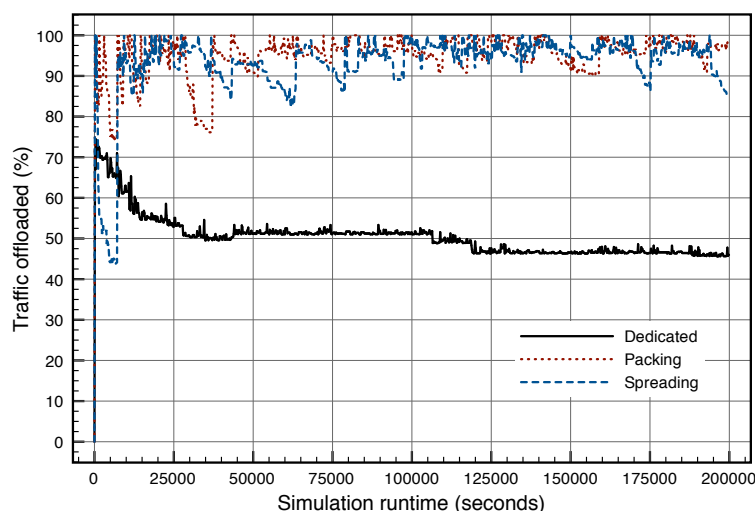


Figure 4.12: The offload percentage per offloading strategy.

As observed in Figure 4.12, the *dedicated* strategy is the one that offloads the least traffic among the three strategies. On average, the percentage of offloaded traffic is around 50%. The reason for such a low percentage of offloaded traffic comes from the fact that a dedicated strategy assures that a lightpath is allocated to a single flow and to no others. Therefore, a flow will only release this lightpath once it terminates its activity. On the other hand, when multiplexing (grooming) flows over lightpaths, the percentage of offloaded traffic is much higher. By using the *spreading* strategy, the flows are more evenly distributed over lightpaths at the cost of a rather irregular usage of the optical level. That occurs because the lightpaths are equally filled up, which decreases the chance that there is a lightpath with enough bandwidth to accommodate a large flow. When there is enough bandwidth for this large flow, the available bandwidth is mostly consumed by it. On average, the amount of traffic offloaded when using the *spreading* strategy stayed around 91%. On its turn, the *packing* strategy outperforms the *spreading* strategy by offloading approximately 95% of the total traffic. The main reason for that is that the *packing* strategy fills up a

single lightpath first, and then fills up the others next. By doing so, a greater chance to accommodate a large flow over a lightpath exists, which makes the usage of the lightpaths more efficiently.

4.6.3 Energy consumption

As a side research, we also simulated the energy consumption of the three mentioned strategies. We believe that the use of self-management to establish and release lightpaths on-demand may reduce energy consumption. Investigations about a “green Internet” have been considered as a hot topic lately [158] [9] [73]. Within the context of our simulation, we expect that “packing” flows into a single wavelength saves more energy than allocating several wavelengths for flows. In order to observe that, we informally talked with network operators to find out what equipments are mostly involved in optical transmissions as well as their respective power consumptions. The result of that is depicted in Figure 4.13, which represents the same topology shown in Figure 4.11.

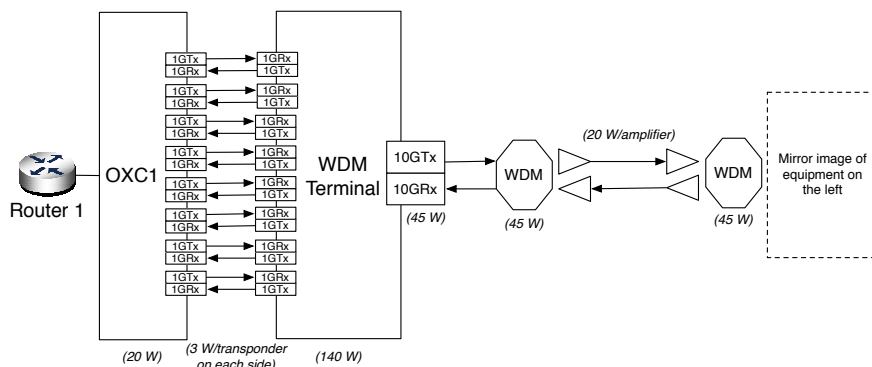


Figure 4.13: The energy consumption.

Here, we only consider the energy consumption at the optical level. The 8 x OC-24 lightpaths depicted in Figure 4.11 are represented in Figure 4.13 as 8 x 1 Gbps transponders. These transponders have their counter-parts in the WDM terminal. On its turn, the WDM terminal has a 10 Gbps transponder that connects with a demux. This demux connects to its counter-part, with amplifiers in between. Then, all the optical equipments aforementioned repeat themselves in inverted order until OXC2.

The power consumption values of each optical element is depicted in Figure 4.13. It is important to highlight that the transponders are assumed to switch on and off

on-demand. That is, they are automatically switched on when there is data to be transmitted and switched off when there is no data transmission (to save energy). The simulation here behaves as follows:

- If no data is transmitted between OXC1 and OXC2, then no link between the OXCs and the WDM terminals is established, i.e., there is no light being generated. The minimum energy consumption in this situation will consist of the energy consumption of the OXCs + the WDM terminals + 10GTx/Rx packs + WDMs + amplifiers on both sides.
- If there is at least one link established, the minimum energy consumption will consist of the minimum energy consumption above, plus the corresponding links transponders.
- The maximum energy consumption will occur when all 8 links between the OXCs and the WDM terminals are established.

What we observe here is the amount of energy consumed while varying offloading strategies to accommodate flows over established lightpaths. Somewhat opposite to our initial expectation, the obtained result (Table 4.2) shows that there is no significant reduction in the energy consumption when using any of the considered strategies. The reason for that comes from the fact that the volume of traffic is so intense that all the 8 links are kept established most of the time. The *packing* strategy indeed saves more energy than the other strategies, but the difference is negligible.

Dedicated	Spreading	Packing
99.98%	99.94%	99.92%

Table 4.2: The percentage of time with all the 8 links up and running.

4.7 Concluding remarks

THIS chapter focused on the autonomic decision process of our self-management approach. Special attention was drawn to the algorithm (Subsection 4.3.4) used in our autonomic decision process as well as to our decision policy (Subsection 4.3.3). Our autonomic decision process was presented by showing the main actions taken when deciding the destiny of flows, *i.e.*, their promotion or demotion. These actions were also related with cache theory due to their similarities. However, as presented in Section 4.2, even though there are similarities between our algorithm

and well-known caching algorithms, they significantly differ in focus: caching algorithms focus on reducing latency in obtaining information, while our algorithm focus on selecting big flows to the optical level.

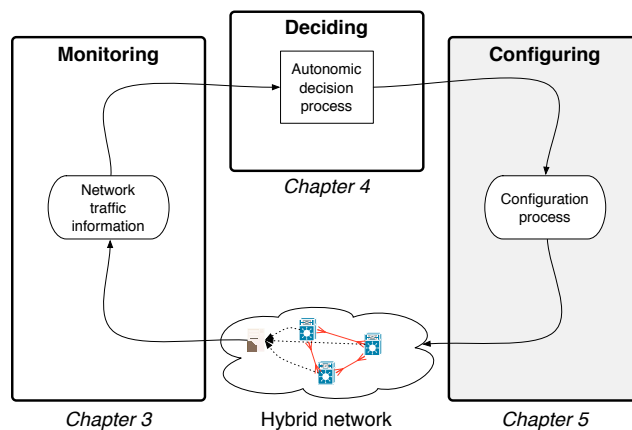
On its turn, our decision policy is strongly based on the assumptions (Section 4.4) that: 1) a flow has a constant throughput during its lifetime, and 2) the longer a flow has already lasted, the higher the chances are this flow will continue transferring data. These assumptions allowed us to weigh flows as a way to estimate their future behavior in place of precisely calculating it, which is practically impossible. A validation (Section 4.5) was carried to compare how our proposal behaves when compared with the current manual way to manage lightpaths (referred to as today's approach) as well as with the best theoretical result (referred to as theoretical maximum approach). The results showed that our proposal shows an accuracy of approximately 90% when compared with the best theoretical result.

Last but not least, we also observed how different offloading strategies can influence the percentage of IP traffic moved to the optical by means of offloading large elephant flows (Section 4.6). By using the *packing* strategy, a larger percentage of IP traffic can be offloaded to the optical level, since such a strategy increases the chances of accommodating flows with large bandwidth requirements. As a side research, we also observed whether the use of different offloading strategies along with self-management principles would reduce power consumption in optical transmission. Our analysis showed that, independently of the offloading strategy employed, the amount of traffic is constantly high over time, which results in lightpaths being kept established most of the time.

Chapter 5

The impact of self-managing lightpaths

With the introduction of self-management principles over lightpaths, one aspect to be regarded is the impact that the self-management of lightpaths may cause. When autonomically creating and releasing lightpaths for selected IP flows, some undesirable effects may be experienced. The objective of this chapter is to present the impact that moving flows to the optical level on the fly may have on the flow throughput. In order to observe that, we perform simulations while observing the flow throughput performance. Following that, we also perform a literature study in order to highlight additional aspects to be regarded when employing self-management of lightpaths.



The organization of the present chapter is as follows:

- *Section 5.1 presents the side effect of moving flows from the IP level to the optical level on the fly.*
- *Section 5.2 presents some additional aspects to be considered when employing self-management of lightpaths.*

5.1 The side effect of moving flows on the fly

IN this section we are particularly interested in observing whether there is any degradation of performance when flows are moved on the fly over lightpaths. We suspect that when ongoing flows are moved from the IP level to the optical level, many packets belonging to these flows can arrive out of order at the receiver (and even discarded). That may occur due to the fact that some of these packets can be transferred more quickly over lightpaths than the other packets over IP paths. For our analysis, we are particularly interested in observing our suspicion on TCP flows due to: 1) their high importance in current networks, 2) their representation of most of the IP traffic, and 3) the fact that TCP is connection-oriented, which results in TCP being more complex (*e.g.*, packet reordering) to guarantee a reliable service than connectionless protocols (*e.g.*, UDP). Since TCP flows are highly representative, we believe that our self-management approach will deal more with TCP flows than with any other kind of flows. Therefore, a study on the effects of employing self-management principles on TCP flows is very relevant to our research.

Within this context, the question we pose is: *how would moving flows on the fly impact on the TCP throughput?* It is known that TCP performs packet reordering, but little is known about its limits under conditions imposed by moving flows on the fly. It may happen that due to the switching between IP and optical levels at a high traffic rate, TCP will need to drop packets. Therefore, the goal of this section is to present a study of the impact of optical switching on TCP throughput.

In order to address our research question, we first start with a review of the literature to examine the different versions of TCP used today. Out of many TCP flavors, we chose TCP CUBIC [67] due to its special design for high-speed networks and for being the default version in Linux systems. Next, we identified a set of factors that may limit the throughput of a TCP (CUBIC) flow. These factors are then taken into account in our simulations and we observe how they impact on the TCP performance under the conditions imposed in our analysis. We conducted our simulations using the ns-2 network simulator [112].

5.1.1 Simulation Setup

In this subsection, we present the details of our simulation, including the network topology, the evaluation criteria, and the proposed scenarios.

Network Topology

Figure 5.1 shows the topology used in our simulations. It consists of three routers ($r1$, $r2$, and $r3$) and two nodes (Sender and Receiver) connected by two differ-

ent paths: the IP path ($r1-r2-r3$) and the optical path ($r1-r3$).

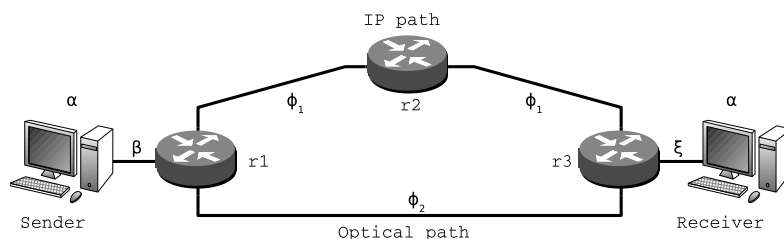


Figure 5.1: Topology used in the simulations and limiting factors (Greek letters).

The simulation starts first with the sender opening a single unidirectional TCP connection with the receiver and then sending data forwarded via the IP path. After reaching pre-defined throughput values, the router $r1$ performs the move of the TCP flow, forwarding thus all the data to the receiver via the optical path. It is important to observe that the move affects only one direction of the flow ($sender \rightarrow receiver$), and that the acknowledgment packets (ACKs) in the reverse direction ($receiver \rightarrow sender$) continue to use the IP path. We chose this approach because ACK traffic is significantly smaller than TCP data traffic to justify being moved to an optical connection. For a short period of time after the move – the *transient phase* – there will be data packets on both the IP and optical paths. After the transient phase, the simulation reaches a new phase, in which there are only data packets on the optical path and only ACKs on the IP path. The simulation finishes soon afterwards, depending on the scenario used.

Given the network topology, we now present the evaluation criteria used for our simulations.

Evaluation Criteria

To evaluate how the TCP flow is affected by moving flows on the fly, we analyzed the trace files generated by ns-2 to compute the throughput perceived by the receiver machine. We are interested not only in observing the maximum throughput, but also in determining *how the throughput changes* immediately after the optical switching.

It is known that the maximum throughput of a TCP flow is limited by such factors as the application protocol utilized (FTP, http, etc.), the TCP buffers sizes employed, the link capacity, and whether there is packet loss on the network [150]. In this work we investigate the impact on TCP throughput when flows are limited by TCP buffer sizes and link capacity on different parts of the network.

Simulation Scenarios

Based on the aforementioned limiting factors for TCP throughput, we have defined four simulation scenarios that represents different configuration setups for optical switching enabled networks. These scenarios are defined as follows:

- **Scenario A:** the size of the TCP buffers (α s in Figure 5.1) are the limiting factors for TCP throughput;
- **Scenario B:** the capacity of the sender's local link (β in Figure 5.1) is the limiting factor for the TCP throughput;
- **Scenario C:** the capacity of the core links (ϕ_1 and ϕ_2 in Figure 5.1) acts as the limiting factor; in fact, we have subdivided scenario C in two: when the capacity of IP links is the same on the optical links ($\phi_1 = \phi_2$) and when the capacity of optical links is larger than the one from the IP links ($\phi_2 > \phi_1$);
- **Scenario D:** the receiver local link is the limiting factor (ξ in Figure 5.1).

Figure 5.2 provides an overview of the mapping between the evaluated scenarios (in capital letters) and the network parts.

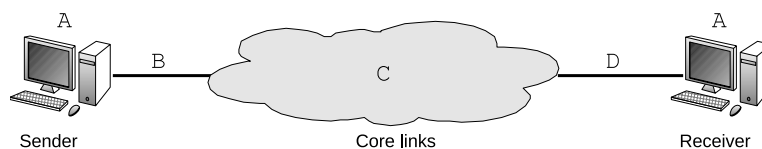


Figure 5.2: Overview of the evaluated scenarios and the network.

After defining the scenarios, we defined *the moment* at which a flow should be moved to the optical level. For each scenario, we have configured its limiting factor to restrict flows to three data rates (100 Mbps, 1 Gbps, and 10 Gbps) and then we performed the move at each one of these rates. We could have used flow duration as a threshold instead, but this would lead to different comparison conditions, since flows would present dissimilar throughput values before the move.

In addition, we have evaluated flows on the aforementioned rates according to three Round-Trip Time (RTT) values (Table 5.1), given the limiting factor specified by each scenario. In high speed networks, RTT is usually a function of the physical distance between the sender and the receiver. In this study, we have simulated three different situations in which sender and receiver are located in hypothetical cities: (i) close to each other, (ii) relatively far from each other, and (iii) very far from each

Distance	RTT before switching	RTT after switching
Close	10 ms	6 ms
Relatively Far	100 ms	60 ms
Very far	1000 ms	600 ms

Table 5.1: *RTT values employed in the simulations.*

other. In each case, we assumed that the RTT value after the move would be smaller than before, since packets could avoid the routing process at the IP level.

Table 5.2 summarizes the values of the parameters used in our simulation scenarios, where each line represents a single simulation. For example, we have evaluated three cases for Scenario B, when the flow rate reaches the limiting rates (100 Mbps, 1 Gbps, and 10 Gbps) specified by its limiting factor (β). For each case, we have defined the values for the limiting factors (Greek letters of Figure 5.1). Due to space constraints, we only present those cases where the RTT is equal to 10 ms. An equal number of simulations was conducted for 100 ms and 1000 ms RTT. For these RTT values, only α values in Scenario A change – they are multiplied by 10 in the case of 100 ms and by 100 when RTT is 1000 ms.

Scenario	Limiting Rate	α (rtt=10ms)	β	ϕ_1	ϕ_2	ξ
A	100 Mbps	0.125 MB	622.08 Mbps	622.08 Mbps	622.08 Mbps	622.08 Mbps
	1 Gbps	1.25 MB	2.488 Gbps	2.488 Gbps	2.488 Gbps	2.488 Gbps
	10 Gbps	12.5 MB	39.813 Gbps	39.813 Gbps	39.813 Gbps	39.813 Gbps
B	100 Mbps	1.16 GB	100 Mbps	622.08 Mbps	622.08 Mbps	622.08 Mbps
	1 Gbps	1.16 GB	1 Gbps	2.488 Gbps	2.488 Gbps	2.488 Gbps
	10 Gbps	1.16 GB	10 Gbps	39.813 Gbps	39.813 Gbps	39.813 Gbps
C1	100 Mbps	1.16 GB	622.08 Mbps	100Mbps	100Mbps	622.08Mbps
	1 Gbps	1.16 GB	2.488 Gbps	1 Gbps	1 Gbps	2.488 Gbps
	10 Gbps	1.16 GB	39.813 Gbps	10 Gbps	10 Gbps	39.813 Gbps
C2	100 Mbps	1.16 GB	622.08 Mbps	100 Mbps	622.08 Mbps	622.08 Mbps
	1 Gbps	1.16 GB	2.488 Gbps	1 Gbps	2.488 Gbps	2.488 Gbps
	10 Gbps	1.16 GB	39.813 Gbps	10 Gbps	39.813 Gbps	39.813 Gbps
D	100 Mbps	1.16 GB	622.08 Mbps	622.08 Mbps	622.08 Mbps	100 Mbps
	1 Gbps	1.16 GB	2.488 Gbps	2.488 Gbps	2.488 Gbps	1 Gbps
	10 Gbps	1.16 GB	39.813 Gbps	39.813 Gbps	39.813 Gbps	10 Gbps

Table 5.2: *Scenarios and values used on the limiting factors for RTT = 10 ms.*

TCP Congestion Control and why TCP CUBIC

Current versions of TCP employ four intertwined algorithms to provide congestion control: slow start, congestion avoidance, fast retransmit, and fast recovery [3]. When a TCP connection is established, the sender sets a value for its congestion window ($cwnd$) smaller or equal to the sender's maximum segment size ($smss$),

i.e., $cwnd \leq 2 \times smss$. As acknowledgment messages reach the sender, it increases its congestion window *exponentially* using the slow start algorithm until reaching a pre-determined value, the slow start threshold ($ssthresh$). When the congestion window size becomes larger than the slow start threshold ($cwnd \geq ssthresh$), TCP stops using the slow start algorithm and starts to use the congestion avoidance algorithm, which, by default, increases sender's congestion window *linearly*.

This linear increase of the congestion window minimizes the risk of congestion in the network. However, when this algorithm is employed in networks with large Bandwidth-Delay Product (BDP), it may lead to severe underutilization. As observed by Ha *et. al* [67], it may take 1.4 hours for TCP to grow its window to the BDP in a network of 10 Gbps capacity, round-trip time equal to 100 ms and packet size of 1.250 bytes. Only after this time the sender is able to utilize the full capacity offered by the network. If the transmission ends before this time, the sender would have underutilized the path.

In order to deal with this problem in high speed networks, several congestion avoidance algorithms have been proposed, such as FAST [82], HSTCP [61], STCP [85], HTCP [89], Westwood [98], BIC-TCP [180], and CUBIC [67]. Among those, we have selected CUBIC (version 2.1) to be used in our simulations since it is the default version used in the Linux kernel – thus allowing us to have a more realistic simulation.

Simulation Tool: ns-2

In our simulations we used the ns-2 network simulator, version 2.33. Besides being one of the most used simulation tools by the network community, we choose ns-2 also because its latest versions were enhanced with an extension named TCP-Linux [166]. This extension allows to import directly the source code of TCP congestion avoidance algorithms from the Linux kernel, instead of using a code specifically developed for the simulator. Since TCP CUBIC is available in the Linux kernels for a while, we were able to have a more realistic simulation by using code from the kernel. In the following subsections, we describe each simulation scenario in more detail and present the results obtained from executing our simulations.

5.1.2 Scenario A: TCP Buffers as the Limiting Factor

In this scenario we configured the TCP buffers to act as the limiting factor of the TCP throughput. Therefore we set values of the other limiting factors (links capacity: β , ϕ , and ξ in Figure 5.1) high enough to allow the network to handle all the data generated by the sender, as presented in Table 5.2. Then, the next step was to determine the required buffer sizes. To do that, we simply calculated the bandwidth-delay

product, by multiplying the limiting flow rate to perform the switch (100 Mbps, 1 Gbps or 10 Gbps) by the RTT before the switch. This value indicates the maximum amount of data on the network at any given time, and TCP buffers must have at least this size in order to handle these rates.

Analyzing all the results for Scenario A, we observed one single behavior independently of RTT and buffer sizes: after the switching, the TCP throughput increases without any packet loss. Therefore, we have chosen to present only the cases where flows are limited to 1 Gbps before switching.

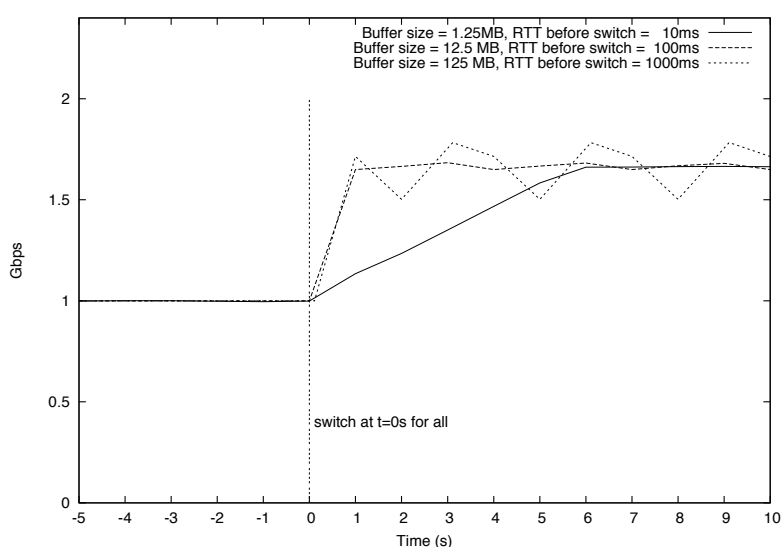


Figure 5.3: Throughput of TCP flows in Scenario A (Limiting rate = 1 Gbps , Granularity = 1000 ms).

Figure 5.3 shows these results. In this figure, the switch between levels occurs at $t = 0$, denoted by a vertical line. As it can be seen, before the switch, the flows present a stable rate of 1 Gbps limited only by their own buffers. However, after the switch, no throughput reduction is observed. In fact, we can observe that CUBIC reacts well to the new configuration, and the throughput increases quickly to the expected new theoretical value of 1.667 Gbps (obtained by dividing the buffer size by the RTT).

When computing throughput, one important factor is the granularity, *i.e.*, the time interval used to average the data rate transmitted. If the granularity is very coarse, it may mask some effects that might occur in the network. If it is too fine, it may present a too detailed view, which makes the acquisition of a general view of

the throughput more complex. Figure 5.4 presents the throughput results obtained from the same simulation of Figure 5.3, but with different granularity values.

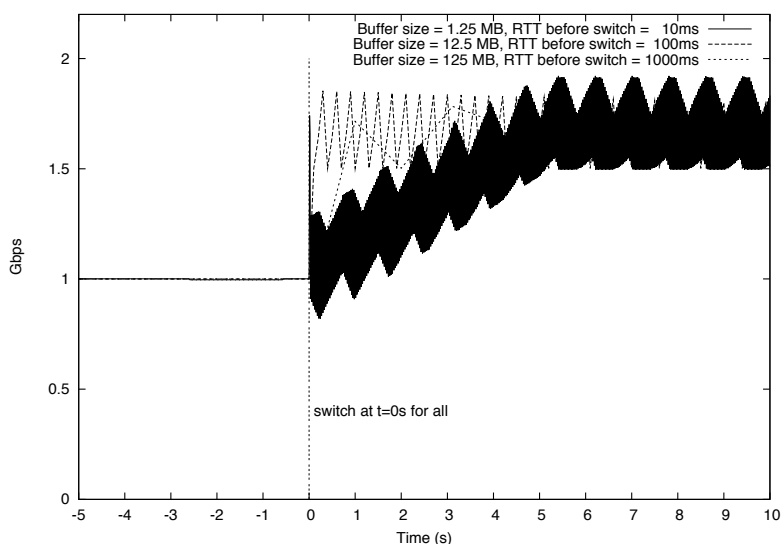


Figure 5.4: Throughput of TCP flows in Scenario A (Limiting rate = 1 Gbps , Granularity = RTT (before the move)).

In Figure 5.4, we can notice the drawbacks of having a finer granularity and also an oscillation on the throughput after the move. In our study, we decided to employ the same granularity (1000 ms) on all the subsequent figures to have common comparison criteria. In cases where this value masks important events during the transmission, we then highlight that and present numerical results in the text.

Despite the granularity used, we can conclude based on Figure 5.4 that when the buffer size is the limiting factor, the users will experience a brief decrease on the throughput when a flow transition occurs followed by an oscillating increase of the throughput.

5.1.3 Scenario B – Sender’s Local Link as the Limiting Factor

For Scenario B, we configured the sender’s local link (β in Figure 5.1) as the limiting factor for TCP throughput and set high values for the other parameters, as defined in Table 5.2.

When imposing the limiting factor on the sender’s link, one could intuitively assume that moving flows on the fly in the core network would not affect the TCP performance, since the limiting factor is not placed on the core network. However,

this is not the case. What happens, in fact, is a fast increase in the throughput perceived by the receiver, followed by a decrease.

To better demonstrate this, Figure 5.5 shows the results for the case where the local access link has a capacity of 1 Gbps and the RTT value is equal to 100 ms. In this figure, the dotted line represents the throughput when we run the simulation without executing the switch, while the solid line represents the same simulation, but with the switch being executed at $t = 100$ (vertical dotted line).

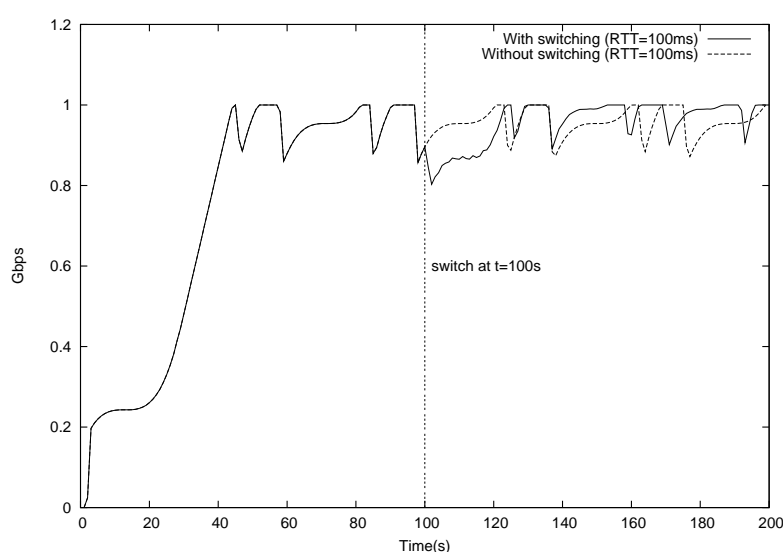


Figure 5.5: Throughput of TCP flows in scenario B for $RTT=100$ ms, $\beta = 1$ Gbps and granularity equals to 1000 ms.

Analyzing Figure 5.5 further, it is possible to observe that the throughput seems to decrease after the switch in comparison to when no switch occurs. What happens is that between 100 and 100,1s a peak of 1.304 Gbps takes place (not shown in the graph due to the granularity used), due to packets arriving from both IP and optical paths during the transient phase. However, packets arriving from the optical path during this phase have, in fact, a higher sequence number than the ones coming from the IP path. This causes the receiver to perform reordering and ask for retransmissions (so 14.186 duplicate ACKs are sent), resulting on the reduction of the congestion window and, finally, to a throughput reduction. This can be seen at $t = 102$ s, where the throughput reaches 802.98 Mbps.

Based on these results, we can conclude that, when the sender's local link is the limiting factor, there is no major problem when moving flows on the fly. There

are only some oscillations in the throughput due to reordering, which TCP CUBIC recovers relatively fast (few seconds) from that.

5.1.4 Scenario C – Core link as the Limiting Factor

In Scenario C, we configured the core links as the limiting factors for TCP throughput. This scenario was divided into two, where the first (C1) has the same link capacity on both IP and optical links ($\phi_1 = \phi_2$) and the second (C2) has a large capacity on the optical link ($\phi_2 > \phi_1$). To set the backbone links as the limiting factor, we have to configure the other parameters as listed in Table 5.2.

The behavior observed for the TCP throughput associated with the Scenario C1 was basically the same as that observed in Scenario B, where it oscillates during the transient phase and then normalizes. Therefore, in this subsection we give more focus on how different RTT values impact the throughput of TCP, when flows are moved to the optical level.

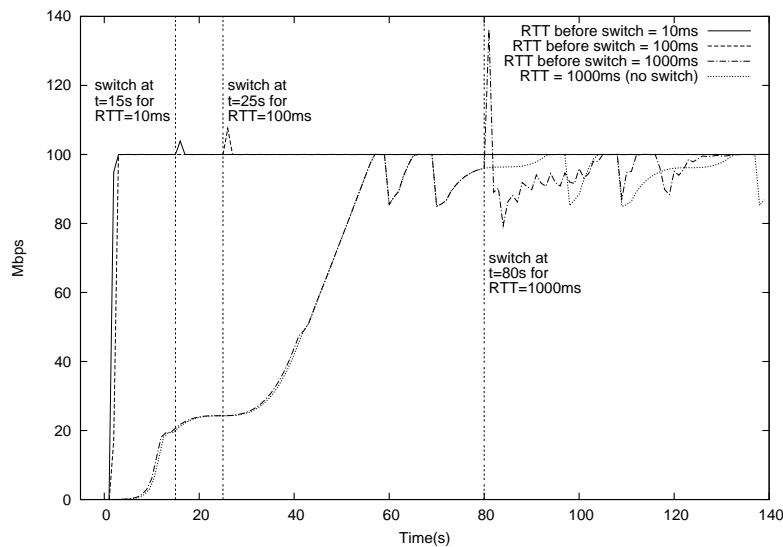


Figure 5.6: Throughput of TCP flows in Scenario C1 when $\phi_1 = \phi_2 = 100$ Mbps, and granularity equals to 1000 ms.

Figure 5.6 presents the results for 100 Mbps flows. In this figure, one can observe a peak in the throughput for each curve after the switch. This happens because packets arrive from both paths during the transient phase. In the first moment, one could conclude that difference in the peaks are due to different RTT values.

However, when calculating the throughput with a granularity of 10 ms, we observed peaks of almost 200 Mbps (due to 100 Mbps flows on each path) for the three RTT values, which are masked when computing the throughput using a granularity of 1000 ms.

In the case of Scenario C2, a similar behavior was observed as in Scenario C1 during the transient phase: the throughput oscillation. However, after this oscillation, the throughput increases significantly, due to the higher link capacity on the optical path. Figure 5.7 shows the results when ϕ_2 is equal to 39.813 Gbps. For instance, when the RTT is equal to 10 ms, after the switch the TCP throughput reaches the maximum link capacity in 71s¹. Moreover, it is possible to observe that larger RTT values (100 ms) demand more time to reach the maximum link utilization, but they still benefit from being moved to the optical level.

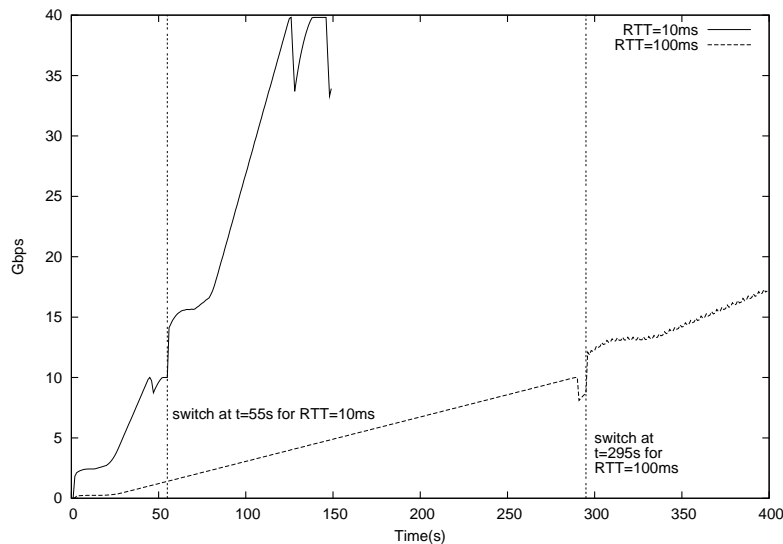


Figure 5.7: Throughput of TCP flows in Scenario C2 (1000 ms granularity).

Taking into account these results, when the limiting factor for TCP is present within the core network, we can conclude that moving flows on the fly presents no major problem. There is only an oscillation on the throughput during the transient phase due to reordering. We can also observe that the larger the RTT values, the longer it takes for the TCP flow to recovery from the move.

¹The throughput increases linearly for both curves in Figure 5.7 (and not following a cubic function as expected) due to a limitation of TCP CUBIC 2.1, which was removed in version 2.2 [67].

5.1.5 Scenario D – Receiver’s Local Link as the Limiting Factor

For our final scenario, we have set the capacity of the receiver local link (ξ in Figure 5.1) as the limiting factor for TCP throughput. As in previous scenarios, the other variables were over dimensioned in order to identify the impact of a limiting factor in the receiver local link, as described in Table 5.2.

The major problems caused by moving flows on the fly were observed in Scenario D. Figure 5.8 shows the results when the local receiver’s local link has a capacity of 10 Gbps (ξ), while the other links have a capacity of 39.813 Gbps.

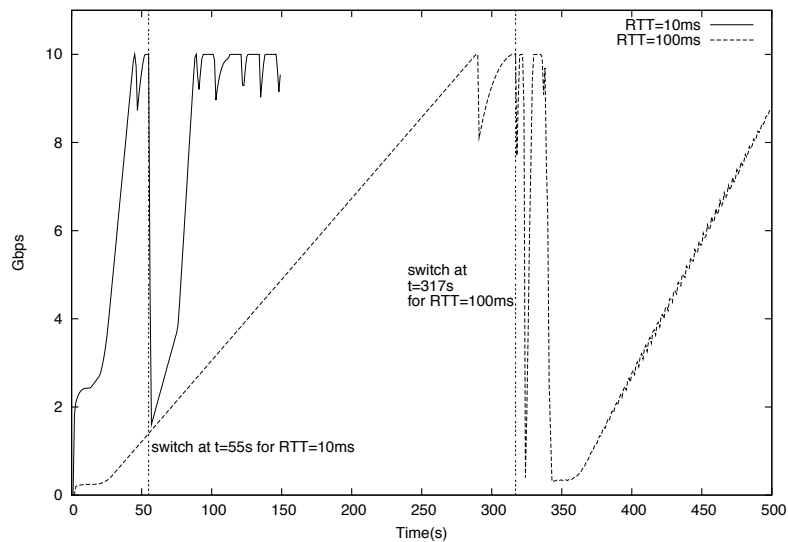


Figure 5.8: Throughput of TCP flows for Scenario D (1000 ms granularity)

As can be seen, there is a significant reduction in the throughput for both RTT values after the switch. The reason for this is that the router r_3 receives a large volume of data during the transient phase, at approximately 10 Gbps from each link (twice as much it can forward via the receiver local link), which causes its queue to be filled up and packets to be dropped when there is no more space in the queue (we used a 500 packets queue size for all simulations). For instance, in the case where RTT is 10 ms (solid line), 335.107 packets are discarded by r_3 within 200 ms (between 55,0 and 55,2 seconds), which represents a total of 502.66 MB of lost data. This, in turn, causes the receiver to send a massive number of duplicated ACKs, which leads the sender to reduce its congestion window. The consequences can be seen at $t = 57.0s$, when the throughput reaches its minimum value of 1.602 Gbps.

The same reasoning applies to the curve with 100 ms RTT. In Scenario B we also observed reduction of the congestion window. However, in Scenario B this was only due to packet reordering, whereas in Scenario D a huge number of packets were discarded, causing the decrease of the congestion window to be much larger.

After the transient phase, for the 10 ms RTT case, the TCP flow takes 33 seconds to reach the same throughput levels as before the switch. This clearly indicates, under the conditions imposed in this scenario, that moving flows on the fly would not be perceived as a seamless transition by end users. Instead, they would notice a significant reduction on the throughput followed by a recovery and then similar throughput values to before the move, but with a smaller RTT.

The results obtained in this scenario suggest that the capacity of the receiver's local link and the router's buffer size should be taken into account. One possibility would be to over provision the receiver's local link in order to support the aggregated volume of data. In the case of our simulation, the receiver's local link should have been over provisioned in order to cope with the aggregated volume of data coming from both optical and IP paths.

A Closer Look at R3's Queue

Since in this scenario the bottleneck is located at the receiver's access link, a closer look at router $r3$ queue may provide some insight on how the packet discards occur.

Figure 5.9 shows how $r3$'s queue size changes along with time. Intuitively, one could think that $r3$'s queue would be used only during the transient phase, when there are flows on both paths. However, this is not the case. As one can see, the queue grows even before the switching, when data flows only on the IP path. For instance, at $t = 45,8s$ the queue size is equal to 468 packets (maximum is equal to 500). This can be explained by the window increase algorithm employed by TCP CUBIC. Since CUBIC does not take into account properties such as delay (as TCP Vegas), it keeps increasing the window until a packet loss occurs. This aggressive increase of the congestion window can create a burst of data – which may grow beyond the 10 Gbps output capacity of $r3$. As a consequence, $r3$ has to queue packets. If there is not enough space in buffers, packets will be simply discarded, as in between $t = 45,8s$ and $t = 46,0s$, where 3 packets were discarded.

This discard, in turn, causes the receiver to send duplicate ACK messages – which leads to a decrease on the congestion window size and finally on throughput reduction. This can be seen at $t = 47s$ in Figure 5.8, where it reached 8.73 Gbps. When the switching occurs at $t = 55s$, the queue size is already 343 packets – which means less space for buffering the packets coming on both ways. As mention in the previous subsection, this causes a massive discard of packets, the throughput is

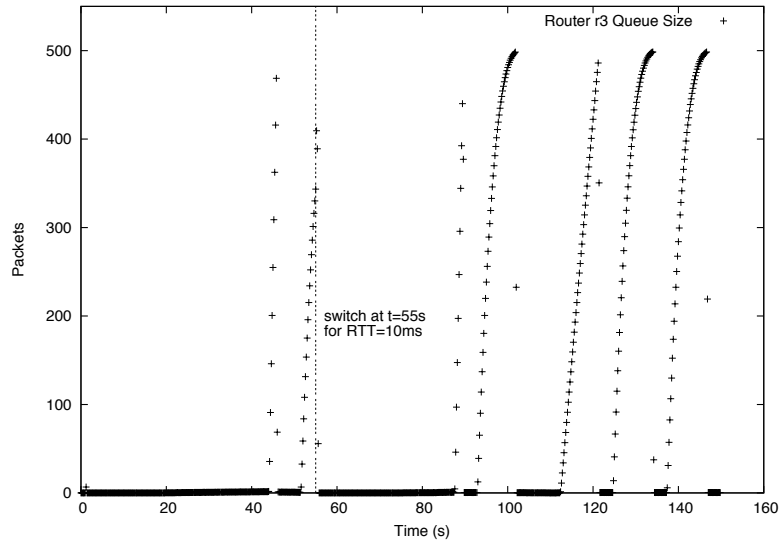


Figure 5.9: Router r_3 Queue Size for 10Gbps case with $RTT = 10ms$

reduced to 1.602 Gbps at $t = 57s$.

Comparing the solid curve in Figure 5.8 with Figure 5.9, it is possible to see that the throughput reductions on Figure 5.8 match with the queue size reduction in Figure 5.9. The aforementioned reasoning explains all these reductions. Due to that, administrators may expect reduction on TCP CUBIC throughput after moving flows on the fly if the receiver's local link and the router's buffer size are not regarded.

5.1.6 Concluding remarks

In this section, we presented an analysis of the impact of moving flows on the fly on TCP throughput. TCP CUBIC was the TCP flavor employed in our analysis due to its design for high-speed networks and its use in recent versions of Linux kernels. Our analysis was performed using the ns-2 simulator and considered four scenarios as limiting factors of TCP throughput (Subsection 5.1.1). The scenarios were chosen by selecting the known factors that may limit the TCP throughput. Our conclusion is that depending on the scenario, different levels of impact on the throughput of TCP flows are observed.

When the size of the TCP buffers in the sender and receiver is the limiting factor (Scenario A), the throughput has just a small oscillation when the move place (*i.e.*,

the transient phase). In fact, the throughput considerably increases when the transient phase is over, due to the smaller RTT values existing in the optical path. The smaller RTT value allowed the sender to have a higher transmission rate.

Differently of Scenario A, we have perceived some considerable oscillations in the throughput of a flow when considering the transmission capacity of the sender access link (Scenario B) and the capacity of the backbone links (Scenario C) as limiting factors. Both scenarios resulted in oscillations in the throughput during the transient phase due to the reordering of packets in the receiver. In spite of this oscillation, the TCP throughput recovers relatively fast from this oscillation after the transient phase is over.

More importantly, when the receiver local link is the bottleneck (Scenario D), we founded huge impact on the performance of the TCP throughput. A significant drop in the transmission rate was observed during the transient phase due to large volumes of data arriving at the input of the receiver's local link. When high rates (*e.g.*, 10 Gbps) are received in both optical and IP paths, the router's queue at the receiver's side (r_3 in Figure 5.1) is rapidly filled and packets are dropped due to lack of queue space. It is worth emphasizing that the decrease in the throughput is not only due to reordering, but mainly due to massive number of discarded packets, which was not observed in the previous scenarios. The results in Scenario D suggest thus that the transmission capacity of the receiver local link and the routers buffer size should be regarded before moving flows on the fly.

5.2 Additional aspects

IN this section, we present some additional aspects to be regarded when self-management of lightpaths is used. The aspects presented in this section are with respect to lightpath capacity estimation, routing, atomicity, restoration, and security, respectively. Even though these aspects have already been widely researched within the context of IP networks, the advent of self-management on hybrid networks may bring new challenges, therefore they should be reconsidered. It is worth highlighting that we do not aim in this thesis at proposing any improvement for these aspects, but mainly at informing the reader about them.

5.2.1 Estimation of lightpath capacity

Once a flow is elected to be moved to the optical level, a lightpath needs to be allocated for this flow. Lightpaths have different capacities, which are defined by the today's standards SONET [5] and SDH [79]. Based on the rates supported by SONET

and SDH (Subsection 1.1.2), our self-management system has to decide what transmission rate is more appropriated to be allocated for an elected IP flow. As presented in Subsection 4.4.1, most of the large flows present a constant throughput, but some of them may vary their throughput over time (Figure 4.4(b)). Hence, for a proper allocation of a lightpath, not only the flow average throughput should be regarded, but also its variance. By considering variance, a safety margin in the allocation of a lightpath is considered, which decreases the chances that a lightpath runs into congestion situations.

Even if the calculation of the variance is not difficult by itself, the acquisition of network information that gives the means to calculate it is more challenging. In order to ease this calculation, most network managers usually trust monitoring tools, such as NetFlow, to first find out the average flow rate and then apply a safety margin (for example 30%) over the current rate in order to consider variance.

However, this safety margin is rather simplistic and not always correctly infers the right lightpath capacity to be allocated. Van de Meent [154] performed a study on network link dimensioning, in which he suggests two approaches to infer link capacity while considering variance estimation: a direct approach and an indirect one. The direct approach consists of measuring a link rate at regular time intervals and then calculating traffic variance by applying the standard sample variance estimator on the measured traffic rate values. At the other hand, the indirect approach does not need regular time intervals to calculate the variance. According to Van de Meent, his proposed indirect approach exploits the relationship between traffic variance and the occupancy of the buffer in front of the link to be dimensioned. By taking snapshots of the buffer occupancy, the variance can be inferred based on the buffer content's distribution. For further details, see [154].

Despite the fact that Van de Meent's indirect approach is suitable to estimate link capacities, his approach is not suitable to make the same estimation based on individual flows. The reason for that comes from the fact that the finest granularity considered by his indirect approach is at the link (interface) level, whereas our finest granularity is at the flow level. In contrast to the indirect approach, the direct approach is more generic and can be used to estimate capacities based on both links and individual flows. Large flows can be evaluated by our decision maker, since large flows are regularly reported (active flow timeout) to our decision maker. Based on this regularity, the decision maker may estimate throughput and variance of individual flows and therefore decide on a proper lightpath allocation. The use of a direct approach is thus more appropriate than the indirect approach to estimate lightpath capacity for the elected flows.

Even when considering variance, a lightpath can still run into congestion due to the nature of flows. We see two possibilities to deal with that: 1) packets are

dropped in an attempt to make the sender to slow down its transmission or 2) the decision maker tries to allocate a bigger lightpath, before a congestion takes place.

5.2.2 Routing

Not only a proper lightpath capacity estimation is essential for an optimum use of the optical level, but also a proper path selection between the end-points of a flow communication. In order to select the best path, routing metrics, such as path load or delay, are applied on paths. Then routing algorithms, such as Enhanced Interior Gateway Routing Protocol (EIGRP) [30], take these metrics into account to select paths in a network.

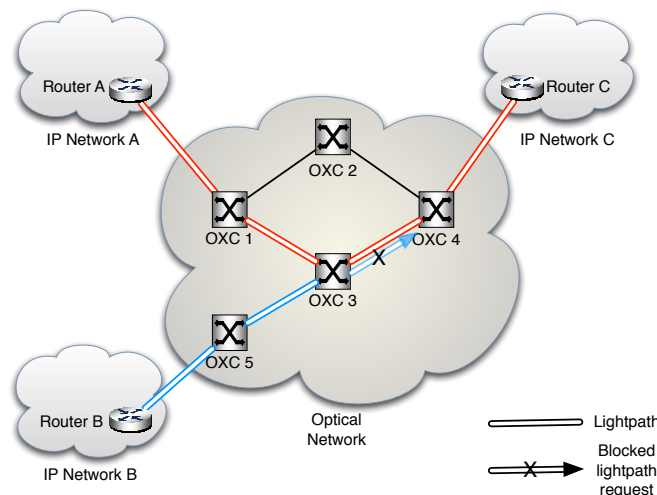


Figure 5.10: Example of a blocking request situation.

Within the context of our research, a routing algorithm should choose a path primarily based on the lightpath capacity required for a flow. However, it should also choose a path that results in a lower blocking probability to other flows eligible to join the optical level. For instance, Figure 5.10 shows one example of a subsequent lightpath request that was denied due to lack of resources (lightpath capacity).

Router A is connected to Router C through a lightpath that extends across the OXCs 1, 3, and 4. Router B tries to request a lightpath in order to be connected to Router C as well. However, the lightpath segment between OXC 3 and OXC 4 cannot be established due to lack of resources. This is one example of a bad routing

decision when selecting lightpaths. If the routing decision process would have chosen a lightpath between Router A and C across the OXCs 1, 2, and 4, the request of a lightpath for Router B could have been attended.

Different approaches can be used to select paths in a managed network. We narrow down these approaches into distributed and centralized. A distributed approach mostly relies on distributed routing protocols, such as distance-vector routing protocols and link-state routing protocols. The main goal of any routing protocol is to dynamically communicate information about all network paths used to reach a destination and to select, among those paths, the best one to reach a destination network. Some examples of routing protocols are OSPF [110], IS-IS [113], and EIGRP [30].

When centralized, a single management station is in charge of maintaining the full network map. This requires network nodes to send updates about their link states to a single point rather than to the entire network. A centralized network map can therefore be considered simpler to implement. Moreover, with an entire topology located in a single place, faster paths calculation can be reached. The main disadvantage of centralizing the routing decision process is with regard to a central point of failure. If the management station is down, the routing decision process is unable to attend request for lightpaths.

5.2.3 Atomicity

Another consideration to be taken is regarding atomicity in the lightpaths setup, *i.e.*, either all segments along the desired lightpath are established or none of them [15]. Atomicity is important in order to guarantee a proper full end-to-end communication. If some event whatsoever (*e.g.*, lack of resources) occurs during the setup stage, a roll-back action shall be taken in order to put the hybrid network back to its state before the setup attempt.

In Chapter 2, we presented the conventional approaches to manage hybrid networks: direct management (Subsection 2.1.1) and indirect management (Subsection 2.1.2). As stated in Chapter 2, network operators are directly involved in both management approaches, but with a different degree of responsibility in the specific act of establishing and releasing lightpaths.

In the direct management approach, network operators use well-established network management technologies, such as SNMP in order to setup lightpaths on every device along the desired path. The responsibility of assuring atomicity is therefore also up to network operators. If by any chance it is not possible to set up a lightpath segment, network operators have to manually roll-back all segments previously established.

On its turn, in the indirect management approach, network operators use GMPLS to establish and release lightpaths. Network operators are in charge of triggering the establishment and release actions while GMPLS executes these actions. The path between two end-points can be explicitly defined by network operators or leave it up to GMPLS to find the best choice. The main advantage of using GMPLS to setup lightpaths is that whether any error occurs, GMPLS does the roll-back automatically by the exchange of RSVP-TE control messages [51].

5.2.4 Restoration

Even though atomicity and restoration can be related, they are not completely the same. The main difference between atomicity and restoration is that the former recovers from a failure during the establishment of a lightpath while the latter reacts to failures in a lightpath already established.

Similar to the act of rolling-back a failed attempt to set up a lightpath, it is up to network operators to manually restore an established lightpath. This is usually done by isolating the faulty node or link and rerouting to an alternative path.

In case of using GMPLS, the restoration process conventionally employs dynamic resource establishment and may require dynamic route calculation as well in order to overcome an error situation [80]. This can be performed either at an intermediate node, where a new segment is calculated from that node to the destination end-point, or at the source node, where a new path is calculated from the source to the destination.

The main advantage of using GMPLS is that it makes the restoring process simpler. GMPLS has the capability of detecting a failure, notifying it, and recovering from it automatically. In the event of failure, a new lightpath segment is established around the failure. This new segment may be calculated on the fly, or it may be pre-preserved as a secondary segment. The main advantage of considering the latter is that there is a certain guarantee that an alternative segment will be quickly available when needed in the event of a failure.

5.2.5 Network security

Network security consists of providing means to protect network resources from unauthorized accesses or malicious activities. Consistent and continuous monitoring of network activity is important to prevent or detect any misuse that may fall upon a managed network.

Within the context of our self-management proposal, network security should be regarded to prevent any inappropriate use of lightpaths. For instance, DoS attacks

that may be transiting at the IP level should be unable to be moved over lightpaths. The reason from that comes from the fact that by unintentionally offloading such attacks to the optical level, the scale of these attacks can become larger.

Security concepts, such as authorization and intrusion detection should be considered in order to tighten security in hybrid networks that employ self-management of lightpaths. By doing so, malicious flows could be verified prior to the offload to the optical level. If there is any suspicion that such flows are behaving differently from the expected, their offload over lightpaths could be blocked and their behavior could be logged for audit purposes and for later high level analysis.

5.3 Summary

WE have presented in this section some important aspects to be considered when employing self-management of lightpaths in hybrid optical and packet switching networks. We gave special focus on the impact that moving flows on the fly has on the performance of TCP (CUBIC) throughput (Section 5.1). In general, TCP is able to recover relatively fast from the moment that TCP flows are moved to the optical level (the transient phase). Some oscillations were observed in the throughput performance due to packet reordering, but they did not present major problem for the TCP throughput. However, when the transmission capacity of the receiver local link becomes the bottleneck, the receiving of high rates from both optical and IP paths makes the router's queue at the receiver's side to be rapidly filled up and packets are dropped due to lack of queue space. As a result, TCP throughput suffers major degradation of performance. Therefore, the transmission capacity of the receiver local link and the routers buffer size should be regarded before moving flows on the fly.

Some additional aspects were also highlighted in Section 5.2. Even though these aspects have been extensively researched within the scope of IP networks, the introduction of our self-management proposal may bring some new challenges. We did not present any improvement for these aspects, but we raised the concern that they should be reconsidered with the advent of our self-management of lightpaths on hybrid optical and packet switching networks.

Chapter 6

Conclusions

This chapter presents the conclusions for the research presented in this thesis. Here, we provide the answers for our research questions introduced in Section 1.4 as well as some suggestions for future research.

The organization of the present chapter is as follows:

- *Section 6.1 presents an overall conclusion of the research presented in this thesis as well as the answers for our research questions.*
- *Section 6.2 finalizes this thesis by giving pointers for further research.*

6.1 Overall conclusion

IN this thesis, we presented how self-management of lightpaths in hybrid optical and packet switching networks can be implemented. We started this thesis by posing our high level research question, namely “*Is the idea of self-management of hybrid optical and packet switching networks technically feasible for the future Internet?*” We then refined this high level question into several subquestions, which we provide now the answers as follows:

Research question (1): what is the state-of-the-art in the management of hybrid networks?

As presented in Chapter 2, two main approaches are conventionally used to manage current hybrid networks, namely direct and indirect management approaches. Both approaches depend on human intervention to cope with hybrid devices and lightpaths, but with different degrees of dependability. In the direct management approach, the human manager uses well-known management interfaces and protocols, such as CLI, SNMP, and TL1 in order to directly manage every hybrid device along a desired lightpath. In its turn, in the indirect management approach, the human manager is in charge of only triggering the actions of establishing and

releasing lightpaths. Meanwhile, hybrid devices execute such actions by coordinating themselves through the exchange of signaling messages. The most well-known technology for that in hybrid optical and packet switching networks is GMPLS. We concluded that these conventional management approaches have certain shortcomings, such as lengthy configuration process and heavy dependence on human intervention to select and move IP flows to the optical level as well as to establish and release lightpaths. This intervention can therefore take a considerable amount of time to be performed. Some paradigms (OBS and OPS) have been proposed to speed up the setup process, but they are still in their experimental stage and it may still take a considerable amount of time for them to be fully deployed on what we foresee as a future optical Internet.

Research question (2): How can the monitoring of IP flows be performed?

In Chapter 3, we presented three possible techniques to monitor IP traffic on current hybrid networks, namely *packet-based techniques*, *SNMP-based techniques*, and *flow-based techniques*. Each technique presents pros and cons when used in high-speed hybrid networks as detailed in Section 3.4. The choice in this thesis for a flow-based monitoring technique is due to its wide usage on current networks as well as due to its reliability to report network information. We also considered the use of traffic sampling in the monitoring processes of high-speed networks since traffic sampling is employed to further decrease the amount of processed data as well as to reduce the consumption of storage and processing power.

Research question (3): How can autonomic decisions be made?

Autonomic decisions can be taken based on a specific objective. As presented in Chapter 4, our autonomic decision process has as main objective to move the biggest elephant flows with high survival probability from the IP level to the optical level. In order for our decision process to do that, we instruct our decision process with the assumptions that: (1) most of large flows present a constant throughput, and (2) the longer a large flow has lasted, the higher its survival probability is. These assumptions were proved to be true, as shown in Section 4.4. In Section 4.5, we validated our autonomic decision approach by showing that such assumptions allow a good level of accuracy to select large flows to the optical level when compared with the best theoretical result. Lastly, we also gave some attention to different grooming strategies in order to accommodate flows over lightpaths (Section 4.6). In our investigation, three strategies were considered, namely *dedicated strategy*, *spreading*

strategy, and *packing strategy*. Our analysis showed that when using the *packing strategy*, the lightpaths are more efficiently used.

Research question (4): What are the side effects of moving IP flows to the optical level on the fly?

In Section 5.1, we observed the impact of moving flows to the optical level on the fly. We chose to observe this impact on TCP flows due to the facts TCP is a highly representative protocol (represent most of the traffic in current networks) as well as it is connection-oriented, which means that it is more complex to cope with errors than connectionless protocols. The result of our analysis showed that in general TCP flows react quickly to the changes imposed by the move on the fly (that is, during the transient phase). Some oscillations in the TCP throughput are observed during the transient phase, but an increase in the flow transmission rate is seen afterwards. However, when the receiver local link is the bottleneck, high rates received in both optical and IP paths can make the routers queue at the receivers side to be rapidly filled up. As a result, packets are dropped due to lack of queue space. It is worth highlighting although that this bottleneck is more a network infrastructure issue than an issue when moving flows on the fly. It is strongly suggested therefore that the transmission capacity of the receiver local link and the routers buffer size should be regarded before moving flows on the fly.

Overall conclusion: Based on the research work presented in thesis, our main conclusion is that our self-management proposal for hybrid optical and packet switching networks is technically feasible to be deployed in the near future. The reason for that comes from the fact our self-management proposal takes into account technologies and infrastructures that are available today. Current technologies, such as NetFlow (Subsection 3.4.4) and GMPLS (Subsection 2.1.2) play an important role in the monitoring and configuring aspects of our self-management proposal. Equally important, hybrid optical and packet switching infrastructures are also a reality as it is the case of SURFnet6 (Subsection 1.1.1). Moreover, we highlight that our self-management proposal does not require the introduction of any new protocol or the modification of today's network layers. However, it is important to say that it is still early to say whether our self-management proposal is the best approach. Additional technical aspects (as presented in Section 5.2) as well as non-technical aspects, such as economical issues, should be regarded prior to a full implementation of our proposal. We believe although that this thesis is a stimulating first step towards the employment of self-management principles on current (and future) hybrid optical and packet switching networks.

6.2 Future research

WE suggest here some possibilities for further research as follows:

- We presented in this thesis the feasibility of using our self-management proposal on hybrid networks as well as how this proposal can be implemented. It would be interesting therefore that an implementation would be made in order to evaluate our self-management proposal in an experimental hybrid optical and packet switching network.
- Our self-management approach has been designed considering an intra-domain network. We have not considered its usage in inter-domain networks. We believe that the use of self-management in an inter-domain scenario can be more challenging. For instance, the self-management approach may have to deal with different network policies or different business models, which become the act of managing lightpaths in different domains more complicated. Moreover, the self-management approach may have to cope with a greater number of devices spanning through several domains, which may not scale well. An investigation about how to employ our self-management approach in a multi-domain environment would be therefore interesting.
- As stated in Section 2.4, our self-management approach is designed to take decisions in a centralized way. Further research on distributing the decisions over the hybrid network (the hybrid devices) would be interesting. As generally known, the use of a centralized approach can lead the management system to get overloaded. On the other hand, the use of a distributed approach may be less efficient and more complex to implement. The investigation of a distributed approach is therefore a compelling future research.
- In this thesis, we focused on the impact of moving flows on the fly on the TCP throughput. Nonetheless, other aspects should be regarded as well (Section 5.2). For instance, by employing our self-management proposal, there is a possibility of moving DoS attacks over lightpaths. This could result in speeding up attacks over the Web and even making them stronger, since with large bandwidth more packets could be sent to bring down a server on the Web.
- Last but not least, we believe that other flow definitions may have a certain influence on the amount of traffic offloaded to the optical level as well as on the way flows are accommodated over lightpaths. Investigating therefore the effects of other definitions for a flow is advised.

Appendix A

Statistical & mathematical background information

The purpose of this appendix is to present more details about statistical and mathematical methods applied on this thesis.

A.1 Decision trees

DECISION trees can be defined as a predictive model; *i.e.*, a mapping from observations about an item to conclusions about its target value (the dependent variable). Decision trees can be divided into **classification trees** (in which the dependent variable has a discrete outcome) or **regression trees** (in which the dependent variable has a continuous outcome). Independent of the kind of decision tree, their elements are summarized in *leaves* and *branches*. *Leaves* represent classifications, whereas *branches* represent conjunctions of features that lead to those classifications.

Regression trees

If the dependent variable (target value) is a continuous value, then a regression tree is employed. It is worth mentioning that when using a regression tree to predict the value of a dependent variable, the mean value of the dependent variable in leaf (a node of the tree) is the estimated value. An example of a regression tree is shown in Figure A.1. In this example, the dependent variable is the Heat Index (H.I.)¹ and the independent variables are air temperature (T) and relative humidity (H).

In this tree, we see that the value of the Heat Index is on average 30 °C. However, the average Heat Index increases as the air temperature and relative humidity increase too. For example, with an air temperature above or equal to 43 °C and a relative humidity above or equal to 40%, the Heat Index is on average 58 °C. On the other side, as the air temperature and relative humidity decrease, so the Heat Index

¹The Heat Index is an index that combines air temperature and relative humidity in order to determine the human-perceived equivalent temperature, commonly referred as felt air temperature.

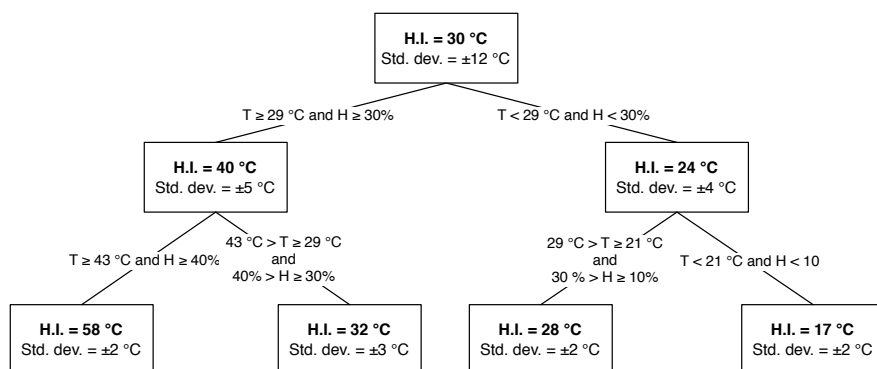


Figure A.1: Regression tree.

too. A regression tree can thus provides continuous values to the dependent variable (Heat Index) based on some predictors (air temperature and relative humidity).

Classification trees

On its turn, if the dependent variable is categorical, then the use of a classification tree is more appropriated. Instead of trying to estimate a continuous value for the dependent variable, the classification tree uses the values of the predictor variables

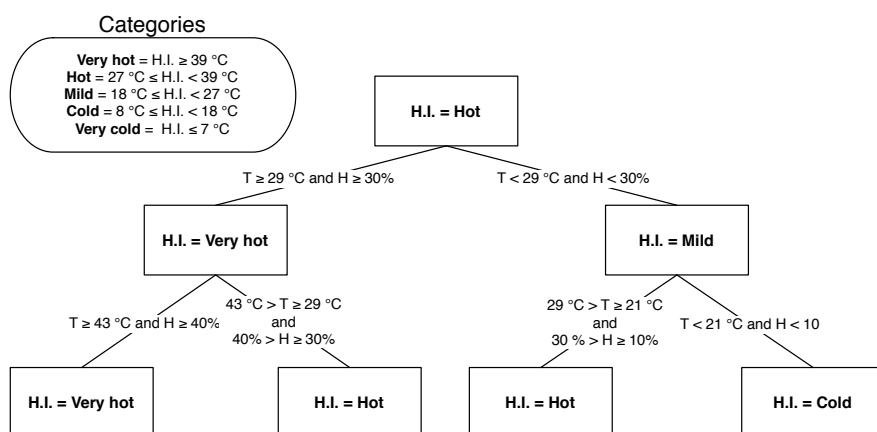


Figure A.2: Classification tree.

to move through the tree until it reaches a terminal node. When the terminal node is reached, the classification tree attempts to predict what category is more suitable for that node. An example of a classification tree is shown in Figure A.2. In this example, the dependent variable is still the Heat Index, but instead of predicting its temperature, the Heat Index is categorized. The categories can be manually defined, but optimizations can be performed in order to try to find the optimal category division.

A.2 The CHAID algorithm

THE CHAID algorithm was created by Kass [84]. Its main feature is a recursive partitioning method that relate a number of independent variables (also known as *predictors*) against a single dependent variable. At each step, CHAID chooses the independent variable that has the strongest relation with the dependent variable. For example, observations about wind speed, humidity, and temperature could be used as predictors for rain precipitation (dependent variable). The algorithm work as follows:

Step 1 - Prepare the predictors: CHAID begins by equally dividing the predictors into partitions with an even number of observations. This division comes from the statistical hypothesis Chi-square Goodness-of-Fit test [142], which is used to test if a sample of data comes from a population with a specified distribution. The Chi-square test initially divides a number of observations from a population into a number of partitions. The main reason for that is that hypothetically the number of observations in each partition would occur with the same frequency.

Step 2 - Cross-tabulates the predictor partitions: Once the observations are equally divided into partitions, CHAID algorithm first cross-tabulates (*i.e.*, it creates a contingency table) each partition of a predictor. Second, it performs a pairwise chi-square test of independence between the adjoining partitions in order to calculate levels of significance (p -values²). If the respective p -value for a given pair of predictor partitions is not statistically significant as defined by a certain α value (significance level of a test), then it will merge the respective predictor partitions and repeat this step. If the statistical significance (the p value) for the respective pair of predictor categories is significant (less than the respective α value), then CHAID will optionally compute a Bonferroni adjusted p -value for the set of partitions for the respective predictor.

²The p -value stands for the probability of a certain relationship being observed by chance. The smaller this value is, the more statistically significant the relationship is.

Step 3 - Compare the predictors: Once the predictors have been appropriately partitioned, CHAID algorithm cross-tabulates each predictor partitions against the dependent variable. Similar to step 2, CHAID algorithm performs a Chi-square test of independence in order to calculate levels of significance (p). The predictor which shows the smallest p value is then placed at the first depth of the CHAID tree along with its partitions.

Step 4 - Select subsequent tree levels: Subsequent to the decision on the first level predictor and its best merged partitions, CHAID begins to place other predictors beneath the initial predictor. In order to accomplish that, CHAID begins anew, only at each partition of the first level. As described in step 3, CHAID builds contingency tables of each independent variable versus the dependent variable, but the remaining independent variables all have the 1st level independent variable included. After CHAID has built all the crosstabs for each remaining predictors, each crosstab is tested for significance using chi-square test of independence, in the same manner discussed in step 2. The crosstabs which are insignificant are then omitted. Those crosstabs that are significant are placed beneath their respective partitions if they are the most significant for that partition.

Each of these partitions are independently reanalyzed till further sub-divisions cannot be performed. In this way, CHAID partitions the data into mutually exclusive, exhaustive subsets that best describe the dependent variable.

Bibliography

- [1] M. Abrams, C. R. Standridge, G. Abdulla, E. A. Fox, and S. Williams, "Removal policies in network caches for World-Wide Web documents," in *SIGCOMM '96: Conference proceedings on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 1996, pp. 293–305.
- [2] Alcatel-Lucent, "Alcatel-Lucent Bell Labs announces new optical transmission record and breaks 100 Petabit per second kilometer barrier," Nov. 2009. [Online]. Available: http://www.alcatel-lucent.com/wps/portal/newsreleases/detail?LMSG_CABINET=Docs_and_Resource_Ctr&LMSG_CONTENT_FILE=News_Releases_2009/News_Article.001797.xml&lu_lang_code=en
- [3] M. Allman, V. Paxson, and W. Stevens, "TCP Congestion Control," RFC 2581 (Proposed Standard), Internet Engineering Task Force, Apr. 1999. [Online]. Available: <http://www.ietf.org/rfc/rfc2581.txt>
- [4] L. Andersson and G. Swallow, "The Multiprotocol Label Switching (MPLS) Working Group decision on MPLS signaling protocols," RFC 3468 (Informational), Internet Engineering Task Force, Feb. 2003. [Online]. Available: <http://www.ietf.org/rfc/rfc3468.txt>
- [5] ANSI, "Synchronous Optical Network (SONET) - Basic Description including Multiplex Structure, Rates, and Formats," T1.105-2001, 1995, American National Standards Institute (ANSI).
- [6] G. Armitage, "MPLS: the magic behind the myths [multiprotocol label switching] ," *Communications Magazine, IEEE*, vol. 38, no. 1, pp. 124–131, Jan. 2000.
- [7] E. Asmare, A. Gopalan, M. Sloman, N. Dulay, and E. Lupu, "A Mission Management Framework for Unmanned Autonomous Vehicles," in *Mobile Wireless Middleware, Operating Systems, and Applications, Second International Conference (Mobilware)*, ser. ICST Lecture Notes, vol. 7, Apr. 2009, pp. 222–235.
- [8] E. Baaren, L. v. d. Wijngaert, and E. Huizer, "'Who's Afraid of High Def?' Institutional Factors Influencing HDTV Diffusion in the Netherlands," in *ICDS '08: Proceedings of the*

- Second International Conference on Digital Society.* Washington, DC, USA: IEEE Computer Society, 2008, pp. 42–48.
- [9] M. Baldi and Y. Ofek, "Time for a "Greener" Internet," in *1st International Workshop on Green Communications (GreenComm'09) in conjunction with the IEEE International Conference on Communications (IEEE ICC 2009), Dresden, Germany*, Jun. 2009, pp. 1–6.
- [10] A. Banerjee, J. Drake, J. Lang, B. Turner, K. Kompella, and Y. Rekhter, "Generalized Multiprotocol Label Switching: An Overview of Routing and Management Enhancements," *IEEE Communications Magazine*, vol. 39, no. 1, pp. 144–151, Jan. 2001.
- [11] T.-R. Banniza, D. Boettle, R. Klotsche, P. Schefczik, M. Soellner, and K. Wuenstel, "A european approach to a clean slate design for the future internet," *Bell Labs Technical Journal*, vol. 14, no. 2, pp. 5–22, 2009.
- [12] A. Bassi, S. Denazis, A. Galis, C. Fahy, M. Serrano, and J. Serrat, "Autonomic Internet: A Perspective for Future Internet Services Based on Autonomic Principles," in *2nd IEEE International Workshop on Modelling Autonomic Communications Environments (MACE 2007)*, San Jose, California, USA, 2007.
- [13] J.-M. Beaufils, "How Do Submarine Networks Web the World?" *Optical Fiber Technology*, vol. 6, no. 1, pp. 15 – 32, 2000.
- [14] L. A. Belady, "A study of replacement algorithms for a virtual-storage computer," *IBM Systems*, no. 5, pp. 78–101, 1966.
- [15] B. Berde, D. Papadimitriou, and M. Jager, "Traffic Engineering Element for GMPLS Networks," in *The 10th IEEE/IFIP Network Operations and Management Symposium, 2006 (NOMS 2006)*, Vancouver, Canada, Apr. 2006, pp. 1–4.
- [16] G. Bernstein, B. Rajagopalan, and D. Saha, *Optical Network Control: Architecture, Protocols, and Standards.* Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2003.
- [17] U. Black, *MPLS and Label Switching Networks.* Upper Saddle River, NJ, USA: Prentice Hall PTR, 2000.
- [18] D. Blumenthal, P. Prucnal, and J. Sauer, "Photonic packet switches: architectures and experimental implementations," *Proceedings of the IEEE*, vol. 82, no. 11, pp. 1650–1667, Nov. 1994.
- [19] N. Brownlee, "Traffic Flow Measurement: Meter MIB," RFC 2720 (Proposed Standard), Internet Engineering Task Force, Oct. 1999. [Online]. Available: <http://www.ietf.org/rfc/rfc2720.txt>
- [20] N. Brownlee and K. Claffy, "Understanding Internet traffic streams: dragonflies and tortoises," *Communications Magazine, IEEE*, vol. 40, no. 10, pp. 110–117, Oct. 2002.
- [21] P. Cao and S. Irani, "Cost-aware WWW proxy caching algorithms," in *USITS'97: Proceedings of the USENIX Symposium on Internet Technologies and Systems on USENIX Symposium on Internet Technologies and Systems.* Berkeley, CA, USA: USENIX Association, 1997, pp. 18–18.

- [22] I. Cerutti and A. Fumagalli, "Traffic grooming in static wavelength division multiplexing networks," *Communications Magazine, IEEE*, vol. 43, no. 1, pp. 101–107, Jan. 2005.
- [23] D. Chakraborty, A. Ashir, T. Suganuma, G. M. Keeni, T. K. Roy, and N. Shiratori, "Self-similar and fractal nature of Internet traffic," *International Journal of Network Management*, vol. 14, no. 2, pp. 119–129, 2004.
- [24] K. chan Lan and J. Heidemann, "A measurement study of correlations of Internet flow characteristics," *Computer Networks*, vol. 50, no. 1, pp. 46 – 62, 2006.
- [25] S. A. Chaudhry, A. H. Akbar, K.-H. Kim, S.-K. Hong, and W.-S. Yoon, "HYWINMARC: An Autonomic Management Architecture for Hybrid Wireless Networks," in *Emerging Directions in Embedded and Ubiquitous Computing, EUC 2006 Workshops*, ser. Lecture Notes in Computer Science, vol. 4097. Berlin, Heidelberg: Springer-Verlag, Aug. 2006, pp. 193–202.
- [26] L. Cheng, A. Galis, B. Mathieu, K. Jean, R. Ocampo, L. Mamatras, J. Rubio-Loyola, J. Serfat, A. Berl, H. de Meer, S. Davy, Z. Movahedi, and L. Lefèvre, "Self-organising Management Overlays for Future Internet Services," in *3rd IEEE International Workshop on Modelling Autonomic Communications Environments (MACE 2008)*, 2008, pp. 74–89.
- [27] L. Cherkasova and G. Ciardo, "Role of Aging, Frequency, and Size in Web Cache Replacement Policies," in *HPCN Europe 2001: Proceedings of the 9th International Conference on High-Performance Computing and Networking*. London, UK: Springer-Verlag, 2001, pp. 114–123.
- [28] L. Ciarletta, S. Piekarec, A. Aghasaryan, H. Pouyllau, S. Haar, E. Fabre, and N. Mbarek, "Multi-Domain Self Aware Management : Negotiation and Monitoring," in *13th International Conference on Telecommunications - ICT 2006*, Funchal, Madeira island/Portugal, May 2006.
- [29] Cisco Systems, *Internetworking Technologies Handbook, Fourth Edition*. Cisco Press, 2003.
- [30] Cisco Systems, "Enhanced Interior Gateway Routing Protocol," Sep. 2005. [Online]. Available: http://www.cisco.com/en/US/tech/tk365/technologies_whitepaper09186a0080094cb7.shtml
- [31] Cisco Systems, "Introduction to Cisco IOS NetFlow - A Tehnical Overview," Oct. 2007. [Online]. Available: http://www.cisco.com/en/US/products/ps6601/prod_white_papers_list.html
- [32] Cisco Systems, "Flexible NetFlow," May 2009. [Online]. Available: <http://www.cisco.com/web/go/fnf>
- [33] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954 (Informational), Oct. 2004. [Online]. Available: <http://www.ietf.org/rfc/rfc3954.txt>
- [34] B. Claise, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," RFC 5101 (Proposed Standard), Internet Engineering Task Force, Jan. 2008. [Online]. Available: <http://www.ietf.org/rfc/rfc5101.txt>

- [35] M. Crovella, "Performance Evaluation with Heavy Tailed Distributions," in *JSSPP '01: Revised Papers from the 7th International Workshop on Job Scheduling Strategies for Parallel Processing*. London, UK: Springer-Verlag, 2001, pp. 1–10.
- [36] M. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: evidence and possible causes," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 835–846, Dec 1997.
- [37] S. Dar, M. J. Franklin, B. T. Jónsson, D. Srivastava, and M. Tan, "Semantic Data Caching and Replacement," in *VLDB '96: Proceedings of the 22th International Conference on Very Large Data Bases*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1996, pp. 330–341.
- [38] C. de Laat, E. Radius, and S. Wallace, "The Rationale of the Current Optical Networking Initiatives," *Future Generation Computer Systems*, vol. 19, no. 6, pp. 999–1008, 2003.
- [39] M. de Vos, "LOFAR: the first of a new generation of radio telescopes," in *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, vol. 5, Mar. 2005, pp. 865–868.
- [40] H. Derbel, N. Agoulmine, and M. Salaün, "ANEMA: Autonomic network management architecture to support self-configuration and self-optimization in IP networks," *Computer Networks*, vol. 53, no. 3, pp. 418–430, 2009.
- [41] Distributed Management Task Force, Inc., "CIM-XML," May 2009. [Online]. Available: <http://www.dmtf.org/standards/wbem/CIM-XML>
- [42] Distributed Management Task Force, Inc., "Common Information Model (CIM) Standards," May 2009. [Online]. Available: <http://www.dmtf.org/standards/cim/>
- [43] Distributed Management Task Force, Inc., "Distributed Management Task Force," May 2009. [Online]. Available: <http://www.dmtf.org/>
- [44] Distributed Management Task Force, Inc., "Specification for CIM Operations over HTTP," May 2009. [Online]. Available: <http://www.dmtf.org/standards/documents/WBEM/DSP200.html>
- [45] Distributed Management Task Force, Inc., "Web services for management," May 2009. [Online]. Available: <http://www.dmtf.org/standards/wsman/>
- [46] S. Dobson, S. Denazis, A. Fernández, D. Gaïti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, and F. Zambonelli, "A survey of autonomic communications," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 1, no. 2, pp. 223–259, 2006.
- [47] N. Duffield, D. Chiou, B. Claise, A. Greenberg, M. Grossglauser, and J. Rexford, "A Framework for Packet Selection and Reporting," RFC 5474 (Informational), Internet Engineering Task Force, Mar. 2009. [Online]. Available: <http://www.ietf.org/rfc/rfc5474.txt>
- [48] R. Enns, "NETCONF Configuration Protocol," RFC 4741 (Proposed Standard), Internet Engineering Task Force, Dec. 2006. [Online]. Available: <http://www.ietf.org/rfc/rfc4741.txt>

- [49] C. Estan, "Internet traffic measurement: what's going on in my network?" Ph.D. dissertation, University of California, San Diego, 2003.
- [50] L. Fallon, D. Parker, M. Zach, M. Leitner, and S. Collins, "Self-forming Network Management Topologies in the Madeira Management System," in *Proceedings of the 1st international conference on Autonomous Infrastructure, Management and Security (AIMS 2007)*, Oslo, Norway. Berlin, Heidelberg: Springer-Verlag, July 2007, pp. 61–72.
- [51] A. Farrel, A. Ayyangar, and J. Vasseur, "Inter-Domain MPLS and GMPLS Traffic Engineering – Resource Reservation Protocol-Traffic Engineering (RSVP-TE) Extensions," RFC 5151 (Proposed Standard), Feb. 2008. [Online]. Available: <http://www.ietf.org/rfc/rfc5151.txt>
- [52] A. Feldmann, "Internet clean-slate design: what and why?" *SIGCOMM Computer Communication Review*, vol. 37, no. 3, pp. 59–64, 2007.
- [53] B. Fenner and J. Flick, "Management Information Base for the User Datagram Protocol (UDP)," RFC 4113 (Proposed Standard), Internet Engineering Task Force, Jun. 2005. [Online]. Available: <http://www.ietf.org/rfc/rfc4113.txt>
- [54] T. Fioreze, L. Granville, A. Pras, A. Sperotto, and R. Sadre, "Self-Management of Hybrid Networks: Can We Trust NetFlow Data?" in *11th IFIP/IEEE International Symposium on Integrated Network Management (IM 2009)*, Long Island, New York, USA. Piscataway: IEEE Computer Society Press, Jun. 2009, pp. 577–584.
- [55] T. Fioreze, L. Granville, R. Sadre, and A. Pras, "A Statistical Analysis of Network Parameters for the Self-management of Lambda-Connections," in *Proceedings of the 3rd International Conference on Autonomous Infrastructure, Management and Security (AIMS 2009)*, Enschede, The Netherlands, ser. Lecture Notes in Computer Science, vol. 5637. Berlin: Springer Verlag, Jun. 2009, pp. 15–27.
- [56] T. Fioreze, R. van de Meent, and A. Pras, "An Architecture for the Self-management of Lambda-Connections in Hybrid Networks," in *13th EUNICE Open European Summer School and IFIP TC6.6 Workshop on Dependable and Adaptable Networks and Services (EUNICE 2007)*, Enschede, The Netherlands, ser. Lecture Notes in Computer Science, vol. 4606. Germany: Springer Verlag, May 2007, pp. 141–148.
- [57] T. Fioreze, M. O. Wolbers, R. van de Meent, and A. Pras, "Finding Elephant Flows for Optical Networks," in *Application session proceeding of the 10th IFIP/IEEE International Symposium on Integrated Network Management (IM 2007)*, Munich, Germany. Piscataway: IEEE Computer Society Press, May 2007, pp. 627–640.
- [58] T. Fioreze, M. O. Wolbers, R. van de Meent, and A. Pras, "Characterization of IP Flows Eligible for Lambda-Connections in Optical Networks," in *Proceedings of the 11th IEEE/IFIP Network Operations & Management Symposium (NOMS 2008)*, Salvador, Bahia, Brazil. Piscataway: IEEE Computer Society Press, Apr. 2008, pp. 256–262.
- [59] T. Fioreze, M. O. Wolbers, R. van de Meent, and A. Pras, "Offloading IP Flows onto Lambda-Connections," in *Proceedings of the 18th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management, (DSOM 2007)*, San Jose, USA, ser. Lecture Notes in Computer Science, vol. 4785. Berlin: Springer Verlag, Aug. 2007, pp. 183–186.

- [60] G. Fisk, M. Fisk, C. Papadopoulos, and J. Neil, "Eliminating Steganography in Internet Traffic with Active Wardens," in *IH '02: Revised Papers from the 5th International Workshop on Information Hiding*. London, UK: Springer-Verlag, 2003, pp. 18–35.
- [61] S. Floyd, "HighSpeed TCP for Large Congestion Windows," RFC 3649 (Experimental), Internet Engineering Task Force, Dec. 2003. [Online]. Available: <http://www.ietf.org/rfc/rfc3649.txt>
- [62] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and S. Diot, "Packet-level traffic measurements from the Sprint IP backbone," *IEEE Network Magazine*, vol. 17, no. 6, pp. 6–16, Nov.-Dec. 2003.
- [63] GÉANT, "Géant," May 2009. [Online]. Available: <http://www.geant.net>
- [64] J. R. Goodman, "Using cache memory to reduce processor-memory traffic," in *ISCA '98: 25 years of the international symposia on Computer architecture (selected papers)*. New York, NY, USA: ACM, 1998, pp. 255–262.
- [65] L. Z. Granville, D. M. da Rosa, A. Panisson, C. Melchior, M. J. B. Almeida, and L. M. R. Tarouco, "Managing computer networks using peer-to-peer technologies," *Communications Magazine, IEEE*, vol. 43, no. 10, pp. 62–68, Oct. 2005.
- [66] A. Gupta, J. Hennessy, K. Gharachorloo, T. Mowry, and W.-D. Weber, "Comparative evaluation of latency reducing and tolerating techniques," in *ISCA '91: Proceedings of the 18th annual international symposium on Computer architecture*. New York, NY, USA: ACM, 1991, pp. 254–263.
- [67] S. Ha, I. Rhee, and L. Xu, "CUBIC: a new TCP-friendly high-speed TCP variant," *SIGOPS Operating Systems Review*, vol. 42, no. 5, pp. 64–74, 2008.
- [68] B. Haverkort, R. El Abdouni Khayari, and R. Sadre, "A Class-Based Least-Recently Used Caching Algorithm for World-Wide Web Proxies," in *Computer Performance Evaluations, Modelling Techniques and Tools. 13th International Conference, TOOLS 2003*, ser. Lecture Notes in Computer Science, vol. 2794. Springer, 2003, pp. 273–290.
- [69] E. Hernandez-Valencia, "Hybrid Transport Solutions for TDM/Data Networking Services," *IEEE Communications Magazine*, vol. 40, no. 5, pp. 104–112, May 2002.
- [70] B. Hoanca, "DWDM Fundamentals, Components, and Applications," *Journal of Optical Networking*, vol. 1, no. 5, pp. 184–185, 2002.
- [71] P. Horn, "Autonomic computing: IBM's Perspective on the State of Information Technology," 2001. [Online]. Available: <http://www.research.ibm.com/autonomic/manifesto/autonomic.computing.pdf>
- [72] IETF, "Multiprotocol Label Switching IETF Working Group," May 2009, Internet Engineering Task Force (IETF). [Online]. Available: <http://www.ietf.org/dyn/wg/charter/mpls-charter.html>
- [73] H. Imaizumi and H. Morikawa, "Directions towards Future Green Internet," in *The 12th International Symposium on Wireless Personal Multimedia Communications (WPMC'09)*, Sendai, Japan, Sep. 2009.

- [74] Internet Society, "Internet Engineering Task Force," May 2009. [Online]. Available: <http://www.ietf.org/>
- [75] Internet2, "Internet2 homepage," May 2009. [Online]. Available: <http://www.internet2.edu>
- [76] ISO, "Information processing systems - Open Systems Interconnection, Specification of Abstract Syntax Notation One (ASN.1)," International Standard 8824, Dec. 1987, International Organization for Standardization (ISO).
- [77] ISO, "Information processing systems - Open Systems Interconnection, Specification of Basic Encoding Rules for Abstract Notation One (ASN.1)," International Standard 8825, Dec. 1987, International Organization for Standardization (ISO).
- [78] ISO, "Information technology – Open Systems Interconnection – Systems management overview," ISO/IEC 10040:1998, 1998, International Organization for Standardization (ISO).
- [79] ITU, "Network Node Interface for the Synchronous Digital Hierarchy (SDH)," Recommendation G.707, 2000, ITU-Telecommunication Standardization Sector.
- [80] A. Jajszczyk and P. Rózycki, "Recovery of the Control Plane after Failures in ASON/GMPLS Networks," *IEEE Network*, vol. 20, no. 1, pp. 4–10, 2006.
- [81] B. Jennings, S. van der Meer, S. Balasubramaniam, D. Botvich, M. Foghlu, W. Donnelly, and J. Strassner, "Towards autonomic management of communications networks," *Communications Magazine, IEEE*, vol. 45, no. 10, pp. 112–121, Oct. 2007.
- [82] C. Jin, D. Wei, and S. Low, "FAST TCP: motivation, architecture, algorithms, performance," in *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 4, Mar. 2004, pp. 2490–2501.
- [83] R. Karedla, J. S. Love, and B. G. Wherry, "Caching strategies to improve disk system performance," *Computer*, vol. 27, no. 3, pp. 38–46, 1994.
- [84] G. V. Kass, "An Exploratory Technique for Investigating Large Quantities of Categorical Data," *Applied Statistics*, vol. 29, no. 2, pp. 119–127, 1980.
- [85] T. Kelly, "Scalable TCP: improving performance in highspeed wide area networks," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 2, pp. 83–91, 2003.
- [86] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003.
- [87] M.-S. Kim, Y. J. Won, and J. W. Hong, "Characteristic analysis of internet traffic from the perspective of flows," *Computer Communications*, vol. 29, no. 10, pp. 1639–1652, 2006.
- [88] E. Lastdrager and A. Pras, "Consistency of Network Traffic Repositories: An Overview," in *Proceedings of the 3rd International Conference on Autonomous Infrastructure, Management and Security (AIMS 2009), Enschede, The Netherlands*, ser. Lecture Notes in Computer Science, vol. 5637. Berlin: Springer Verlag, Jun. 2009, pp. 173–178.

- [89] D. Leith and R. Shorten, "H-TCP: TCP for High-speed and Long-distance Networks," in *Proceedings of the 2nd International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet 2004)*, Argonne, Illinois, USA, Feb. 2004.
- [90] A. Leon-Garcia and I. Widjaja, *Communication Networks: Fundamental Concepts and Key Architectures*, 2nd ed. New York: McGraw-Hill Companies, 2003.
- [91] F. Li, N. Seddigh, B. Nandy, and D. Matute, "An Empirical Study of Today's Internet Traffic for Differentiated Services IP QoS," *IEEE Symposium on Computers and Communications*, p. 207, 2000.
- [92] A. Liakopoulos, A. Zafeiropoulos, A. Polyraakis, M. Grammatikou, J. M. González, M. Wodczak, and R. Chaparadza, "Monitoring Issues for Autonomic Networks: The EFIPSANS Vision," in *1st European Workshop on Mechanisms for the Future Internet*, Salzburg, Austria, Jul. 2008.
- [93] E. Lupu, N. Dulay, M. Sloman, J. Sventek, S. Heeps, S. Strowes, K. Twidle, S.-L. Keoh, and A. Schaeffer-Filho, "AMUSE: autonomic management of ubiquitous e-Health systems," *Concurrency and Computation: Practice and Experience*, vol. 20, no. 3, pp. 277–295, 2008.
- [94] M. Maier, *Optical Switching Networks*. Cambridge University Press, 2008.
- [95] F.-T. Man, "A Brief History of TL1," *Journal of Network and Systems Management*, vol. 7, no. 2, 1999.
- [96] E. Mannie, "Generalized Multi-Protocol Label Switching (GMPLS) Architecture," RFC 3945 (Proposed Standard), Oct. 2004. [Online]. Available: <http://www.ietf.org/rfc/rfc3945.txt>
- [97] Maplesoft, "Math Software for Engineers, Educators & Students," Apr. 2009. [Online]. Available: <http://www.maplesoft.com/>
- [98] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "TCP westwood: Bandwidth estimation for enhanced transport over wireless links," in *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2001, pp. 287–297.
- [99] K. McCloghrie and F. Kastenholz, "The Interfaces Group MIB," RFC 2863 (Draft Standard), Internet Engineering Task Force, Jun. 2000. [Online]. Available: <http://www.ietf.org/rfc/rfc2863.txt>
- [100] K. McCloghrie, D. Perkins, and J. Schönwälder, "Conformance Statements for SMIPv2," RFC 2580 (Standard), Internet Engineering Task Force, Apr. 1999. [Online]. Available: <http://www.ietf.org/rfc/rfc2580.txt>
- [101] E. McKean, Ed., *The New Oxford American Dictionary, Second Edition*. New York: Oxford University Press, 2005.
- [102] K. Meyer, M. Erlinger, J. Betser, C. Sunshine, G. Goldszmidt, and Y. Yemini, "Decentralizing Control and Intelligence in Network Management," in *Integrated Network Management*, 1995, pp. 4–16.

- [103] M. Miyazawa, K. Ogaki, and T. Otani, "Multi-layer network management system with dynamic control of MPLS/GMPLS LSPs based on IP flows," in *The 11th IEEE/IFIP Network Operations and Management Symposium, 2008 (NOMS 2008), Salvador, Brazil, Apr. 2008*, pp. 263–270.
- [104] M. Molina, A. Chiosi, S. D'Antonio, and G. Ventre, "Design principles and algorithms for effective high-speed IP flow monitoring," *Computer Communications*, vol. 29, no. 10, pp. 1653 – 1664, 2006.
- [105] D. C. Montgomery and G. C. Runger, *Applied Statistics and Probability for Engineers*, 4th ed. Wiley, Jun. 2006.
- [106] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring Internet denial-of-service activity," *ACM Transactions on Computer Systems*, vol. 24, no. 2, pp. 115–139, 2006.
- [107] D. S. Moore, *The Basic Practice of Statistics*, 4th ed. W. H. Freeman, 2006.
- [108] T. Mori, M. Uchida, R. Kawahara, J. Pan, and S. Goto, "Identifying Elephant Flows Through Periodically Sampled Packets," in *ACM SIGCOMM*, 2004, pp. 115–120.
- [109] H. Mouftah and P.-H. Ho, *Optical Networks Architecture and Survivability*. Boston: Kluwer Academic Publishers, 2002.
- [110] J. Moy, "OSPF Version 2," RFC 2328 (Standard), Apr. 1998. [Online]. Available: <http://www.ietf.org/rfc/rfc2328.txt>
- [111] C. Murthy and G. Mohan, *WDM Optical Networks: Concepts, Design and Algorithms*. Englewood Cliffs: Prentice Hall Publishers, 2001.
- [112] ns2, "The Network Simulator NS-2," <http://www.isi.edu/nsnam/ns/>.
- [113] D. Oran, "OSI IS-IS Intra-domain Routing Protocol," RFC 1142 (Informational), Feb. 1990. [Online]. Available: <http://www.ietf.org/rfc/rfc1142.txt>
- [114] J. Parekh, G. Kaiser, P. Gross, and G. Valetto, "Retrofitting Autonomic Capabilities onto Legacy Systems," *Cluster Computing*, vol. 9, no. 2, pp. 141–159, 2006.
- [115] K. Pearson, "Mathematical Contributions to the Theory of Evolution. III. Regression, Heredity and Panmixia," *Philosophical Transactions of the Royal Society*, vol. 187, pp. 253–318, 1896.
- [116] C. Pinart and G. Giralt, "On managing optical services in future control-plane-enabled IP/WDM networks," *Journal of Lightwave Technology*, vol. 23, no. 10, pp. 2868–2876, Oct. 2005.
- [117] A. Pras, J. Schönwälder, M. Burgess, O. Festor, G. M. Perez, R. Stadler, and B. Stiller, "Key Research Challenges in Network Management," *IEEE communications magazine*, vol. 45, no. 10, pp. 104–110, Oct. 2007.
- [118] A. Pras, "Network management architectures," Ph.D. dissertation, University of Twente, Enschede, February 1995.

- [119] C. Qiao and M. Yoo, "Optical burst switching (OBS) - a new paradigm for an optical Internet," *Journal of High Speed Networks*, vol. 8, no. 1, pp. 69–84, 1999.
- [120] J. Quittek, S. Bryant, B. Claise, P. Aitken, and J. Meyer, "Information Model for IP Flow Information Export," RFC 5102 (Proposed Standard), Jan. 2008. [Online]. Available: <http://www.ietf.org/rfc/rfc5102.txt>
- [121] J. Quittek, T. Zseby, B. Claise, and S. Zander, "Requirements for IP Flow Information Export (IPFIX)," RFC 3917 (Informational), Internet Engineering Task Force, Oct. 2004. [Online]. Available: <http://www.ietf.org/rfc/rfc3917.txt>
- [122] M. Rabinovich and O. Spatschek, *Web caching and replication*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2002.
- [123] P. Racz, F. Eyermann, and B. Stiller, "Statistical Traffic Sampling Methods for Network Management and Inter-Domain Charging," in *10th EUNICE Open European Summer School (EUNICE 2004)*, Tampere, Finland, Jun. 2004.
- [124] R. Raghunathan, "Management Information Base for the Transmission Control Protocol (TCP)," RFC 4022 (Proposed Standard), Internet Engineering Task Force, Mar. 2005. [Online]. Available: <http://www.ietf.org/rfc/rfc4022.txt>
- [125] B. Rajagopalan, J. Luciani, and D. Awduche, "IP over Optical Networks: A Framework," RFC 3717 (Informational), Mar. 2004. [Online]. Available: <http://www.ietf.org/rfc/rfc3717.txt>
- [126] M. Rose and K. McCloghrie, "Concise MIB definitions," RFC 1212 (Standard), Mar. 1991. [Online]. Available: <http://www.ietf.org/rfc/rfc1212.txt>
- [127] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," RFC 3031 (Proposed Standard), Jan. 2001. [Online]. Available: <http://www.ietf.org/rfc/rfc3031.txt>
- [128] S. Routhier, "Management Information Base for the Internet Protocol (IP)," RFC 4293 (Proposed Standard), Internet Engineering Task Force, Apr. 2006. [Online]. Available: <http://www.ietf.org/rfc/rfc4293.txt>
- [129] R. Sabella, P. Iovanna, G. Oriolo, and P. D'Aprile, "Strategy for Dynamic Routing and Grooming of Data Flows into Lightpaths in New Generation Network Based on the GMPLS Paradigm," *Photonic Network Communications*, vol. 7, no. 2, pp. 131–144, Mar. 2004.
- [130] R. Sadre and B. R. H. M. Haverkort, "Fitting heavy-tailed HTTP traces with the new stratified EM-algorithm," in *4th International Telecommunication Networking Workshop on QoS in Multiservice IP Networks (IT-NEWS)*, Venice, Italy. Los Alamitos: IEEE Computer Society Press, Feb. 2008, pp. 254–261.
- [131] R. Sadre and A. Pras, "Dynamic behavior study of large network flows," 2009, to be submitted.
- [132] M. Salehie and L. Tahvildari, "Autonomic computing: emerging trends and open problems," in *DEAS '05: Proceedings of the 2005 workshop on Design and evolution of autonomic application software*. New York, NY, USA: ACM, 2005, pp. 1–7.

- [133] SBC, "JEMS - Journal and Event Management System," Apr. 2009, Brazilian Computer Society (SBC). [Online]. Available: <https://submissoes.sbc.org.br/>
- [134] Schloss Dagstuhl:Seminar Homepage, "Autonomic Management of Networks and Services," 2007. [Online]. Available: <http://www.dagstuhl.de/de/programm/kalender/semhp/?semnr=07302>
- [135] S. Schmid, L. Eggert, M. Brunner, and J. Quittek, "Towards Autonomous Network Domains," in *25th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2006)*. IEEE Press, Apr. 2006.
- [136] J. Schönwälder, "Network management by delegation—from research prototypes towards standards," *Computer Networks and ISDN Systems*, vol. 29, no. 15, pp. 1843–1852, Nov. 1997.
- [137] J. Schönwälder, "Simple Network Management Protocol (SNMP) Context EngineID Discovery," RFC 5343 (Proposed Standard), Internet Engineering Task Force, Sep. 2008. [Online]. Available: <http://www.ietf.org/rfc/rfc5343.txt>
- [138] J. Schönwälder, A. Pras, and J. P. Martin-Flatin, "On the future of internet management technologies," *Communications Magazine, IEEE*, vol. 41, no. 10, pp. 90–97, Oct. 2003.
- [139] J. Schönwälder, A. Pras, M. Harvan, J. Schippers, and R. van de Meent, "SNMP Traffic Analysis: Approaches, Tools, and First Results." in *Integrated Network Management*. Piscataway: IEEE Computer Society Press, May 2007, pp. 323–332.
- [140] C. Shen, D. Pesch, and J. Irvine, "A Framework for Self-Management of Hybrid Wireless Networks Using Autonomic Computing Principles," in *CNSR '05: Proceedings of the 3rd Annual Communication Networks and Services Research Conference*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 261–266.
- [141] A. J. Smith, "Cache Memories," *ACM Computing Surveys*, vol. 14, no. 3, pp. 473–530, 1982.
- [142] G. W. Snedecor and W. G. Cochran, *Statistical Methods*, 8th ed. Iowa State University Press, 1989.
- [143] R. Sommer and A. Feldmann, "Netflow: information loss or win?" in *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurement*. New York, NY, USA: ACM, 2002, pp. 173–174.
- [144] A. Soule, K. Salamatia, N. Taft, R. Emilion, and K. Papagiannaki, "Flow classification by histograms: or how to go on safari in the internet," in *SIGMETRICS '04/Performance '04: Proceedings of the joint international conference on Measurement and modeling of computer systems*. New York, NY, USA: ACM, 2004, pp. 49–60.
- [145] J. Strassner, *Policy-Based Network Management: Solutions for the Next Generation (The Morgan Kaufmann Series in Networking)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003.
- [146] SURFnet, "Surfnet," May 2009. [Online]. Available: <http://www.surfnet.nl>

- [147] SURFnet, "Surfnet6 lightpaths mark start of new internet era," May 2009. [Online]. Available: http://www.surfnet.nl/info/en/artikel_content.jsp?objectnumber=107197
- [148] A. S. Tanenbaum, *Modern Operating Systems*. Upper Saddle River, NJ, USA: Prentice Hall Press, 2007.
- [149] Tcpdump/libpcap, "TCPDUMP/LIBPCAP public repository," May 2009. [Online]. Available: <http://www.tcpdump.org/>
- [150] M. Timmer, P. T. de Boer, and A. Pras, "How to identify the speed limiting factor of a tcp flow," in *4th IEEE/IFIP Workshop on End-to-End Monitoring Techniques and Services*, Apr. 2006, pp. 17–24.
- [151] M. Toure, G. Berhe, P. Stolf, L. Broto, N. Depalma, and D. Hagimont, "Autonomic management for grid applications," in *PDP '08: Proceedings of the 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing (PDP 2008)*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 79–86.
- [152] G. Tselentis, J. Domingue, A. Galis, A. Gavras, and D. Hausheer, *Towards the Future Internet: A European Research Perspective*. Amsterdam, The Netherlands: IOS Press, 2009.
- [153] S. Uhlig and O. Bonaventure, "Understanding the Long-Term Self-Similarity of Internet Traffic," in *COST 263: Proceedings of the Second International Workshop on Quality of Future Internet Services*. London, UK: Springer-Verlag, 2001, pp. 286–298.
- [154] R. van de Meent, "Network link dimensioning: a measurement & modeling based approach," Ph.D. dissertation, University of Twente, Enschede, March 2006.
- [155] J. van den Berg, R. Litjens, A. Eisenblätter, M. Amirijoo, O. Linnell, C. Blondia, T. Kürner, N. Scully, J. Oszmianski, and L. Schmelz, "Self-organisation in future mobile communication networks," in *ICT Mobile Summit*, Stockholm, Sweden, Jun. 2008.
- [156] S. van der Meer, W. Donnelly, J. Strassner, B. Jennings, and M. Ó. Foghlú, "Emerging principles of autonomic network management," in *1st IEEE International Workshop on Modelling Autonomic Communications Environments*, 2006, pp. 29–48.
- [157] R. Van Renesse, K. P. Birman, and W. Vogels, "Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining," *ACM Transactions on Computer Systems*, vol. 21, no. 2, pp. 164–206, 2003.
- [158] W. Vereecken, L. Deboosere, D. Colle, B. Vermeulen, M. Pickavet, B. Dhoedt, and P. Demeester, "Energy efficiency in telecommunication networks (invited paper)," in *Proceedings of NOC2008, the 13th European Conference on Networks and Optical Communications*, 2008, pp. 44–51.
- [159] K. V. Vishwanath and A. Vahdat, "Evaluating distributed systems: does background traffic matter?" in *ATC'08: USENIX 2008 Annual Technical Conference on Annual Technical Conference*. Berkeley, CA, USA: USENIX Association, 2008, pp. 227–240.
- [160] S. Waldbusser, "Remote Network Monitoring Management Information Base," RFC 2819 (Standard), Internet Engineering Task Force, May 2000. [Online]. Available: <http://www.ietf.org/rfc/rfc2819.txt>

- [161] S. Waldbusser, "Remote Network Monitoring Management Information Base Version 2," RFC 4502 (Draft Standard), May 2006. [Online]. Available: <http://www.ietf.org/rfc/rfc4502.txt>
- [162] J. Wang, "A survey of web caching schemes for the Internet," *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 5, pp. 36–46, 1999.
- [163] J. Wang, V. Rao Vemuri, W. Cho, and B. Mukherjee, "Improved approaches for cost-effective traffic grooming in WDM ring networks: nonuniform traffic and bidirectional ring," in *IEEE International Conference on Communications (ICC 2000)*, vol. 3, 2000, pp. 1295–1299.
- [164] U. Warrior, L. Besaw, L. LaBarre, and B. Handspicker, "Common Management Information Services and Protocols for the Internet (CMOT and CMIP)," RFC 1189 (Historic), Internet Engineering Task Force, Oct. 1990. [Online]. Available: <http://www.ietf.org/rfc/rfc1189.txt>
- [165] R. Waterman, B. Lahaye, D. Romascanu, and S. Waldbusser, "Remote Network Monitoring MIB Extensions for Switched Networks Version 1.0," RFC 2613 (Draft Standard), Jun. 1999. [Online]. Available: <http://www.ietf.org/rfc/rfc2613.txt>
- [166] D. X. Wei and P. Cao, "NS-2 TCP-Linux: an NS-2 TCP implementation with congestion control algorithms from Linux," in *WNS2 '06: Proceeding from the 2006 workshop on ns-2: the IP network simulator*. New York, NY, USA: ACM, 2006.
- [167] Wikipedia, "Cache algorithms — Wikipedia, the free encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/Cache_algorithms
- [168] Wikipedia, "Classification tree — Wikipedia, the free encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/Classification_tree
- [169] Wikipedia, "Correlation — Wikipedia, the free encyclopedia," 2009. [Online]. Available: <http://en.wikipedia.org/wiki/Correlation>
- [170] Wikipedia, "Differentiated services— Wikipedia, the free encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/Differentiated_services
- [171] Wikipedia, "Heavy-tailed distribution — Wikipedia, the free encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/Heavy-tailed_distribution
- [172] Wikipedia, "Ipv4 — Wikipedia, the free encyclopedia," 2009. [Online]. Available: <http://en.wikipedia.org/wiki/IPv4>
- [173] Wikipedia, "Operations research — Wikipedia, the free encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/Operations_research
- [174] Wikipedia, "Summary statistics — Wikipedia, the free encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/Summary_statistics
- [175] Wikipedia, "Synchronous optical networking — Wikipedia, The Free Encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/Synchronous_Digital_Hierarchy

- [176] Wikipedia, "Transmission control protocol — Wikipedia, the free encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/Transmission_Control_Protocol
- [177] Wikipedia, "User datagram protocol — Wikipedia, the free encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/User_Datagram_Protocol
- [178] Wikipedia, "Wavelength-division multiplexing — Wikipedia, the free encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/Wavelength-division_multiplexing
- [179] Wikipedia, "Weibull distribution — Wikipedia, the free encyclopedia," 2009. [Online]. Available: http://en.wikipedia.org/wiki/Weibull_distribution
- [180] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (BIC) for fast long-distance networks," in *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 4, Mar. 2004, pp. 2514–2524.
- [181] Y. Zhang, L. Breslau, V. Paxson, and S. Shenker, "On the characteristics and origins of Internet flow rates," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4, pp. 309–322, 2002.
- [182] T. Zseby, E. Boschi, N. Brownlee, and B. Claise, "IP Flow Information Export (IPFIX) Applicability," RFC 5472 (Informational), Internet Engineering Task Force, Mar. 2009. [Online]. Available: <http://www.ietf.org/rfc/rfc5472.txt>
- [183] T. Zseby, M. Molina, N. Duffield, S. Niccolini, and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection," RFC 5475 (Proposed Standard), Internet Engineering Task Force, Mar. 2009. [Online]. Available: <http://www.ietf.org/rfc/rfc5475.txt>
- [184] T. Zseby, T. Hirsch, and B. Claise, "Packet Sampling for Flow Accounting: Challenges and Limitations," in *Passive and Active Network Measurement*, 2008, pp. 61–71.

Glossary

conditional probability is the probability of some event happening given the occurrence of another event.

dark fiber is an unlit optical fiber.

elephant flows are flows that despite being few in number, they represent most of the network traffic.

hybrid network a kind of network that combine more than one networking technology.

IP flow a unidirectional sequence of IP packets that share the same properties.

lightpath a direct optical data connection over an optical fiber.

multi-service hybrid device works as both a switch at the optical level and a router at the IP level.

multiplexing is a process of combining multiple signals into one single signal over a shared medium.

NetFlow is an embedded instrumentation software developed by Cisco Systems that is used to characterize network information.

traffic grooming the process of multiplexing many flows into a single lightpath.

Acronyms

ACI Autonomic Computing Initiative.

ANSI American National Standards Institute.

ASN.1 Abstract Syntax Notation One.

AUTOI Autonomic Internet.

BDP Bandwidth-Delay Product.

BER Basic Encode Rules.

Bps Bytes per second.

CHAID CHI-squared Automatic Interaction Detector.

CIM Common Information Model.

CLI Command Line Interface.

CMIP Common Management Information Protocol.

CR-LDP Constraint-Routing Label Distribution Protocol.

DiffServ Differentiated Services.

DMTF Distributed Management Task Force.

DNS Domain Name System.

DoS Denial of Service.

DWDM Dense Wavelength Division Multiplexing.

EFIPSANS Exposing the Features in IP version Six protocols that can be exploited/extended for the purposes of designing/building Autonomic Networks and Services.

EIGRP Enhanced Interior Gateway Routing Protocol.

FIFO First In First Out.

FSC Fiber-Switch Capable.

GDS Greedy Dual-Size.

GMPLS Generalized Multiprotocol Label Switching.

HDTV high-definition television.

HED Hyper-Exponential Distribution.

IETF Internet Engineering Task Force.

IPFIX IP Flow Information eXport.

IS-IS Intermediate System-to-Intermediate System.

ITU International Telecommunications Union.

JEMS Journal and Event Management System.

LAN Local Area Network.

LER Label Edge Router.

LFF Largest File First.

LRU Least Recently Used.

LSC Lambda Switch Capable.

LSP Label Switch Path.

LSR Label Switch Router.

MAN Metropolitan Area Network.

MIB Management Information Base.

MPLS Multiprotocol Label Switching.

MRU Most Recently Used.

NMS Network Management System.

NTP Network Time Protocol.

OBS Optical Burst Switching.

OCS Optical Circuit Switching.

OPS Optical Packet Switching.

OSPF Open Shortest Path First.

OXC Optical Cross-Connected.

Pps Packets per second.

PSC Packet Switching Capable.

QoS Quality of Service.

RMON Remote Monitoring.

RSVP-TE Resource Reservation Protocol for Traffic Engineering.

RTT Round-Trip Time.

SDH Synchronous Digital Hierarchy.

SLRU Segmented LRU.

SMI Structure of Management Information.

SNMP Simple Network Management Protocol.

SONET Synchronous Optical Networking.

STM Synchronous Transport Module.

STS Synchronous Transport Signal.

TCP Transmission Control Protocol.

TDM Time-Division Multiplexing.

TL1 Transaction Language 1.

ToS Type of Service.

UAV Unmanned Autonomous Vehicle.

UDP User Datagram Protocol.

UT University of Twente.

WAN Wide Area Network.

WBEM Web-Based Enterprise Management.

WDM Wavelength Division Multiplexing.

XML Extensible Markup Language.

Index

- Autonomic decision process, 81
 - algorithm, 84
 - assumptions, 87
 - decision policy, 84
 - validation, 93
- Cache management, 78
- Command Line Interface, 19
- Conventional management approaches, 17
 - Direct management approach, 19
 - Indirect management approach, 23
- Flows
 - definitions, 41
 - elephant, 45
 - grooming, 96
- Generalized Multiprotocol Label Switching, 24
- Hybrid network, 1
- IP traffic monitoring
 - flow-based techniques, 58
 - packet-based techniques, 55
 - SNMP-based techniques, 57
- Multiplexing, 4
 - SONET and SDH, 4
- Multiprotocol Label Switching, 23
- Network management
 - FCAPS, 6
 - Network parameters
 - behavior, 44
 - identifiers, 40
 - Packet sampling
 - effects, 62
 - random sampling, 62
 - systematic sampling, 62
 - Self-management
 - definitions, 28
 - impact, 104
 - lightpaths, 31
 - manifesto, 27
 - Simple Network Management Protocol, 20
 - Statistical analysis
 - classification tree, 48
 - conditional probability, 46
 - correlation, 47
 - summary statistics, 45
 - SURFnet6, 3
 - Transaction Language 1, 20
 - Web-Based Enterprise Management, 21

About the author



Tiago Fioreze was born in Tapera, Rio Grande do Sul, Brazil on October 10, 1979. He received his Bachelor of Science (B.Sc.) degree in Computer Science from the Federal University of Santa Maria (UFSM) in 2003 and his Master of Science (M.Sc.), in the same area, from the Federal University of Rio Grande do Sul (UFRGS) in 2005. In 2009, he enrolled as a Ph.D. candidate in the Design and Analysis of Communication Systems (DACS) group at the University of Twente. DACS is a research group that belongs to the Faculty of Electrical Engineering, Mathematics, and Computer Science (EEMCS), which is a faculty under the organizational umbrella of the Centre for Telematics and Information Technology (CTIT). During his Ph.D., he had the opportunity of participating in the European Network of Excellence on Management Solutions for Next Generation Networks (EMANICS) and the Gigaport-NG Research on Networks projects. As a hobby in his spare time, Tiago Fioreze likes taking photographs. He is also an enthusiastic about sports, specially indoor soccer (Futsal). Passionate about soccer as most Brazilian people, his favorite soccer team is Grêmio Foot-ball Porto Alegrense.

A list of his publications in reverse chronological order:

- Aiko Pras, Ramin Sadre, Anna Sperotto, Tiago Fioreze, David Hausheer, Jürgen Schönwälder, “Using NetFlow / IPFIX for Network Management”, *Journal of Network and Systems Management (JNSM)*, Springer New York, December 2009, Vol. 17, Num. 4, ISSN 1064-7570, pp. 482-487
- Giovane C.M. Moura, Tiago Fioreze, Pieter-Tjerk, Aiko Pras, “Optical Switching Impact on TCP Throughput Limited by TCP Buffers”, *Proceedings of the 9th IEEE International Workshop on IP Operations and Management (IPOM 2009)*, 29-30 October 2009, Venice,

Italy, Lecture Notes in Computer Science, Vol. 5843, ISSN 0302-9743 ISBN 978-3-642-04967-5, pp. 161-166

- Tiago Fioreze, Lisandro Granville, Ramin Sadre, Aiko Pras, “*A Statistical Analysis of Network Parameters for the Self-Management of Lambda-Connections*”, Proceedings of the 3rd International Conference on Autonomous Infrastructure, Management and Security (AIMS 2009), June 30 - July 2, 2009, Enschede, The Netherlands, Lecture Notes in Computer Science, Vol. 5637, ISSN 0302-9743 ISBN 978-3-642-02626-3, pp. 183-186
- Rick Hofstede, Tiago Fioreze, “*SURFmap: a network monitoring tool based on the Google Maps API*”, Application session proceedings of the 11th IFIP/IEEE International Symposium on Integrated Network Management (IM 2009), 1-5 June 2009, Long Island, New York, USA, ISBN 978-1-4244-3487-9, pp. 676-690
- Tiago Fioreze, Lisandro Zambenedetti Granville, Aiko Pras, Anna Sperotto, Ramin Sadre, “*Self-management of Hybrid Networks: can we trust NetFlow data?* Mini-conference proceedings of the 11th IFIP/IEEE International Symposium on Integrated Network Management (IM 2009), 1-5 June 2009, Long Island, New York, USA, ISBN 978-1-4244-3487-9, pp. 577-584
- Tiago Fioreze, Mattijs Oude Wolbers, Remco van de Meent, Aiko Pras, “*Characterization of IP Flows Eligible for Lambda-Connections in Optical Networks*”, Proceedings of the 11th IFIP/IEEE Network Operations and Management Symposium (NOMS 2008), 07-11 April 2008, Salvador, Bahia, Brazil, ISSN 1542-1201 ISBN: 978-1-4244-2066-7, pp. 256-262
- Tiago Fioreze, Mattijs Oude Wolbers, Remco van de Meent, Aiko Pras, “*Offloading IP flows onto lambda-connections*”, Proceedings of the 18th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM 2007), 29-31 October 2007, San Jos, California, USA. Lecture Notes in Computer Science, Vol. 4785, ISSN 0302-9743 ISBN 978-3-540-75693-4, pp. 183-186
- Tiago Fioreze, Remco van de Meent, Aiko Pras, “*An Architecture for the Self-management of Lambda-Connections in Hybrid Networks*”, Proceedings of the 13th EUNICE Open European Summer School 2007 (EUNICE 2007), 18-20 July 2007, Enschede, The Netherlands. Lecture Notes in Computer Science, Vol. 4606, ISSN 0302-9743 ISBN 978-3-540-73529-8, pp. 141-148
- Tiago Fioreze, Aiko Pras, “*Self-management of lambda-connections in optical networks*”, Proceedings of the 1st International Conference on Autonomous Infrastructure, Management and Security (AIMS 2007) Student Workshop, 21-23 June 2007, Oslo, Norway. Lecture Notes in Computer Science, Vol. 4543, ISSN 0302-9743 ISBN 978-3-540-72985-3, pp. 212-215
- Tiago Fioreze, Mattijs Oude Wolbers, Remco van de Meent, Aiko Pras, “*Finding Elephants Flows for Optical Networks*”, Application session proceedings of the 10th IFIP/IEEE International Symposium on Integrated Network Management (IM 2007), 21-25 May 2007, Munich, Germany, ISBN: 1-4244-0799-0, pp. 627-640

-
- Tiago Fioreze, Aiko Pras, *"Using Self-management for Establishing Light Paths in Optical Networks: an Overview"*, Poster session proceedings of the 12th EUNICE Open European Summer School 2006 (EUNICE 2006), 18-20 September 2006, Stuttgart, Germany, ISBN: 3-938965-02-9, pp. 17-20
 - Tiago Fioreze, Ricardo Neisse, Lisandro Zambenedetti Granville, Maria Janilce Almeida, Aiko Pras, *"A Policy-Based Hierarchical Approach for Management of Grids and Networks"*, Application session proceedings of the 10th IEEE/IFIP Network Operations and Management Symposium (NOMS 2006), 3-7 April 2006, Vancouver, Canada, ISBN: 1-4244-0143-7, ISSN: 1542-1201, 1-14