UNIVERSITAT POLITÈCNICA DE CATALUNYA

Departament d'Arquitectura de Computadors

GMPLS-OBS Interoperability and Routing Scalability in Internet

Ph.D. thesis by

Pedro Pedroso

Advisor:
Dr. Davide Careglio
Co-Advisor:
Prof. Josep Solé i Pareta

PhD Program: Arquitectura de Computadors

October 2011

Acknowledgements

"Be free to do whatever you wanna do ... Be free to go wherever you wanna go ..." (unknown)

To my parents and my sister, because family always cames first. I am so proud of having YOU as my family!

To Sofia, my princess who shared with me these amazing four years of my life.

To Davide and Josep, for providing me with this unique research opportunity and such nice experiences. *Moltes gràcies!*

To Mirek and Dimitri, from whom I learned a lot. It was a pleasure and a privilege to work with you.

To Pipo, Fred and Ricardo, my friends, my brothers.

Contents

Summary

This Ph.D. thesis is a research contribution within the Optical Networking and Future Internet Architecture fields. Future Internet means any further development of the original Internet, which is a quite spread topic due to the wide diversity of technologies that it comprises. This being said, this thesis aims to a Future Optical Internet architecture. In particular, it focus on an IP over optical burst switching (OBS) scenario, addressing both its transport and IP network problems, namely the Generalized Multi-Protocol Label Switching (GMPLS) operability as the control plane for the OBS network and the inter-domain routing scalability in Internet.

The popularization of Internet has turned the telecom world upside down over the last two decades. Network operators, vendors and service providers are being challenged to adapt themselves to Internet requirements in a way to properly serve the huge number of demanding users (residential and business). The Internet (data-oriented network) is supported by an IP packet-switched architecture on top of a circuit-switched, optical-based architecture (voice-oriented network), which results in a complex and rather costly infrastructure to the transport of IP traffic (the dominant traffic nowadays).

In such a way, a simple and IP-adapted network architecture is desired. Basically, the main concerns in the horizon are both in terms of bandwidth usage efficiency and quality-of-service (QoS) provisioning, which affect both i) the transport network, which needs to satisfy such upper (IP) packet-oriented architecture and services, and ii) the IP network itself, as it needs to cope with the non-stopping, already huge number of client applications and its dynamics efficiently.

From the transport network perspective, the idea is to progress towards an IP-over-WDM architecture, where the IP packets can travel as close as possible to the (optical) physical mean. This would eliminate redundancy in IP transport and reduce operational costs. Two of its main requirements are i) a control plane providing protection and restoration mechanisms as well as automatic resource and connection management as defined by the Automatically Switched Optical Networks (ASON) - an ITU-t standard; and ii) an all-optical switching layer oriented to packet transport. Both GMPLS and OBS technologies are part of the set of solutions, providing intelligence in the control and management of resources (i.e. GMPLS) as well as a good network resource access and usage (i.e. OBS).

The GMPLS framework [Farrel06] is the key enabler to orchestrate a unified optical network control and thus reduce network operational expenses (OPEX), while increasing operator's revenues [Pioneer02a]. The GMPLS is now mature - is well studied and standardized [RFC3945], and can be easily extended by IETF whenever new requirements arise - and its deployment seems almost inevitable in the upcoming years [Morrow05].

Simultaneously, the OBS technology, which avoids the current hurdle of optical buffering, is one of the well positioned switching technologies to realise the envisioned IP-over-WDM network architecture. OBS leverages on the statistical multiplexing of data plane resources to enable sub-wavelength in optical networks [Qiao99]. This is attained by aggregating packets arriving at ingress nodes into bursts, which are sent to the destination over a buffer-less optical network infrastructure. High bandwidth utilization is achieved in OBS by means of one-way resource reservation schemes, which minimize the overhead introduced during the burst signaling process. However, these schemes only by themselves do not ensure successful burst delivery as, due to the statistical multiplexing of the wavelength channels, bursts may contend and some of them be lost.

Despite the GMPLS principle of unified control, little effort has been put on extending it to incorporate the OBS technology and many open questions still remain. The GMPLS and OBS interoperability design has been dragging on for quite a long time. The first general attempt is dated from the year 2000 and since then no major work has ever come up with significant neither functional or operational proposals, nor performance analysis.

From the IP network perspective, the Internet is facing scalability issues as enormous quantities of service instances and devices must be managed. Nowadays, the scalability, convergence, and stability properties of Internet inter-domain routing and addressing systems are being questioned. It is believed that the current Internet features and mechanisms cannot cope with the size and dynamics of the Future Internet.

For instance, the current size and growth rate of the routing tables (RT) are reaching unbearable values. The memory-space consumption is huge. Neither full information about all possible unicast/multicast sets, nor store the global topological information can be maintained anymore in order to cope with the growth of Internet.

Compact Routing is one of the main breakthrough paradigm on the design of a routing system scalable with the Future Internet requirements. It intends to address the fundamental limits of current stretch-1 shortest-path routing in terms of RT scalability (aiming at sub-linear growth). It is well positioned as the main alternative to overcome the poor scaling properties of the current Internet routing system.

Although "static" compact routing works fine, scaling logarithmically on the number of nodes even in scale-free graphs such as Internet, it does not handle dynamic graphs. Multihoming and mobility require topology-independence node identifiers as any topology-driven, policy-driven or protocol-driven change event must trigger the exchange of routing messages to timely update the non-local topology information of each node in the network as well as the renumbering of the IP devices.

Moreover, as multimedia content/services proliferate, the multicast is again under the spotlight as bandwidth efficiency and low RT sizes are desired. However, it makes the problem even worse as more routing entries should be maintained. Multicast problems from the 90's remain unsolved as many multicast protocols are still laid on underlying unicast routing protocol and full topology graph view. Overlaying multicast routing on top of unicast suffers however from the same scaling limitations as current unicast routing with the addition of the level of indirection added by the multicast routing application.

In the context of Compact Routing, there is one attempt only from 2009 dealing with multicast scenario. In [Abraham09], the authors present a source-routed, labeled (topology-aware) compact multicast routing algorithm with some theoretical results but considerable missing details on how they reach to certain assumptions as well as a numerical validation.

In a nutshell, this thesis contributes with detailed solutions dealing both with i) GMPLS-OBS control interoperability (Part I), fostering the desired unified control over multiple switching domains and reducing redundancy in IP transport and ii) new Internet routing schemes for multicast scenarios (Part II), in order to overcome Internet inter-domain routing system scalability problem. Exhaustive simulation campaigns are run in both cases in order to assess the reliability and feasible of the proposals.

Part I consists in a complete study of the proposed GMPLS-OBS control architecture and model. From a functional perspective, the proposed solution tries to overcome those technology-specific interoperability issues. From an operational perspective, it offers (absolute) QoS guarantees to OBS networks by making use of the traffic-engineering (TE) features of GMPLS. Keys extensions to the GMPLS protocol standards are equally approached and suggested to cope with such architecture. The successful numerical simulations reinforce the feasibility of our proposals.

Part II consists in the study of new routing paradigms so as to design, develop, and validate distributed and dynamic routing schemes suitable for the future Internet and its evolution. In such a way, the AnyTraffic Labeled routing concept and algorithm is introduced that saves on forwarding entries by sharing a single forwarding entry to unicast and multicast traffic type. On the other hand, the first known name-independent (i.e. topology unaware) compact multicast routing algorithm is proposed. Very promising results (addressing the triple trade-off: stretch x RT size scale x dynamics) have been achieved by addressing the root causes (disruptive attitude) instead of short-term incremental fixes to attenuate symptoms (evolutionary attitude).

Introduction

In the EULER Project [1] *we have defined Internet as: a large, dynamic and heterogeneous collection of distributed systems interconnected for global end-to-end communications of many different types between any interested parties connected to it (host).*

The popularization of Internet has turned the telecom world upside down over the last two decades. A whole infrastructure that had been ruled by telephony (i.e. voice-traffic) is now taken by data-traffic. The move from analogue to digital information storage and transmission has forced to the design of a single network capable of delivering multiple services: IP technology has overcome any other competitor. Nowadays, practically all forms of end-user communications make use of the TCP/IP architecture [Ghani00]. This explosive growth finds its reason in the (IP) protocol simplicity along with the fast expansion of Internet and its easy access. The Internet has created a quite heterogeneous telecom world as it comprises a plethora of new technologies, either wireless or wired, either mobile or fixed, and has brought millions of people online, who not only received but also generate and inject information in the networks. Internet has changed the rules of the game!

What are we struggling for?

Those millions of people worldwide are generating enormous amount of traffic with a growth rate between 40 and 50 percent/year [CISCO10]. The rapid deployment of broadband access technologies like FTTx, WiFi, WiMax and LTE, is bringing high bandwidth to those end-users. This potentiates the emergence of bandwidth demanding applications increasing the traffic injected in the networks. The cycle is vicious as more bandwidth availability leads to the emergence of applications using more bandwidth, which in turn requests even more bandwidth. Each one of these new services or applications is based on the IP protocol. We are witnessing a convergence of multiple services into a single IP packet-oriented network infrastructure. This led ITU-t to define the Next Generation Network (NGN) as a packet-based network able to provide telecommunication services and able to make use of multiple broadband, QoS-enabled transport technologies [ITU04]. New requirements in terms of bandwidth utilization and QoS provisioning are in the horizon of both the transport network, which needs to satisfy such upper (IP) packet-oriented infrastructure and services, and the IP network itself, as it needs to cope with such huge number of client applications and its dynamics efficiently.

Figure 1 illustrates perfectly why data-centered technologies are needed to keep pace with these new requirements from the "telecom revolution" [MCE10]. The traffic-volume curve does not seem to decelerate. In fact, as long the data dominated period has started, the curve has been altered (its growth trend has become exponential) reflecting the paradigm shift introduced by Internet - an end-user not only receives information, but he can also generate it, bringing much more traffic to the networks -. This contrasts with the voice dominated period, during which the curve was showing signs of slowing down. The problem resides on the transport network costs. The network-cost curve shows the same trend as the traffic-volume curve if current existing technologies remain (i.e. those voice-oriented). Profitability will be achieve if and only if the network-cost curve stays under the revenue one. This is enough to push every telecom player (operators, vendors, network and service providers) to change their infrastructures.

Three high-level requirements are bound together to the achievement of an efficient and effective future Internet/NGN architecture offering high bandwidth access: Quality, Cost

[1] see Technical Acknowledgements

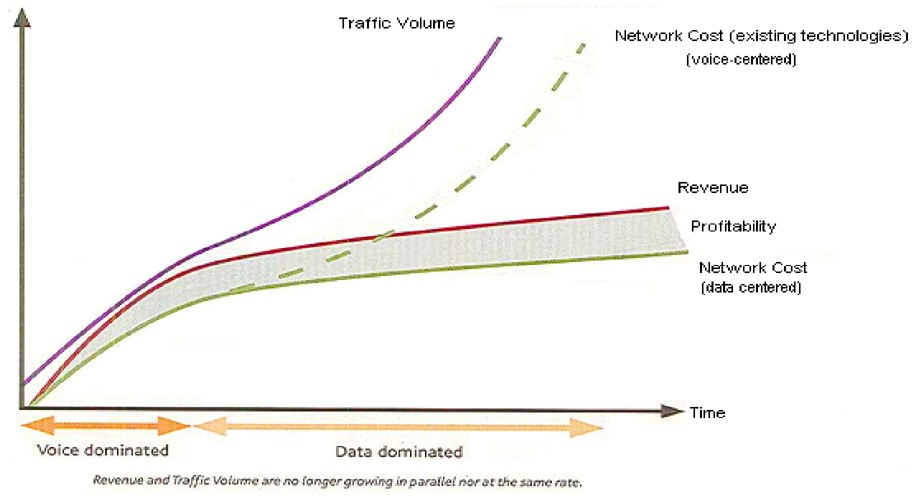


Figure 1: Comparison between the voice and data dominated periods and the forecast of the evolution of several network parameters: {traffic volume, revenue, cost and profitability} [MCE10].

and Scalability. This means high quality at the lowest possible cost in a network, highly scalable and offering high bandwidth access. All this to serve a huge number of clients and highly dynamic traffic volumes. Such requirements are pointed out below and illustrated in Fig. 2.

✓ High Quality: end-users are interested in quality more than in quantity (i.e. capacity), which they do not perceive directly. The distribution of high-definition (HD) content demands the guarantee of the QoS metric at any rate. Indeed, a new metric has been introduced to measure user experience (QoE), given a more accurate quality assessment. Those services and applications are proliferating: 1) IP video: HD-TV, 3D-TV, video on demand (VoD) and pay-per-view (PPV), gaming, home shopping, video-conferencing; 2) Voice over IP (VoIP) service, and 3) data services as web browsing, internet chat and multimedia file sharing.

✓ Low Cost: Profitability is the difference between revenues and expenses and at the end of the day it is what matters the most in such competitive and profit-driven world. In order to achieve high profitability, network OPEX should be maintained as low as possible. As shown in Fig. 1, the network cost is strictly correlated with the existing technologies.

✓ High Scalability: scalability is the ability of a system, network, or process, to handle growing amounts of work in a graceful manner or its ability to be enlarged to accommodate that growth [Bondi00]. Poor scalability can result in poor system performance, requiring the re-engineering or duplication of systems. High scalability is desirable considering the high number of devices present in today's networks and the amount of traffic generated and received by end-users (either residential or business). Centralized control and manual intervention should give way to intelligent and distributed systems. It is not feasible anymore to concentrate the control in centralized entities or have and maintain global knowledge of the network.

What must be changed?

The global network infrastructure consists of a transport network platform of high-transmission capacity in the core. It is accessed through wireless (e.g. WiMax, LTE) and wired (e.g. GPON, FTTx) networks to reach a wide diversity of users [Mahony06]. As said before, we are watching a convergence of different services - quad-play service provision: voice, video(multimedia), data and mobile (wireless) - into an IP packet-switched network architecture, albeit the transport infrastructure itself remains voice-oriented. The dominance of such data-oriented services over a voice-oriented transport infrastructure has created a heavy and redundant network protocol stack (IP/MPLS/SDH/WDM) which is costly and rather wasteful (as shown in Fig.1). Ideally, the data switching should be performed in an optical based transport network (OTN) infrastructure (aim $\Rightarrow$ IP directly over WDM), which should evolve from a circuit-oriented to a packet-oriented switching technology. The goal is to have a finer granular

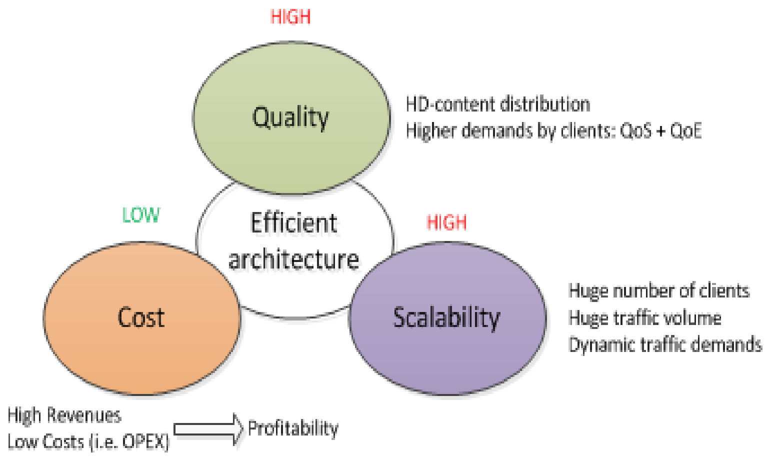


Figure 2: High-level requirements of the Future Internet/NGN architecture.

access to optical transmission resources (aim: ⇒ all-optical packet-switching), better resource usage and transport cost reduction. Equally important is the reduction of manual intervention, which results in unbearable costs to network operators (aim: ⇒ intelligent control plane - ASON/GMPLS standards). In terms of the client network (IP/MPLS), changes must take place too. The continuous increase of the volume of traffic is leading the telecom players (both industry and academia) to question if the current protocols and algorithms are scalable and efficient enough to cope with it (aim: ⇒ new routing schemes to the Internet). The main question can be formulate as follows:

How to efficiently manage such high and dynamic volume of packet-switched traffic, either from a transport or application perspective?

Towards IP-over-WDM Architecture The current network architecture is complex and rather costly to the transport of IP traffic. The IP-over-WDM is a consensual architecture to the Future Optical Internet. It is simple, logical and cheap. This requires complementary features to be deployed in order to provide flexible and granular access to bandwidth as well as protection and recovery mechanisms. On the one hand, an all-optical switching layer is envisioned to better cope with the packet-oriented traffic at the optical layer [IEEE02]. This is desired chiefly due to the lower transport cost in the optical layer (as one does not need to go through every node at the IP level). The cost/power consumption per bit in optical layers devices is 1/3 to 1/5 of the IP layer devices. On the other hand, in a way to endow the OTN with intelligence, to reduce manual intervention and provide it with protection and recovery mechanisms, a control plane is essential. It automates as well as speeds up the involved processes and actions such as provisioning, topology and resource discovery, and recovery. A distributed control approach is the most desirable choice for the next generation of intelligent optical networks because of its: better scalability; cost-reduction; robustness; flexibility; and adequateness to dynamic traffic environments.

Packet-oriented Optical-Switching Layer The bandwidth capacity in optical networks is not a problem *per se*. Thanks to the improvements in fibre technology, the cost of increasing raw bandwidth capacity has been decreasing about as fast as the traffic grows. The last advances in optical technology namely WDM, DWDM and more recently DWDM+PDM have made it possible to exploit the huge potential bandwidth of optical fibres (up to 10 Tbps per single fibre) [Jinno07]. Those technologies, which can already achieve 32 wavelength channels (WDM) or 80-120 wavelengths per fibre (DWDM) of 10 Gbps or even 40 Gbps each, have the total capacity overriding the 1 Tbps barrier. Thus, the main goal is to achieve optimal usage and access of these resources according to high volumes of traffic and end-to-end QoS demands.

Recently, Carrier Ethernet has emerged as a cost-effective and easy alternative solution to the OTN. It represents the set of attributes needed for Ethernet to be successfully deployed in MAN/WAN networks while preserving its inherent characteristics: low costs, simple provisioning, and management and topological flexibility. The Provider Backbone Bridging Traffic Engineering (PBB-TE) (initially Provider Backbone Transport (PBT)) technologies together with new Ethernet OAM standards extend the capabilities of Ethernet to transform it into a true carrier-class technology [IEEE09]. With the same target, IETF suggests the GMPLS Ethernet Label Switching (GELS) as the architecture for a GMPLS based control plane for Ethernet [RFC5828]. Although all these technologies support the growing demand for high-bandwidth video and data services in a short term, and are suitable to deploy Ethernet in transport networks, they are not truly all-optical packet switching solutions and will always suffer from its static connection orient paradigm at the optical layer.

According to [Mahony06], the evolution towards the future Internet has moved research efforts into the realization of multi-service optical networks performing wavelength and sub wavelength switching. Finer granularity is desired at the optical switching layer level so as to seamlessly and efficiently support large amounts of data from different applications presenting diverse characteristics. Optical Circuit and Packet Switching (OCS/OPS), together with OBS, appear as solutions providing all-optical switching of data, high capacity, high flexibility and efficiency on network usage. These technologies differ chiefly on how resources are allocated in the core and in the degree of offered switching granularity.

Control-Plane: ASON/GMPLS The aforementioned IP-over-WDM architecture requires an optical control plane and fosters the desired unified control plane operating in a distribution fashion mode. It takes on an important role, as all the required intelligence (i.e. automatic control operations) should be delegated to this new entity. This CP enables fast and automatic end-to-end connection provisioning and TE features covering multiple switching technologies. Recovery and protection are also under the CP operations. The ITU-t recommendation G.8080 defines the reference architecture for the CP of automatically switched optical networks (i.e. ASONs). In such a scenario, the GMPLS standard [Farrel06] emerges as the most accepted solution to implement the ASON control plane.

IP packet-switched architecture: scalability issues The Internet, supported by an IP packet-switched architecture, is facing scalability issues as enormous quantities of service instances and devices must be managed. Current Internet features are not prepared to accommodate such volume and dynamics. The scalability, convergence, and stability properties of Internet inter-domain routing and addressing systems are concerning many as summarized in the IETF workshop from 2006 [RFC4984]. It is believed that such features and mechanisms cannot cope with the size of the future Internet. Each player (i.e. device) in the Internet requires an IP address, besides mobility and multi-homing support. The high level of addressing aggregation (and also de-aggregation of IP addresses), hierarchical routing in the Internet and the larger number of routing entries to be maintained are leading to a huge memory consumption and computational complexity. To overcome the IPv4 addressing exhaustion, the IPv6 will take place although it will also contribute to the Routing/Forwarding tables (RT/FT) size growth as more bits need to be stored. Last but not least, the mobility is also increasing the number of routing information messages exchanged, which update and recompute the RT/FT upon topological or policy changes). It is also slowing down the network convergence time (i.e. routing algorithm computation) as it requires frequent renumbering of IP devices.

The few years proposal suggesting the separation of the locator and the identifier of a node of the network is quite popular. This is termed as Loc/ID split (LIS) technique. Here, only the locators will be affected by topology changes. While this enhances the flexibility of the routing system, many question if it really addresses the scalability concerns. Nevertheless, the IETF is working on it through the LIS protocol (LISP) [Farinacci11].

Compact Routing is another research field that studies fundamental limits of routing scalability going beyond the poor scaling properties of the current Internet [Krioukov07]. It offers remarkably better scaling than the one observed in Internet routing today (logarithmic vs. exponential RT size growth) aiming at memory efficiency. Also in [Krioukov07] it is said that we are at an apparent impasse as current static, name-dependent (i.e. topology aware) compact routing algorithm does not cope with the desired level of scalability to future Internet. According to the authors, we need radically new ideas that would allow us to construct convergence-free, "updateless" routing, requiring no full view of the network topologies. So far, name-independent compact routing algorithms do not perform better than name-dependent in Internet-like topologies.

On the other hand, we have Greedy routing schemes which basically aim at the same target [Kleinberg05]. These schemes use the geometric properties of the topology, facilitating the packet forwarding process. Its efficiency remains robust even under dynamic network conditions. However, they also require full view of the topology graph. The updateless counterpart [Krioukov08] removes such thing and is in good place to answer to the most pressing questions by making use of hyperbolic geometric space.

Multicast Routing As multimedia content/services are proliferating, the multicast is again under the spotlight as bandwidth efficiency and low RT sizes are desired. But multicast problems from the 90's remain unsolved as many multicast protocols are still laid on underlying unicast routing protocol and full topology graph view. Indeed, routing protocol dependent multicast routing schemes (such as Distance Vector Multicast Routing Protocol and Multicast Open Shortest Path First) have been replaced by routing protocol independent routing schemes such as Protocol Independent Multicast (PIM) and Core Base Trees (CBT). Overlaying multicast routing on top of unicast suffers however from the same scaling limitations as current unicast routing with the addition of the level of indirection added by the multicast routing application.

Contribution of the Thesis

This work is focused in an IP over OBS scenario and deals with both the transport and the IP networks. We address two main problems, namely i) the GMPLS and OBS interoperability and ii) the inter-domain routing scalability in the IP networks.

Concerning the former (Part I of the thesis), the problem is related with the Internet transport architecture and its simplification towards an intelligent Future Optical Internet i.e. an IP-over-WDM architecture provided with an automatic control plane and an all-optical switching layer. The problem to address requires an engineering solution more than a pure scientific one and the objectives are very pragmatic. This research topic was part of two European projects [2], namely NOBEL and BONE. A join collaboration with Anna Manolova from Denmark Technical University (DTU), Denmark was established, resulting in one mobility exchange period (4 weeks).

[2]see Technical Acknowledgements

For what refers to the later (Part II of the thesis), the problem is related with the Internet routing system, which is facing scalability issues. New routing algorithms to the Internet are here designed and proposed from scratch. This research topic is now part of EULER project[2], albeit it has started much before within collaboration with Alcatel-Lucent Bell Labs, Belgium, and Dr. Dimitri Papadimitriou. Several short-term internships were equally performed at the company (in a total of 11 weeks).

In both cases, theoretical analysis is presented, being supported by exhaustive numerical simulations. The idea of the research work is to provide efficient, effective and pragmatic solutions within these two research topics contributing to its progress. Note that we will point the reader to reference literature whenever it is appropriated in order to avoid redundancy and information overloading of well-defined and well-studied technologies.

1) GMPLS-OBS Interoperability: Objectives and Contributions The Future Optical Internet forecasts a more pragmatic IP transport and therefore the IP-over-WDM architecture is the envisioned architecture. Two of its main requirements are i) a control plane providing protection and restoration mechanisms as well as automatic resource and connection management as defined by the ASON standard; and ii) an all-optical switching layer oriented to packet transport, which eliminates redundancy in the IP transport and reduces operational costs. Both GMPLS and OBS technologies are part of the set of solutions, providing intelligence in the control and management of resources (i.e. GMPLS) as well as a good network resource access and usage (i.e. OBS). Currently, these two technologies cannot coexist as GMPLS is not adapted to handle OBS. It is a challenging problem with a lot of engineering flavour. There are not many research work in this field, and none regarding operational performance analysis as far as we know.

- ✓ **Objectives:** to show how GMPLS and OBS should coexist efficiently and why, both from functional and operational perspectives. In such a way, we aim to provide GMPLS with the proper protocol extensions to handle OBS technology and to provide OBS with a well-defined automatic control plane. This would solve OBS multi-domain interoperability problem when a GMPLS control instance is in place and, at the same time, its performance issues, such as high burst losses in highly loaded scenarios (by offering absolute QoS guarantees). It is not our intention to proof GMPLS or OBS benefits. This can be found in extensive literature work, as we point out along Part I.

- ✓ **Contributions:** an intelligent control architecture and operational model is presented, where GMPLS and OBS technologies coexist. We detail on the building blocks forming part of the logical GMPLS/OBS node, as well as on the (vertical) communication channel created between both control layers. A protocol is suggested to such channel and a possible burst control packet standard format is described and compared with other proposals. Moreover, the control model is made QoS-aware, providing absolute QoS guarantees to quality-demanding traffic while achieving an efficient network usage. Optimization techniques are applied on the design of routing and wavelength assignment heuristic algorithms used to accomplish such purposes. The milestone of this work is that all control mechanisms (i.e. intelligence) deployed so far at the OBS control layer, should be moved and provided from the GMPLS control layer, relieving OBS computational complexity. Numerical simulations are performed to validate and demonstrate the efficiency and reliability of such operational model. An event-driven simulator is used consisting in two independent control layers plus a centralized node with a path computation element (PCE). The results achieved show a very good reliability and effectiveness of the proposals.

2) Routing Scalability in Internet: Objective and Contributions The Internet routing system of large backbone operators is facing scalability problems. Millions of devices and service instances are demanding fast network convergence times upon (topological or policies) changes, mobility and multi-homing, and high quality content distribution. The challenge remains on the design of a routing algorithm that does not require full topology view, that limits the routing information exchanges, and that achieves a sub-linear RT size growth. The foreseen RT size values are not scalable with the size and dynamics of the Future Internet topologies.

- ✓ **Objectives:** to develop novel dynamic routing paradigms to design a routing system suitable for the future Internet and its evolution. Specifically, we will focus on the Compact Routing research approach, aiming to find the following trade-offs between: i) Routing table size: so as to enhance memory scalability; ii) Routing scheme stretch $\rightarrow$ so as to ensure routing quality; iii) Communication cost $\rightarrow$ messaging so that each router reaches and maintains a consistent view of non-local network topology; iv) Computational cost $\rightarrow$ processing of routing updates for (re-)computation of RT entries $\rightarrow$ Computational complexity.

- ✓ **Contributions:** introduction of innovative concepts to the Internet routing envisioning sub-linear RT size growth. Firstly, the AnyTraffic concept is introduced where a single forwarding entity is responsible to forward both point-to-point and point-to-multi-point traffic types, saving routing entries with limited degradation of both traffic routing (centralized source-routing approach, requiring full knowledge of the topology). Moving towards a distributed scenario, with limited topological view, we introduce by the first time a name-independent, dynamic and leaf-initiated compact multicast routing. In comparison to the state-of-the-art scheme suggested in [Abraham09], we eliminate the topology-awareness requirement. This resulted in better RT size reduction, lower computational complexity and lower convergence algorithm times when a topology/policy changes occur. Our algorithm minimizes the stretch bound for specialized graphs (satisfying properties of Internet-like graphs) ranging from O(10k) up to O(32k) nodes. Very promising results were achieved.

Structure of the Thesis

This Ph.D. report is structured like a grid, as illustrated in Fig.3. Note that long lasting descriptions to mature architectures, models or algorithms are not included in this report, and the readr is rather pointed out to reference literature or the appendix section.

Vertically, this report is divided in two parts (columns) representing both topics addressed, namely Part I - GMPLS-

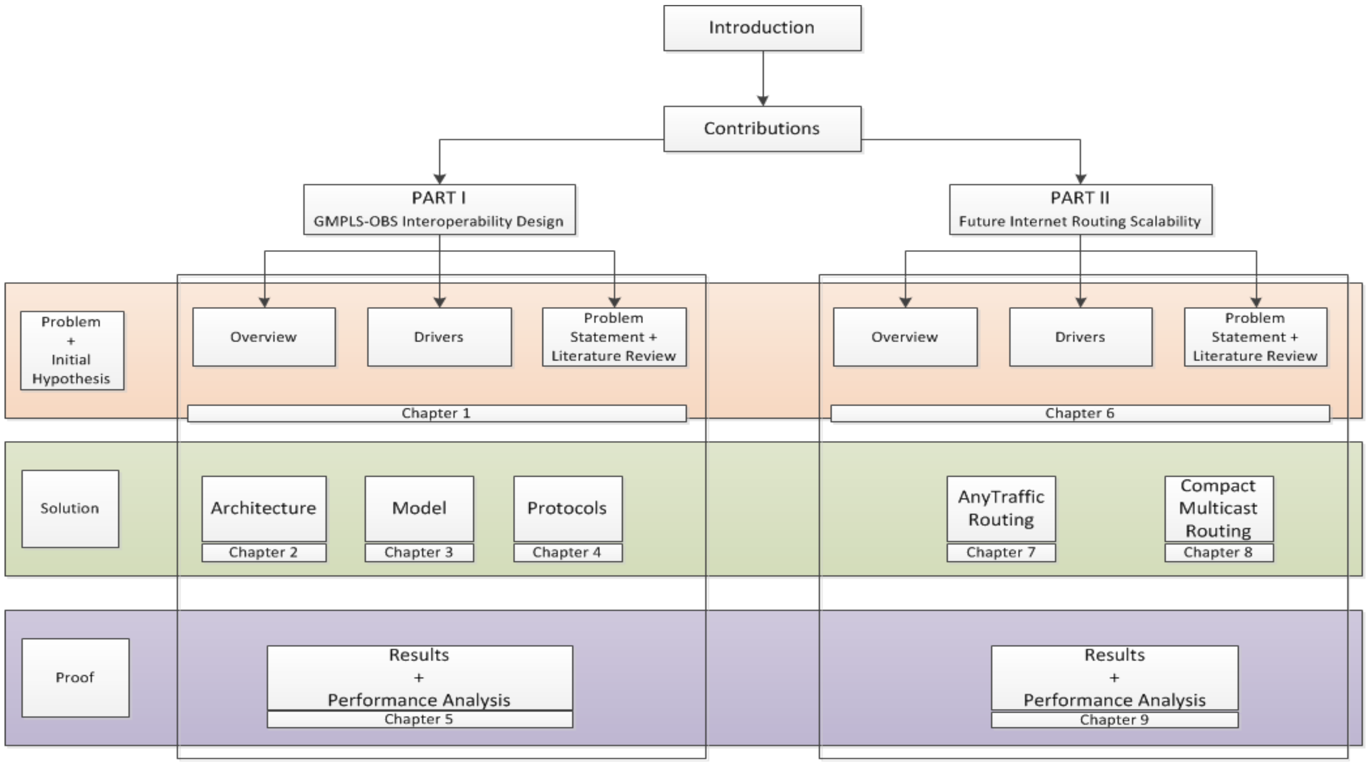

Introduction
Contributions
PART I
GMPLS-OBS Interoperability Design
PART II
Future Internet Routing Scalability
Problem + Initial Hypothesis
Overview
Drivers
Problem Statement + Literature Review
Chapter 1
Overview
Drivers
Problem Statement + Literature Review
Chapter 6
Solution
Architecture
Chapter 2
Model
Chapter 3
Protocols
Chapter 4
AnyTraffic Routing
Chapter 7
Compact Multicast Routing
Chapter 8
Proof
Results + Performance Analysis
Chapter 5
Results + Performance Analysis
Chapter 9

Figure 3: Thesis organigram.

OBS Interoperability Design - and Part II - Routing Scalability in Internet -. Horizontally, it consists in three linked levels (rows), shared by both Parts: we start with 1) the Problem Formulation/Initial Hypothesis, which comprises the description of the drivers that led to the problem and the statement of the problem itself; we then follow to 2) the Solution, where architectures, models, protocols and algorithms are presented; and we end up with 3) the Proof/Analysis of the previous solutions.

Part I refers to the GMPLS-OBS interoperability design and comprises five Chapters:

- ✓ Chapter 1 is the topic overview. It describes the GMPLS-OBS environment, highlights the drivers of the problem of designing such GMPLS-OBS control plane and concludes with the identification of the points to be addressed together with some literature review.

- ✓ Chapter 2, 3 and 4 are part of the Solution. The solution is structured as follows: Chapter 2 is the proposed GMPLS-OBS architecture (or skeleton), Chapter 3 is the control model operating in such architecture (or internal system), which also includes the routing and wavelength assignment policies (or brain); Chapter 4 is the protocol extensions that will rule such control model.

- ✓ Chapter 5 comprises the results achieved and the performance analysis. Different optical network topologies (up to 50 nodes) are considered as well as different traffic scenarios (static and dynamic).

Part II refers to Routing Scalability in Internet and comprises four Chapters:

- ✓ Chapter 6 consists in the problem description: we start by explaining the Internet inter-domain routing environment, what are the drivers that have led to the problem of routing scalability and finally we enumerate the points to be addressed and a literature review.

- ✓ Chapter 7 and 8 represents the solutions proposed. Chapter 7 deals with the new AnyTraffic Labeled routing concept and Chapter 8 with the name-independent compact routing concept aiming at Multicast and AnyTraffic scenarios.

- ✓ Chapter 9 comprises the simulation campaign and the performance analysis. Different graph topologies (up to 32k nodes) are considered as well as both static and dynamic scenarios.

Part I

GMPLS-OBS Interoperability Design

Chapter 1

Overview

The GMPLS and OBS interoperability design has been dragging on for quite a long time. The first general attempt dates from the year 2000 and since then no major works have ever come up with significant neither functional or operational proposals, nor performance analysis. Despite of it, there are countless reasons to believe in such technologies and therefore in any interoperability endeavor.

By bringing intelligence to the networks (i.e. automatic control and service provisioning operations), the GMPLS framework is the key enabler to orchestrate a unified optical network control and thus reduce network OPEX, while increasing operator's revenues [Pioneer02a][Pioneer02b]. The GMPLS is now mature - is well studied and standardized [RFC3945], and can be easily extended by IETF whenever new requirements arise - and its deployment seems almost inevitable in the upcoming years [Morrow05]. Its main advantages are that it is based on existing and widely deployed (IP) protocols at the same time it simplifies the network management by offering fast and flexible service provisioning over different switching domains - such as MPLS, Ethernet, SONET/SDH, wavelength or fiber switching - in a unified way. Alcatel-Lucent and Telefonica are among those operators and network providers interested and working on such distributed control plane. Ciena, Fujitsu and Huawei are world leader companies that have also shown interest (and some of them are actually working) on an optical control plane envisioning packet/optical convergence.

Recently, the OBS technology got an important boost with strong investments made in the industry. The InTune Networks company has invested $32 million to deploy commercially available OBS networks. Also, an FP7 Research European Project called Metro Architectures enabling Subwavelengths (MAINS), was launched to study the OBS architecture (eventually controlled by GMPLS).

In this context, OBS has been catching up as a promising short/mid-term solution to reduce the gap between switching and transmission speeds. Optical Packet Switching (OPS) is the natural extension of legacy packet-based networks to optical domain, yet it is still facing significant costs and technological hurdles (e.g. optical buffering). Hence, OBS, which precisely avoids optical buffers, is one of the well positioned switching technologies to realize the envisioned IP-over-WDM network architecture. It is a far more efficient optical switching layer than the today's too rigid and rather time-costly circuit-based optical transport (e.g. SONET/SDH or OCS) in order to serve the dominant IP/Internet packet-switched traffic. Its deployment is specially desired for Metropolitan Area Networks (MAN), where the traffic demands are highly dynamic.

OBS leverages on the statistical multiplexing of data plane resources to enable sub-wavelength switching in optical networks [Qiao99][Chen04]. This is attained by aggregating packets arriving at ingress nodes into bursts, which are sent to the destination over a buffer-less optical network infrastructure. High bandwidth utilization is achieved in OBS by means of one-way resource reservation schemes, which minimize the overhead introduced during the burst signalling process. However, these schemes alone do not ensure successful burst delivery as, due to the statistical multiplexing of the wavelength channels, bursts may contend and some be lost.

Despite the GMPLS principle of unified control, little effort has been put on extending it to incorporate the OBS technology.

The objective of the Part I is to contribute with a fully detailed GMPLS-OBS control architecture and model. From a functional perspective, the proposed solution tries to overcome those technology-specific interoperability issues. From an operational perspective, it offers (absolute) QoS guarantees to OBS networks together with an optimized resource usage. Keys extensions to the GMPLS protocol standards are equally suggested.

In this Chapter, we start by highlighting the main drivers of the GMPLS-OBS interoperability endeavor and then formulate on the problem: how to integrate OBS in a GMPLS-controlled multi-domain scenario and to identify the GMPLS and OBS constraints on signalling and routing procedures. Literature review is performed, pointing out what has been done so far. In the following Chapters, we elaborate on the architecture and model solutions, the protocol extensions of the GMPLS-OBS interoperable control plane and present the promising results achieved and its performance analysis.

Note: It is not our intention to elaborate exhaustively on the benefits of both OBS and GMPLS (well-studied topics) but rather to highlight and assess the benefits of the technologies' integration itself. Thus, we would to point out the reader to the following references:

- ✓ [Pasqualini05] stating the GMPLS economic reasons;
- ✓ [Morrow05] addressing the current impasse of GMPLS deployment;
- ✓ [Zalesky09] discussing if to do burst or circuit switch;
- ✓ [Rosberg03] presenting a performance analysis of OBS networks;
- ✓ [Sriram03] highlighting the benefits and challenges of OBS.

1.1 Drivers

Both GMPLS and OBS technologies are part of the set of solutions for an IP-over-WDM architecture. On the one hand, GMPLS is a well-established control framework able to introduce traffic engineering, constrained routing and automatic bandwidth provisioning over heterogeneous networks. In other words, it provides the so required intelligence in the control and management of resources as well as protection and restoration mechanisms. On the other hand, OBS is an all-optical switching layer oriented to packet transport, eliminating redundancy in IP transport and reducing operational costs. It takes advantage of the statistical multiplexing paradigm to provide cost-effective and dynamic use of network resources (sub-wavelength granularity), which translate into good network resource access and usage.

Currently, both technologies cannot coexist as GMPLS is not adapted to handle OBS. The challenge remains to provide GMPLS - widely accepted as the standard control plane for the Future Optical Internet - with proper extensions to handle one of the most promising and well positioned switching technologies, OBS - missing a well-defined control plane offering strict QoS provisioning and operating with other switching domains already controlled by GMPLS. The generic character of GMPLS and the data/control plane separation shared by both technologies has opened a space to research.

- ✓ First, the OBS network intelligence (i.e. all carrier-class control features being deployed in the last years) can be largely "moved" to the GMPLS control plane (i.e. it already provides them), thus keeping the OBS layer as simple as possible - following what has been done to some other technologies as e.g. ASONs [Jajszczyk05] or carrier-class Ethernet networks through the GELS framework [Takacs08].

  Note that GMPLS is an intelligent control layer *per si*, provided of proper control mechanisms as mentioned before. In such a way, most of its features can be easily applied to OBS control (and in addition suppress some its current flaws) without incurring in an increase of complexity to GMPLS. In fact, very few extensions are required to the GMPLS standard to support OBS. Moreover, GMPLS guarantees control scalability (e.g. link bundling and unnumbered links mechanisms) and resilience (e.g. protection and restoration mechanisms) and provides a protocol to detect faults in transparent networks (e.g. link management protocol (LMP)).
- ✓ Second, from the network interoperability point of view, the proposed interoperable GMPLS/OBS con-

trol plane can seamlessly enable the coexistence and easy migration between optical circuit-switched and packet/burst-switched networks as well as speed up the OBS development and standardization. In other words, horizontal and vertical OBS integration with other optical switching technologies to which GMPLS is already mature could be achieved effortlessly. This would pave the way towards future multi-service optical networks, offering wavelength and sub-wavelength transport services in the same transport network infrastructure.

Recently, the IETF has launched a new RFC dealing with the GMPLS automated and unified control and management over multi-layer/multi-region, i.e., multi-domain networks [RFC5212]. This is a strong argument to the interoperability. Service provisioning over multi-domains controlled by the same GMPLS instance could be achieved even when OBS is present, e.g., an LSP starting in an IP/MPLS domain, crossing OBS and terminating again in an IP/MPLS domain - see Fig. 1.3.

✓ Finally, the QoS support of the OBS network, which results in complex mechanisms and hard to be adopted at OBS switches, can be drastically improved by making use GMPLS TE features, committed by design to provide QoS guarantees to the supported traffic (e.g., TE LSPs). Therefore, by extending GMPLS to also encompass OBS networks, the setup of end-to-end TE LSPs matching specific QoS figures can be envisaged. The latter will also help to position OBS in the GMPLS framework.

In a nutshell, we aim to define a GMPLS-controlled OBS network both from a functional (architecture) and operational (model) perspective. The wide applicability range of the GMPLS' features and its generic character, leads to a natural process of convergence of these technologies and to a seamlessly coexistence and migration. It also contributes to speed up the OBS CP standardization. The GMPLS-OBS interoperability has strong arguments, yet it brings several difficult and challenging issues as we describe along the Chapter.

1.1.1 OBS: Control Requirements

OBS technology is based on a cost-efficient and dynamic provisioning of sub-wavelength granularity thanks to the statistical multiplexing paradigm, i.e., sharing of resources by different traffic flows. It also means reducing the bandwidth provisioning timescale from that of long-lasting circuits to that of bursts and giving more bandwidth flexibility in order to better respond to the changes in traffic pattern. Table 1.1 summarizes the OBS advantages and drawbacks.

Due the buffer-less nature of OBS (i.e. absence of optical buffers), the data burst is not buffered at any intermediate node achieving all-optical data transmission. Although this is an important advantage concerning the actual optical memories limitations, it also leads to problems like high burst losses as the amount of data increases.

Table 1.1: OBS advantages and drawbacks

Advantages	Drawbacks
·Short-term tech. availability and no need of optical buffers	·High burst loss probability due to buffer-less nature
·High bandwidth transport service at optical layer fitting packet-switched traffic	·High control complexity (signalling, routing, QoS, scheduling, protection) at the switching layer
·High network resource usage due to statistical multiplexing	

In such a way, the control information travels in advance in the OBS CP, separated from the data payload (i.e. burst) to do the configuration of the nodes along a path. This information is carried out in specific (burst) control packets (BCP), signalling several data packets (as one burst comprises several packets). Note that the offset time, which is the time difference between the release of the control and data information, must be larger than the total processing time of the BCP along the path. Thereby, the OBS CP is merely a packet-switched network, which controls the routing of data bursts in the optical network based on the information carried in the BCPs. Such operations require high control complexity. Thus, the OBS control plane should be capable of:

✓ 1) fast reconfiguration not only by the demand of an operator but also by customer request.

✓ 2) achieve efficient use of bandwidth.

✓ 3) fast forwarding (low latency of burst transfer).

✓ 4) high degree of transparency.

One of the most desirable features is the dynamic and fast provisioning of end-to-end connections in response to short burst transfers, i.e., the OBS CP must have the same highly dynamic character of the OBS DP in a way to create, reconfigure, modify and tear down connections according to traffic oscillations. Therefore, an adequate signalling scheme is required to perform it without long delays. The concept of one-way reservation (or Tell-and-Go (TAG)), which is highly desired for OBS networks, has shorter setup time and better throughput performance than two-way reservation concept (or Tell-and-Wait (TAW)). For instance, if an acknowledge (ACK) message of the BCP must be waited, the latency in the burst transmission (typically several milliseconds ms) will be too long compared with the sub-millisecond (μs) burst duration. This brings several challenges as high burst loss probability during highly loaded periods, which is the major drawback of such concept.

- ✓ The QoS figures guaranteed by OCS networks are not accomplished in pure OBS networks. In turn, the strategy must lay down on robust but simple routing methods to forward the bursts with support of TE and QoS provisioning. These routing strategies must avoid or limit the contention problem and do an optimal network usage. The high traffic demand and the consequent scarcity of resources appeals for a good distribution of the traffic over the available resources. In addition, an efficient management of the offset time must be done whose incorrect estimation may also produce burst losses.

- ✓ The topology and resource information dissemination: how to collect and distribute accurately "current" information about the network's resource availability is still an open question. Such parameters must still be defined.

- ✓ Standard BCP format is still missing. It should be as light as possible in order to guarantee fast processing at the OBS control unit. However, it must contain enough information to allow the burst switching, e.g. burst offset-time, burst duration, the incoming wavelength (of the burst), and something indicating the QoS requested. A possible format is suggested in Chapter 2.

It is no less important to refer that such OBS CP should implement burst assembly, contention resolution, QoS provisioning and protection and restoration mechanisms to ensure a well-built control plane. Although there are already several carrier-class features targeting at these problems, all of them are basically provided entirely from the OBS layer, which burdens the OBS core nodes with complex decisions compromising the fast OBS performances.

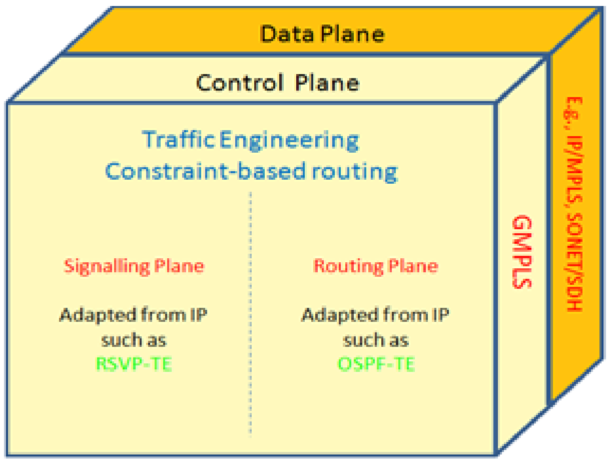
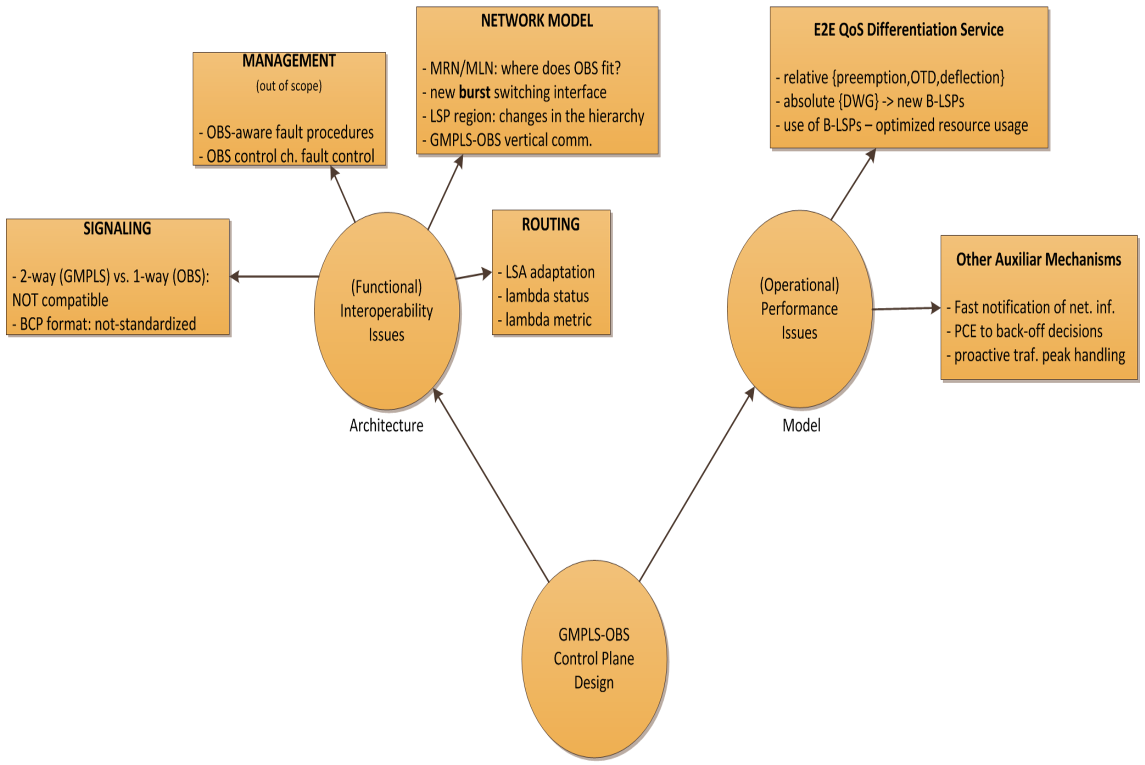


Figure 1.1: GMPLS framework: set of protocols.

1.1.2 GMPLS: the inevitable choice

The GMPLS provides a unified control plane using the same set of control features over various switching domains. Its TE mechanisms have been introduced to improve the control and management of the network as a whole. It makes use of IP-based protocols, which is one of its best advantages, performing signalling, routing, link management and fault protection and restoration. In terms of signalling and routing, we will only consider the Resource Reservation Protocol (RSVP-TE) and the Open Shortest Path First (OSPF-TE) protocols as they are the most commonly used and supported by nearly all the vendors (see Fig.1.1). In terms of management, GMPLS introduces a new protocol called Link Management Protocol (LMP) to provide control channel management and TE link connectivity verification between pair of GMPLS nodes. GMPLS is mature and widely accepted as the standard control plane fulfilling the ASON requirements for next generation optical control networks. Several advantages could be pointed out from the decision to use GMPLS to handle OBS networks.

The common premise of separation of CP/DP is one of the major convergence points between GMPLS and OBS. This similarity makes the actions taken at the CP independent from those at the DP, which helps the interoperability pro-



NETWORK MODEL
- MRN/MLN: where does OBS fit?
- new burst switching interface
- LSP region: changes in the hierarchy
- GMPLS-OBS vertical comm.
E2E QoS Differentiation Service
- relative {preemption,OTD,deflection}
- absolute {DWG} -> new B-LSPs
- use of B-LSPs – optimized resource usage
MANAGEMENT
(out of scope)
- OBS-aware fault procedures
- OBS control ch. fault control
SIGNALING
- 2-way (GMPLS) vs. 1-way (OBS): NOT compatible
- BCP format: not-standardized
(Functional) Interoperability Issues
ROUTING
- LSA adaptation
- lambda status
- lambda metric
(Operational) Performance Issues
Other Auxiliar Mechanisms
- Fast notification of net. inf.
- PCE to back-off decisions
- proactive traf. peak handling
Architecture
Model
GMPLS-OBS Control Plane Design

Figure 1.2: Issue Tree of the GMPLS-OBS CP design.

cess. There is no need of any kind of intervention and modifications at the DP, which is transparent from the CP point of view. In a general sense, the interoperability between GMPLS and OBS features is straightforward.

GMPLS is a powerful control tool that may supply OBS with some features like automatic end-to-end provisioning of connections and automatic management of network resources. Such operations have an important economic impact by reducing considerably the current manual, slowly and costly service provisioning. The GMPLS deployment fosters the IP-over-WDM architecture, "allowing" the removal of legacy and redundant protocol layers. It congregates in itself the main functionalities of those layers, making the IP packets to travel directly over the optical transport network, which decreases the network complexity significantly.

The important scalability issue in large networks as optical backbone networks, where hundreds to thousands of wavelengths transporting user data are expected, can be improved using the new features available in GMPLS: link bundling and unnumbered link mechanisms accommodate the changes in the network in a quick and graceful mode. Also, it is worth noting the feasibility achieved by automated fault management in network operations. Those mechanisms like protection and restoration contribute to a successful operations of the CP in such highly demanding networks like OBS. Moreover, GMPLS LMP protocol offers a unique feature to localize faults in transparent networks.

Last but not least, the integration will speed up the process of developing and standardizing the OBS technology as allowing a smoother migration from OCS to OBS networks in a way that GMPLS is the standard CP for the state-of-the-art optical networks. Also, a normal coexistence between OCS and OBS networks can be achieved. The several matches observed between what OBS CP needs and what GMPLS offers makes the interoperability a natural step.

1.2 Problem Statement and Literature Review

There is a considerable amount of OBS-related works assuming a GMPLS-controlled OBS scenario in literature, albeit little work is to be found dealing specifically with the GMPLS-OBS interoperability design. The initial proposal on the topic, the so-called Labeled OBS (LOBS), was suggested by C.Qiao in [Qiao00] in 2000, and later extended by K.Long et.al in [Long06] in 2005. Also A. Manolova has come out with some slightly different ideas in [Manolova07] in 2007. Nevertheless, all of them approach the topic from a very generic perspective, focusing basically on general architecture integration aspects, but failing to detail and to define proper control models, mechanisms or extensions to the GMPLS standards to handle the OBS technology. Moreover, network performance studies are still missing to prove both the feasibility and reliability of GMPLS-OBS architectural model. Following the issue tree depicted in Fig. 1.2, we aim to address the GMPLS-OBS control plane design from both a functional and operational perspectives.

Functional issues (architecture): issues concerned with the architecture. Thus, we have (vertical) interoperability issues faced when attempting to integrate both technologies, as well as (horizontal) interoperability of OBS with other switching domains under the control of the same GMPLS instance. This requires the definition of a proper GMPLS network model where OBS is present, as well as signalling and routing adaptations to GMPLS and OBS protocols. Only one single control instance of GMPLS is considered (e.g. one autonomous-system (AS)). Minor, straightforward adaptations are required to fault recovery and link management mechanisms. However, the management topic is out of the scope of this thesis. Below, the list of the tasks to be performed:

- ✓ Signalling procedure approach (OBS one-way mode vs. GMPLS two-way mode).
- ✓ Cooperative fully described network architecture and framework.
- ✓ Node functional architecture and collaborative control operations between GMPLS and OBS.
- ✓ GMPLS protocol enhancements (extensions).
- ✓ Adaptation of fault recovery and link management mechanisms.
- ✓ Format of control and data information packets.

Operational issues (model): issues concerned with the OBS performance itself. In general, OBS is a buffer-less technology and OBS networks belong to the class of loss networks [Rosberg03]. This way, the bursts may contend for link resources at core switching nodes and the contention when unresolved leads to burst losses. The problem of burst contention is of fundamental importance in OBS networks. A plethora of techniques have been proposed to upgrade OBS networks with carrier-class features. Nevertheless, OBS still presents high burst losses due to contention, especially in high loaded scenarios, and lacks strict QoS guarantees. Moreover, all the proposed techniques are provided entirely from the OBS layer which burdens OBS core nodes with complex decisions that should be typically made on a per-burst basis, compromising the fast OBS network performance. Below, the list of the tasks to be performed:

- ✓ Optimized route (i.e. LSP) computation together with good network usage aiming at loss reduction.
- ✓ Provisioning of absolute QoS guarantees.
- ✓ Global network status information dissemination.

There are significant work done pointing at OBS routing, contention resolution and some at QoS provisioning, but all of them are suggested to be provided and implemented from an OBS DP/CP perspective. Recalling once again that the idea is to move all these control mechanisms, i.e, the intelligence, to GMPLS to achieve the aforementioned benefits, details on specific GMPLS-OBS integration procedures, network framework, protocols extensions and performance results are still required.

Following the recent survey done by Manolova [Manolova10], we discard the overlay models [Guo07] (either client or server approaches) by considering that they not address the key problem of GMPLS-OBS interoperability design. None of them provides a control solution to OBS as they not use GMPLS for actual OBS control, and fails to cope with both the OBS statistical multiplexing property and the one-way signalling mechanism. Thus, we focus exclusively on the MPLS-like and integrated models.

However, we hold off from those models by providing strict QoS provisioning together with optimized resource usage, and preserving the statistical multiplexing of resources at the same time. OBS must definitely have somehow a connection-oriented behavior to be acceptable. For such reason, we keep the use of the GMPLS LSP concept and aim to prove its benefits in OBS network. Note that although the resource reservation is maintained hop-by-hop by OBS signalling, a

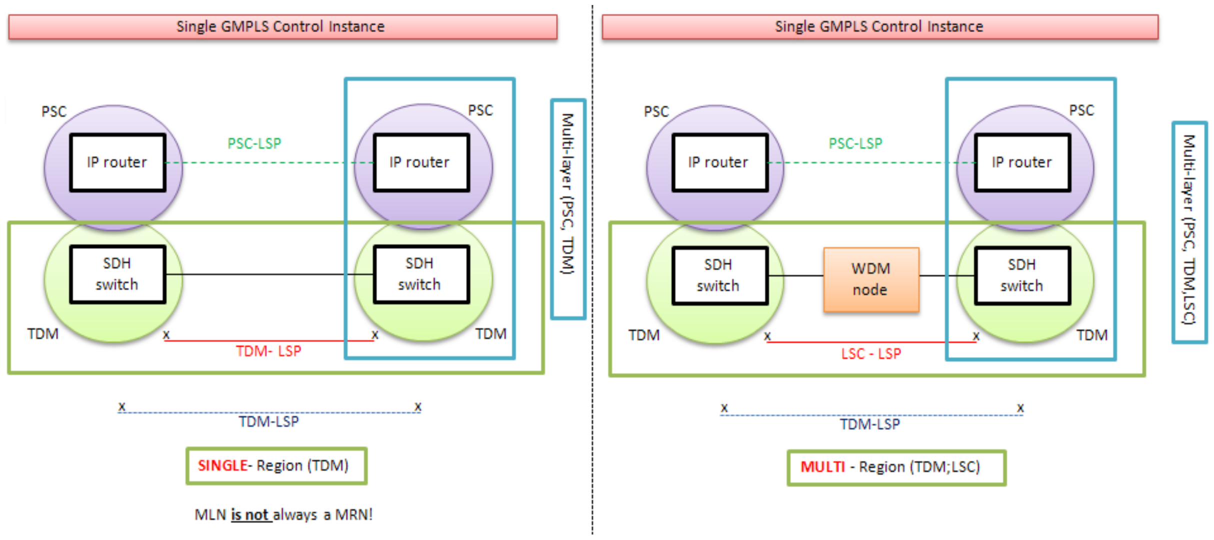


Figure 1.3: MRN/MLN possible scenarios with OBS switching domain.

modified GMPLS LSP provisioning (without resource reservation) will permit to make use of TE techniques that will provide QoS guarantees and improve OBS overall burst loss performance.

Our contributions can be then summarized as the proposal of a solid and scalable architecture for the OBS control plane, gathering the best of the aforementioned related work; definition of key concepts, interfaces and logical relationships, and providing the missing numerical studies that will definitely boost this relevant topic.

1.2.1 Network Model

Vertical Integration: A fully-descriptive network model is required to define the relationship between GMPLS and OBS. From the data perspective, the control plane is due to be "seen" as a single and unified control instance. Albeit, from the control perspective, it will consist in a "dual-layer" control architecture/plane, which will share such such control operations according to time requirements. The data plane may have a multi-granular data switching architecture allowing either circuit (OCS), burst (OBS) or even packet (OPS) optical switching, requiring efficient and fast signalling and routing processes. However, we consider only the OBS switching domain. The problem itself resides on the interoperability of the dual-layer control plane, while the data plane remains independent of the control model devised. As said before, such separation is already embraced by both GMPLS and OBS standard architectures.

Horizontal Integration: The desired of multi-switching domains controlled by GMPLS is a strong argument in favor of the GMPLS-OBS interoperability. In a peer-to-peer or augmented control plane model supported by GMPLS, where the routing information of different domains can be (fully or partially) shared, it would allow the setup of end-to-end (e2e) LSPs crossing multi-switching domains, event when OBS would be present. Figure 1.3 shows several possible scenarios and the different LSP types. For instance, if a OBS node would be presented instead of the WDM node (i.e. OCS) in the right hand side of Fig.1.3, the e2e LSP connectivity would be not possible. Recently, the IETF has launched a new RFC [RFC5212], describing such heterogeneous end-to-end processes as well as the proper GMPLS extensions [RFC6001]. It defines Multi-Region network (MRN) as a set of regions, where a Region is a switching technology domain and Multi-Layer network (MLN) as several layers, where a Layer is the data plane switching granularity level. However, it does not define how to operate when crossing an OBS switching domain. So far, the literature has failed to address this topic.

1.2.2 Signalling Interoperability

Problems and Requirements The signalling of the OBS resource reservation process is the main reason for the controversy around the GMPLS-OBS interoperability. While GMPLS Signalling is based on a two-way protocol, the OBS Signalling only requires a one-way (on-the-fly) protocol, which is one of its milestones. Thus, *how to set-up a GMPLS Label*

Switched Path (LSP) in a burst switching environment?

From an OBS perspective, the interoperable signalling model must preserve the statistical multiplexing property. From a GMPLS perspective, it should maintain GMPLS technology-independent as much as possible, i.e., interoperability and further extensions must not compromise the overall GMPLS applicability to other switching technologies, while guaranteeing QoS provisioning.

The GMPLS-OBS signalling is the major barrier in the entire design and it remains undefined. The current approaches that can be found in the literature fall into two major categories: i) use GMPLS signalling as an auxiliary to perform OBS signalling, as initially suggested by C. Qiao in [Qiao00] and reinforced by K. Long in [Long05a][Long06]; and ii) the removal of LSPs and their establishment and modification of GMPLS Signalling messages to operate in one-way mode as burst control packets (BCPs), as suggested by A. Manolova in [Manolova07]. Both types of approaches are discussed below:

i) GMPLS signalling as an auxiliary to perform OBS signalling: this has been commonly named LOBS. Under this proposal, the control of OBS is based on a hybrid CP, where GMPLS performs the label distribution and the setup of a virtual topology (VT) according to traffic demands and network status, while OBS performs the actual signalling for resource reservation. However, how to actually do it and, moreover, provide strict QoS and acceptable burst loss levels, is still an open question. No models or performance analysis have been ever published.

C. Qiao was the first to propose it [Qiao00]. The so-called LOBS architecture provides the basic ideas for the collaboration between OBS and MPLS/GMPLS, suggesting the creation of a VT of LSPs over the OBS network. It states that a label and a wavelength association must be done only at the burst time scale instead of at connection time scale, i.e., no strict wavelength reservation is performed. This idea of pre-computed LSPs (i.e. VT) without binding them to physical reservation, is the starting point to go forward with the interoperability design. However, this proposal lacks on details about a specific integration strategy. It does not show how it contributes to give any QoS guarantees or reduce the burst losses in OBS networks. Moreover, the LOBS model can be seen as an MPLS control plane falling the MRN/MLN purpose mentioned before (see Fig.1.3). In [Qiao00], Qiao pointed out several issues to be solved that still remain without an answer today: better characterization of LOBS paths, how to extend MPLS/GMPLS and how to overcome the global network status information.

K. Long goes further by proposing a more detailed architecture [Long06]. He has even submitted an IETF draft identifying in a generic way the needed GMPLS extensions to OBS, the first and the only one so far [Long05b]. The collaborative control operations between GMPLS and OBS are better described than before. A node architecture is described as well as a possible GMPLS/OBS control software modules. However, once again, how the path computation should be done remains unclear and how to address the QoS provisioning is not mentioned.

ii) Removal of LSP procedures: as suggested by **A. Manolova** in [Manolova07], GMPLS framework can be used as a complement to missing functionalities of OBS. The LSP procedures are disabled and the BCPs are not labeled. In fact, only routing (OSPF-TE) and management (LMP) features are used. This makes such approach not fully GMPLS compliant. The removal of the LSP concept reduces the TE and QoS capabilities of GMPLS. Although the proposed architecture results in simpler implementation, the OBS control plane is burden with bigger BCPs (carrying all the routing and signalling information), as they add too much overhead and require longer time processing. TE features are much less which difficult the traffic handling. It is also suggested that the RSVP-TE can be used as the one-way OBS signaling. This would imply considerable changes to the protocol and its consequences are not clear.

Summary - protocol extensions or modifications - path computation model guaranteeing strict QoS.

1.2.3 Routing Interoperability

Problems and Requirements On the contrary, the routing is the minor issue of the entire design process. GMPLS routing is the way that network status information is passed to perform path routing computation. As the general GMPLS framework, the GMPLS routing protocols are not OBS-aware, i.e., they need protocol extensions to advertise OBS-related link state. Therefore, extensions to standard link state advertisement (LSA) objects of such routing protocols are required to support OBS-related network status information. The parameters to be flooded must be defined. None of the previous works in the literature has addressed such problem.

Moreover, we still have the problem of global information dissemination as mentioned before by Qiao in [Qiao00]. Routing inaccuracy can severely incur on contention situations, with the degradation of the OBS performance. Distributed routing protocols such as (GMPLS) OSPF-TE or IS-IS can be too "slow" for an OBS environment, especially in large optical networks. "Real-time" information of network resource availability is required to cope with the burst dynamic when computing the path routing of the LSPs. However, how to timely flood the network state information is still an open question.

The solution can be either use a centralized node with a Path Computation Element (PCE) - highly appreciated by network operators - or deploy local mechanisms to solve short-term traffic increments. In addition, a specific notification mechanism may be deployed at the optical level in order to reduce the overload of routing messages (e.g. including some status information in the BCPs), however, the time-issue of the link-state distributed protocols will remain.

Path Computation Element The PCE may be deployed in a centralized manner, sniffing the routing messages and acquiring the full topology view [RFC4657]. This way it can compute accurately the path of each LSP. Nonetheless, this approach has some drawbacks. It reduces the scalability of such CP and will introduce latency in the traffic transmissions. A mixed scenario may also be considered, where the PCE would be placed both in a centralized node and distributed by every GMPLS controller. The path computation could be then switched from time to time between both modes in order to reduce the inaccuracy. So far, there is no work in the literature taking into account such PCE enhanced GMPLS/OBS network architecture.

Summary - protocol extensions or modifications - global network status information dissemination - PCE node incorporation

1.2.4 QoS provisioning in OBS

Looking at the literature, a plethora of techniques have been proposed to upgrade OBS networks with carrier-class features. The main targets of these contributions have been: i) performance, by proposing contention resolution [Yao03], burst scheduling [Barakat06] [Xu03], enhanced route selection [Klinkowski10] and avoidance techniques [Liu03] [Yang06]; ii) reliability, by presenting several protection and restoration schemes to increase OBS network resilience [Lim06] [Xin04]; and iii) QoS differentiation to support the wide range of currently available Internet applications, each one with different QoS requirements.

The recent survey on routing methods [Klinkowski10] shows that the problem of routing with QoS guarantees has not been studied widely in OBS networks. The existing solutions belong mainly to the class of alternative (deflection) routing. These reactive-based strategies employ adaptive methods that introduce relative QoS guarantees by the differentiation of routing decisions with respect to the QoS class, without ensuring specific QoS guarantees. We have relative QoS differentiation techniques such as Offset Time based Differentiation, which is probably the most common technique in OBS [Yoo00] assigning extra offset times to higher priority (HP) classes isolating them from the lower priority (LP) ones; or different sorts of preemption at core OBS nodes are proposed for these purposes [Kaheel03].

On the contrary, absolute QoS differentiation is committed to deliver quantitative QoS levels for high-priority traffic classes, even in highly loaded network scenarios. This is more attractive for upper layer applications, as transport services can be tailored to the specific performance requirements. In [Zhang04], two schemes are proposed: an early dropping scheme that probabilistically drops LP bursts to meet HP QoS requirements and a wavelength grouping scheme that provision the necessary wavelengths for HP traffic classes.

The concept of virtualization and, in particular, the joint problem of routing and wavelength assignment (RWA) with e2e QoS guarantees has not been considered in OBS to the best of our knowledge. Indeed, the common solutions for the e2e QoS provisioning are based on the shortest part routing assumption [Phuritatkul07][Yang07a][Zhang04], which control e2e burst losses by tightly dimensioning the number of wavelengths that supports the HP traffic.

In addition, all the aforementioned techniques are provided entirely from the OBS layer. As mentioned before, this burdens OBS core nodes with complex decisions that should be typically made on a per-burst basis, compromising the fast OBS network performance. In fact, none of them has considered GMPLS TE features (e.g. LSPs) to provide QoS figures, or achieved independent behavior in function of the load scenario with high scalability.

Chapter 2

(G)OBS Control Architecture

The architecture of a distributed computational system is formally described by a set of functions, objects/information, and state together with their behavior, structure, composition and relationships which characterize its domain of applicability. The specification of the associated functional and state models leads to an architectural model comprising a set of components (e.g. procedures, data structures, state machines) and the description of their respective behavior and structure as well as the characterization of their interactions (e.g. messages) - EULER project.

2.1 Principle of (G)OBS

The separation of (data/control) planes given by each technology is maintained at the proposed GMPLS-OBS architecture, or simply (G)OBS, and confines the interoperability problem to a control level. This results in an interoperable control plane (CP), which is composed by two control layers, namely GMPLS and OBS. Yet, the transparent, buffer-less all-optical OBS data plane (DP) will "see" a unique CP. Figure 2.1 illustrates the architecture

An effective and efficient control framework, where GMPLS controller "lies" on top of the actual OBS controller, is then achieved based on the sharing of control tasks among these two control layers, according to time scale demands. Those simple and local control operations (time scale varying in the order of microseconds/milliseconds) such as reading the forwarding table (FT) and selecting the outgoing wavelength from a given set for data burst transmission and local burst contention resolution (within such set) are kept at the switching layer (OBS). On the other hand, the overall network intelligence is concentrated to the GMPLS control layer. All complex operations (time scale varying in the order of minutes, hours, days or even longer) involving global knowledge of the network status are "moved" (i.e. GMPLS provides them) to the GMPLS control layer, such as traffic engineering and path computation, routing notifications, congestion resolution, QoS support, or protection/restoration actions.

The GMPLS may be deployed out-of-band, in/out-of-fiber, and supported by whatever technology and topology. On the contrary, in OBS networks, bursts and their related burst control packets (BCPs) must keep a strict time relationship in order to make one-way reservation feasible. Hence, it is mandatory that OBS control and transport planes share the same resources and topology. This is the reason why an in-fiber out-of-band control plane configuration (i.e. signaling channels) has been considered at the OBS layer - either manually or automatically configured.

The control sharing between both control layers helps to overcome the divergences identified in the previous Chapter 1, avoids the deployment of redundant mechanisms at the OBS layer and the burden of its control unit with increased complexity. At the same time, it fulfils the main requirements defined for a future optical network control plane (i.e. ASON):

i) fast provisioning by keeping the one-way signaling of OBS.

ii) easy, reliable and scalable network operation due to GMPLS features.

The control tasks themselves are performed subject to given traffic demands and QoS requirements with the assistance of

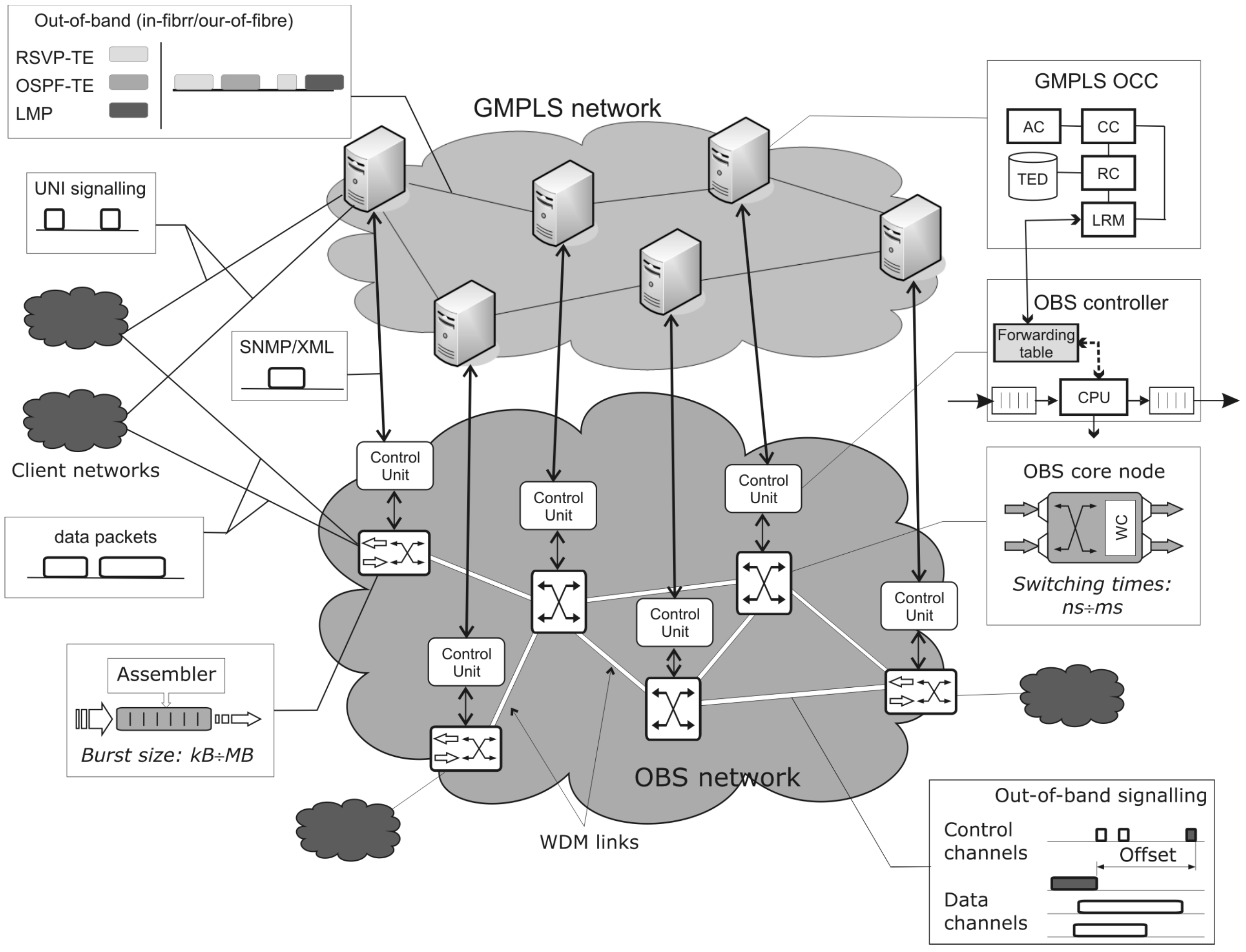


Figure 2.1: Generic Architecture of the proposed (G)OBS Control Plane.

properly defined TE rules, as will be explained in next Chapter 3 (control model). This requires some operational adaptations, although only few extensions (not modifications) to both technologies' standards (see Chapter 4).

This bipartite CP consists of a set of single node entities, each one with two dedicated controllers, one for OBS and the other for GMPLS, as shown further in Fig.2.4. Therefore, the novel (G)OBS node architecture must include a new communication channel as information needs to be exchange between both controllers. In order to keep track of the overall system's conditions (i.e. OBS DP), OBS controller sends resource usage information to GMPLS counterpart (e.g. output link usage), while GMPLS controller sends configuration messages to the OBS counterpart (e.g. forwarding table updates). In Section 2.3, we elaborate on the content of such messages as well as on the protocol to be applied to such communication channel.

Note that, although the proposed (G)OBS CP is intended to work in a distribution fashion, it may require a centralized path computation to a more accurate routing and better resource usage at the expense of slightly slow connection establishment (i.e. setup latency). Therefore, we also considered a centralized node with a PCE - highly appreciated by network operators -. Our intention is to further analyze the impact of such centralized path computation in the overall control model behavior. In this case, network configuration, topology and resource status information are kept at such node and continuously updated.

Connection Control Admission In the (G)OBS network, the GMPLS controller of the network edge node is the responsible for the connection control admission (CAC) by checking the client requirements and performs the path computation. A connection request may arrive through different ways, as illustrated in Fig.2.2. It may arrive through the sig-

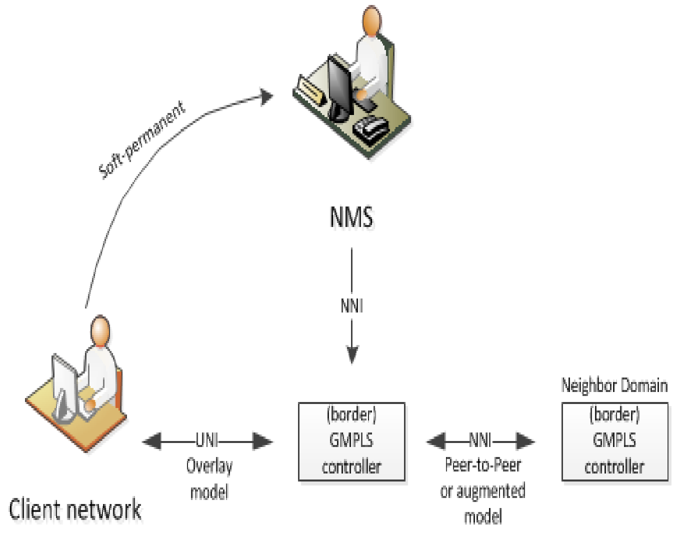

Soft-permanent
NMS
NNI
Client network
UNI
Overlay model
(border) GMPLS controller
NNI
Peer-to-Peer or augmented model
Neighbor Domain
(border) GMPLS controller

Figure 2.2: The different Connection Request Interfaces.

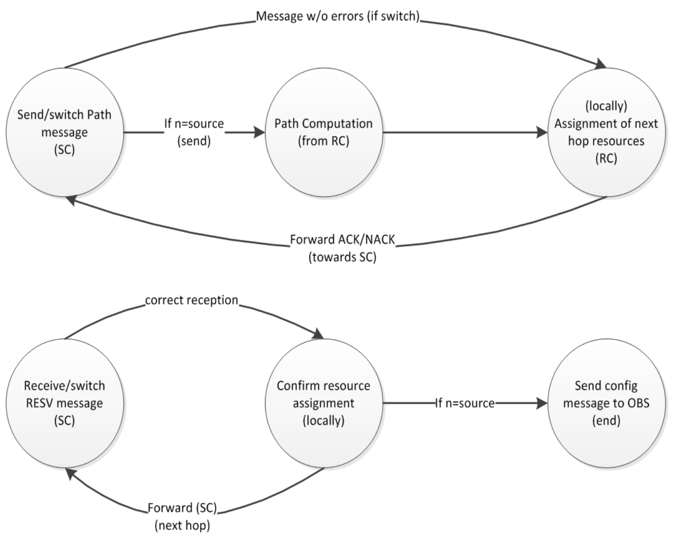

Message w/o errors (if switch)
Send/switch Path message (SC)
If n=source (send)
Path Computation (from RC)
(locally) Assignment of next hop resources (RC)
Forward ACK/NACK (towards SC)
correct reception
Receive/switch RESV message (SC)
Confirm resource assignment (locally)
If n=source
Send config message to OBS (end)
Forward (SC) (next hop)

Figure 2.3: GMPLS signalling procedure: finite state machine.

nalling user-network-interface (UNI) if the GMPLS overlay model is considered, where a client-server mode is applied, or through the external network-to-network interface (E-NNI) if the GMPLS peer-to-peer or augmented model is considered, where routing information is fully or partial shared among the different domains controlled by GMPLS control instance. In this case, the connection request arrives from a peer GMPLS controller of a neighbor domain. Another option is to have a network management system (NMS), operating within the management plane. The NMS is the responsible for the control of a client connection request (permanent connections) in a traditional transport network scenario. However, when a control plane is in place, a hybrid scenario can be considered (soft permanent connection). According to ITU-t Rec. G.8081 [ITU04], a soft permanent connection (SPC) is a combination of a permanent connection segment at the source-user-to-network side, a permanent connection segment at the destination-user-to-network side, and a switched connection segment within the core network. The permanent parts of the SPC are owned by the management plane, and the switched parts are owned by the control plane.

Network Operation A generic description of the overall (G)OBS working mode is given below. It is worth to mention that the proposed architecture was strategically designed to be independent from the model and the routing and wavelength algorithms deployed (see next Chapter 3.5).

This architecture diverges from other literature proposals like Wavelength-Routed (WR) OBS [Düser02] and [Sahara03], and approximates those under the LOBS concept [Qiao00] [Long06]. Here, we manage to keep a "pure" OBS network with one-way signalling working closely with an adapted GMPLS control layer, which remains two-way oriented.

The main difference is that this architecture does not consider physical reservation at the GMPLS signalling time. In other words, the GMPLS control layer is only in charge of establishing (logical) LSPs, assigning the proper set of resources to it at each node along the route according to the given operational control model in place. Hereafter, this type of LSPs are called burst LSP, or simply b-LSP, as they are extended to the transport of bursts (its meaning is fully described in Chapter 4).

In the context of our work, the terms assignment, commitment and allocation, have different meaning, depending if applied from a GMPLS or OBS perspective. From the GMPLS perspective, they mean to determine logically the resources per b-LSP and no physical reservation. From the OBS perspective, it means physical reservation of resources. The term reserve is exclusively used from the OBS perspective.

The b-LSP signalling procedure is then of responsibility of GMPLS and it is performed in a two-way mode. It can be described through a finite state machine, as shown in Fig. 2.3: the forwarding of the Path (top scheme) and the Resv (bottom scheme) messages of the GMPLS signalling protocol. According to the figure, if the node is the source, it first defines the path of the b-LSP. The path computation can be performed locally (distributed PCE) or by a centralized node (central PCE). In principle any path routing can be adopted ranging from simple shortest-path to complex optimization algorithms. In Chapter 3, we provide an example of optimized path computation for absolute QoS support.

Then, it locally assigns the proper set of wavelengths for the next hop. When the Path message reaches the destination node without problems, it sends back a Resv message towards the source. Otherwise, a PathErr is sent backwards (even before it gets to the destination, if it is the case), removing every temporary signalling instance created. At every node it confirms the b-LSP establishment and the assignment of resources. Simultaneously, the GMPLS controller sends a configuration message to OBS controller, containing the updated forwarding table (FT) - discussed in next Section 2.3. Note that only a single GMPLS signalling session is required to transmit all bursts belonging to a b-LSP, unlike other approaches in the literature [Qiao00][Long05a].

In turn, the OBS controller is the one that commits data plane resources for the incoming bursts. As in conventional OBS, once a burst is assembled and ready at the edge node, the corresponding BCP is firstly dispatched. However, the main difference here lies on the fact that BCP and burst are restricted to follow the same b-LSP. At each node along the path, the OBS controller processes the BCP, which contains the label identifying the b-LSP (the only routing information that it carries), and looks up the forwarding table to determine the output port. A scheduling is then executed to reserve the required output wavelength (from a limited set defined by the previous GMPLS signalling), switching the burst accordingly. Such committed resources are then released once the burst is completely transmitted. In this way, the statistical multiplexing benefit of OBS is preserved.

2.2 A Novel Node Architecture

The (G)OBS node is a logical entity composed by two dedicated controllers, which can be physically co-located or not. The GMPLS node controller is responsible to generate, transmit and process GMPLS-related messages that allow the setup, maintenance, reconfiguration, and tear-down of end-to-end connections, i.e. b-LSPs. In turn, the OBS node controller is responsible for the (data) burst generation (i.e. assembly) and transmission (switch matrix configuration), and for the processing of the BCP as in any conventional OBS network.

The new node architecture is depicted in Fig.2.4. It was designed following both the ITU-t G.8080 [ITU06] and IETF GMPLS [RFC3945] recommendations. The architectural components of this interoperable CP are represented by building blocks, which are combined in different ways upon the required functionalities. Below, there is the list of generic control operations required to both controllers.

GMPLS controller

- ✓ b-LSP setup and tear-down, modifications or re-routing (signalling and routing features)
- ✓ TE-database maintenance
- ✓ b-LSP monitoring
- ✓ control channel management (LMP protocol)

OBS Controller

- ✓ switch matrix configuration
- ✓ assignment of resources (i.e. wavelength reservation) - local wavelength scheduling
- ✓ FT maintenance (passed by GMPLS controller)
- ✓ periodically reporting about resource status (data link monitoring) to GMPLS controller
- ✓ classical burst preemption mechanism (if enable)
- ✓ assembly/disassembly (if edge node of a b-LSP)

2.2.1 Description of Building Blocks

In this subsection, each building block of each one of the controllers is bound by the description of its functionalities in the proposed (G)OBS architecture. The protocol controllers (PC) are omitted for simplicity.

i) At GMPLS controller:

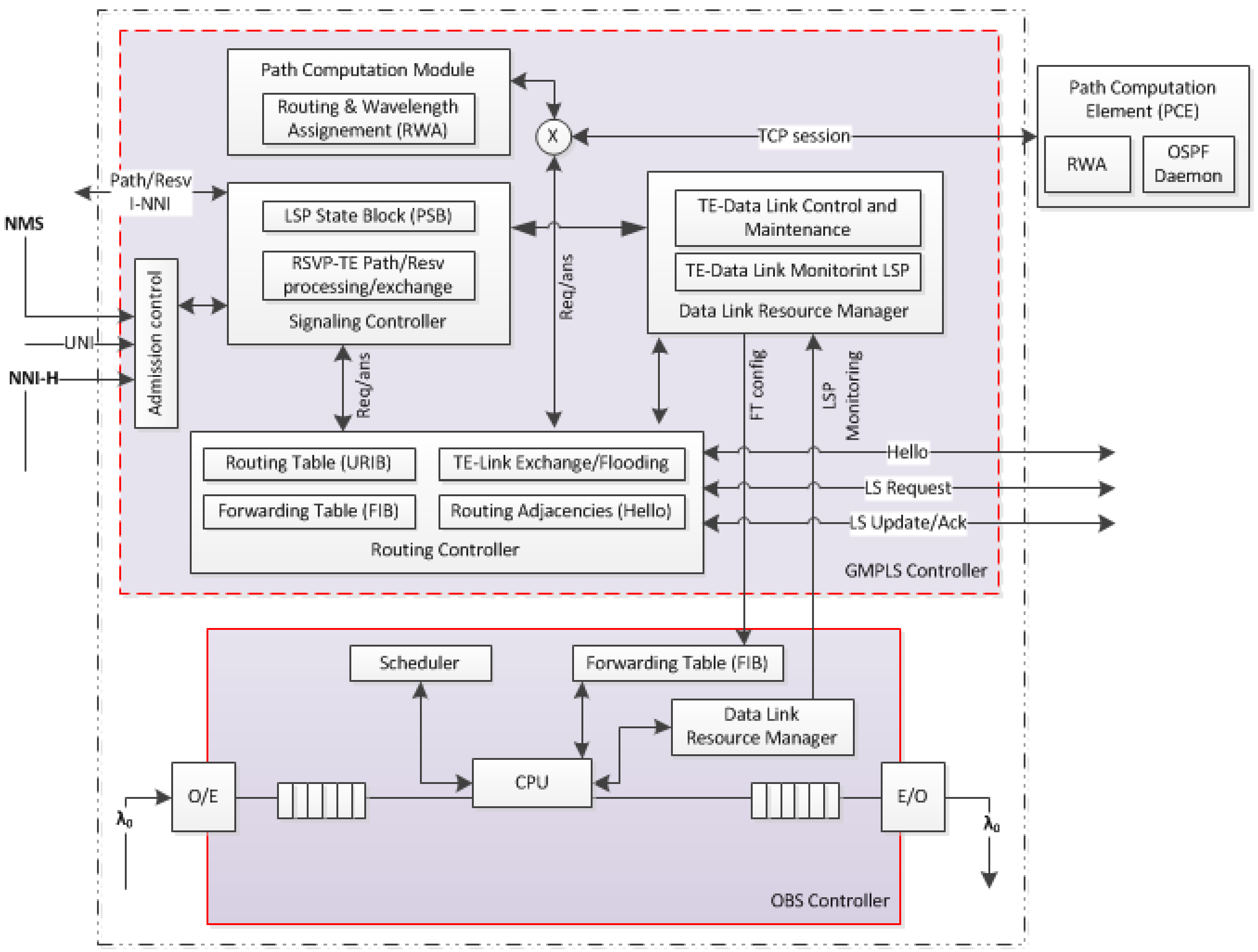


Figure 2.4: (G)OBS node building blocks.

Connection Admission Control (CAC): is essential as it determines if there are sufficient resources to admit a connection request. It may refuse it as well. This is usually performed on a link-by-link basis, based on local conditions and policies. In our case, traffic load and QoS requirements are considered as explained further in next Chapter. The admission of new connections is compatible with existing QoS agreements for existing connections.

Signalling controller (SC): either called connection controller (as defined by G.8080) or signalling controller, it is responsible for the management (i.e. establishment, maintenance and release) of b-LSPs and it coordinates the rest of the GMPLS components, namely RC and G-DLRM. The SC receives the Path/Resv messages and interacts with the CAC component to decide the admission of a connection request. It queries the other components in order to obtain i) path computation information and/or ii) network resource allocation status.

Routing Controller (RC): is responsible for network topology discovery and path computation. It comprises a PCE that performs routing and wavelength assignment at the CP level and responds to requests (from SC) for path (route) information needed to set up connections. This module can be deployed in-node (distributed mode) or out-node (centralized mode). In the latter case, a TCP session must be established (it can be established once and keep active for subsequent connection requests). This relationship is formalized by the PCE protocol [RFC5440].

It also comprises a routing engine (OSPF) responsible to disseminate network status information (TE-link information) and to do the network topology discovery. This component

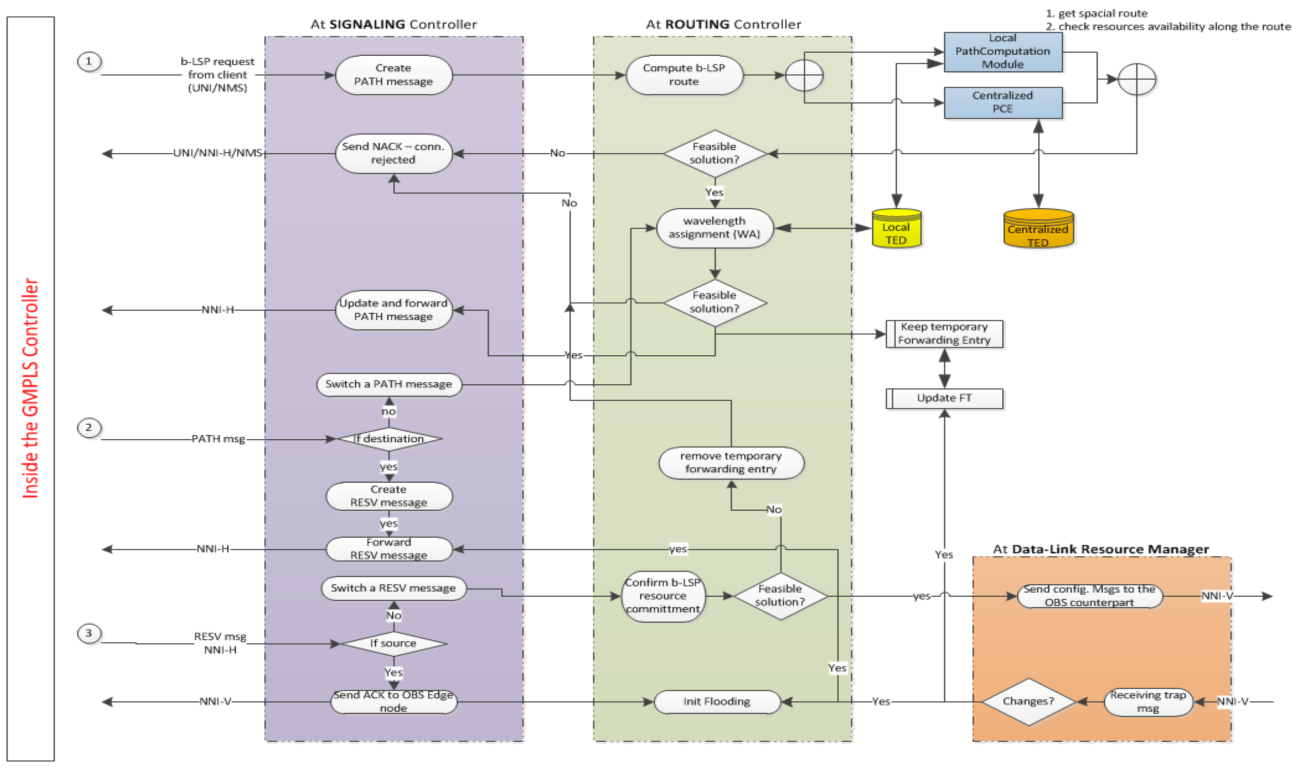


Figure 2.5: Flowchart showing the cooperation among the building blocks of the GMPLS controller upon b-LSP requests.

maintains both a RT and a FT. In such a way, it component is connected to a TE-database (TED) and to a link-state database (LSD). The TED stores all TE-metrics and TE-link information such as link residual bandwidth, b-LSP states (PSB), lambda status (given by the LSAs), SF metric, QoS; while the LSD the RT entries (e.g. link cost) - helps to compute the spatial route.

If the PCE is assumed to be in a network centralized node, then the routing engine as well as the aforementioned databases are also deployed in this node. A routing (OSPF) daemon will "sniff" the (OSPF) routing messages traveling across the network in order to acquire full topology view.

GMPLS Data link Resource Manager (G-DLRM): is responsible to maintain, on a per node basis, an updated view of the state and condition of the local data plane resources such as fiber and wavelengths (i.e. link). The G-DLRM is responsible to monitor the local optical resources in order to maintain the agreed services requirements. Such information is sent periodically by the OBS controller and saved in a Management Information Base (MIB) entity kept at this component (see Section 2.3). Upon changes in the FT (e.g. due to the set-up, tear-down or modification of a b-LSP), it sends configuration messages to the OBS controller containing its updates.

ii) At OBS controller:

Central Processing Unit (CPU): is responsible for the processing of all the tasks and coordination among all the components in the OBS controller.

Forwarding Table (FT): is received from the GMPLS controller counterpart, more specifically, from the G-DLRM component. No RT is maintained at the OBS controller.

Scheduler: comprises a data channel scheduling algorithm to perform the reservation of resources (i.e. wavelength) to a burst duration. In the proposed node architecture, the scheduling algorithm is passed with a set of wavelengths given by the FT. Such set varies according to the incoming label in the BCP.

OBS Data link Resource Manager (O-DLRM): As mentioned above, no automatic optical resources discovery is implemented, so each OBS controller only has a representation of the local resources configured by the NMS. An entity

Table 2.1: Type and content description of the messages to be exchanged between the GMPLS and OBS controllers.

Directionality	**Message**	
	Type	**Content**
GMPLS to OBS	Configuration	- FT updates - Control channel updates
OBS to GMPLS	Status Report	- Outgoing link usage status - Physical failure report - e2e LSP status

called Agent(see Section 2.3) is implemented to send regular updates about such local resources to the GMPLS controller.

2.2.2 Relationships Between Building Blocks

Figure 2.5 describes the relationships among the different building blocks of the GMPLS controller and how they cooperate whenever a signalling message arrives.

The first case is when a b-LSP request arrives at the GMPLS controller belonging to the (G)OBS edge node (1). After it has been admitted by the CAC block, the b-LSP request is handled by the SC, which creates the signalling Path message to be sent towards the destination node. The SC then requests the routing for such b-LSP. The computation is realized at the PCE, a module part of the RC but that may be deployed either locally (at the GMPLS controller) or centralized (at a central node). This element returns a spatial route accomplishing the b-LSP requirements (e.g. load and QoS). In both cases, the computation is based on the information kept at the TED, which is updated periodically through routing messages. If a proper route is found, the resources for the next hop of such route are computed locally at the RC (i.e. wavelength assignment (WA)). Note that the network condition may slightly change even in such short period. In such a way, the RC creates a forwarding entry, which is kept till the confirmation of the b-LSP establishment is received, and forwards the Path message to the next node along the computed spatial route. Otherwise, it sends back a NACK message announcing that it is not possible to establish the b-LSP.

If the node is a transit node (i.e. neither the source nor the destination node), it is responsible to switch the Path or Resv message. In the former case (2), the SC requests to the RC to do the wavelength assignment of resources. In fact, the RC takes into account all the existent b-LSPs requests being controller by such node to do the WA. Again, it creates a temporary entry for such b-LSP. In the same way, it sends back a PathErr message if the assignment is not possible. If the node is the destination node, it creates a Resv message to be sent towards the source node.

In the later case (3), whenever a node receives a Resv message, it confirms the previous assigned resources, i.e., the temporary forwarding entry is passed to the current FT or it removes such temporary forwarding entry (in this case a PathErr is received). In a case that the FT is updated, a configuration message containing it is sent to the OBS controller by the G-DLRM. The flooding of routing messages containing the updated status of the next outgoing link is also initiated. If the node is the source, it sends a confirmation of the b-LSP establishment to the client side seizing the start of the data transmission.

Regarding the OBS controller, the relationships between its building blocks can be described through the BCP processing operation. In such a way, the BCP undergo by the following steps in a core node: firstly, the BCP is optically/electrically (OE) converted at the OBS controller and inserted into the input buffer of the CPU module (see Fig.2.4). At the CPU, the label is taken and read. The FT lookup is done, extracting the next hop and the set of candidate channels to be used by the traffic priority type of burst being expected (i.e. given by a QoS parameter). This set is sent to the local wavelength scheduler block together with the burst duration information extracted from the T_SPEC field of the BCP. Then, a local decision on which output wavelength to select is performed and the proper switch matrix configuration takes place. If the optional features are activated at the BCP, the node extracts also such information and sends it to the GMPLS controller counterpart. In the same way, it inserts new TE-link information about its outgoing links. After this, the BCP is ready to be sent to the next hop. The proper information is updated such as the offset time (if E-OBS architecture is assumed [Klinkowski09c]) and incoming wavelength and the BCP electrically/optically converted.

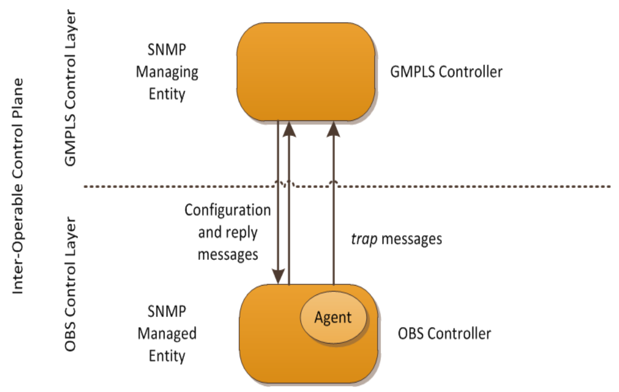


Figure 2.6: SNMP entities in the GMPLS-OBS network architecture.

2.3 Vertical Communication Channel

Owing to the constant exchange of information required between GMPLS and OBS controllers, a new "vertical" communication channel must be "created". Therefore, a proper communication protocol should be applied to formalize the operations over such channel, following the ASON recommendations. Such a protocol must define the format and content of the exchanged messages as well as describe how to handle them and where to keep the data. To the best of our knowledge, this is a complete new topic which has not been found in the literature. We first break-down what information should to be exchanged, as presented in Table 2.1. Then, we elaborate on the protocol fitting such channel requirements.

This channel is bi-directional as there is information traveling in both ways, i.e., from GMPLS to OBS and vice-versa. GMPLS sends configuration messages as it is the decision center, regarding: i) an updated FT every time a modification occurs at the GMPLS level; ii) control channel creation and maintenance updates (from the LMP-related information).

On the other hand, OBS provides GMPLS with periodic reports about the optical-network status. It should report the usage status of its outgoing link and of node itself (e.g. offered load) at every certain time - helps to detect traffic peaks at the GMPLS level -, physical failure report to trigger the protection and restoration mechanisms of GMPLS (e.g. LMP tool); and e2e LSP status information (by the edge nodes), which may help in traffic planning. Nonetheless, one must be careful with the message's overwhelming.

2.3.1 The Protocol: SNMP

The simple network management protocol (SNMP) is the mostly used and deployed network management framework [RFC3417]. It is an Internet standard protocol and is very mature (currently on version3). Note that it does not itself manage the network. It just provides a tool to something or someone to manage it and it perfectly suites the requirements of the vertical communication channel.

The protocol runs between a managing entity and the managed devices. Each managed device executes a software component called Agent, which has local knowledge of management information and reports it via SNMP to the manager entity. The manager entity could be composed by several nodes. In the context of our architecture, the managed entity is the OBS controller and the managing entity the GMPLS controller, as depicted in Fig. 2.6. The collected information is kept as variables in the MIB of the managing entity, which in turn are kept at the G-DLRM component of the GMPLS controller. The variables can be specified accordingly to the information desired to be captured.

The SNMP allows two different modes of operation. It can work on a request-response basis, in which the managing entity sends requests to the managed entity. The agent processes it, performs some action accordingly and reply back. Or, it can work on a notification basis, in which the managed device, through the agent, sends unsolicited messages known as trap messages to the managing entity. This is trigger upon an exceptional situation. Again, in the context of our architecture, we consider both ways. The trap messages are configured to be sent at regular periods containing optical network status information. The other mode is decided by the GMPLS controller whenever it needs (as mentioned earlier).

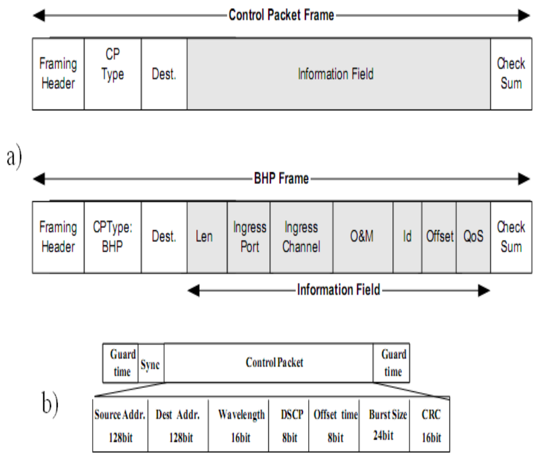


Figure 2.7: BCP formats found in the literature.

2.4 OBS Control Packet Format

The BCP is a control packet of the OBS network containing the proper routing information about the data burst to be optically switched at each node along its path, from the source to the destination node. These packets are sent prior to the data burst and processed electronically at each OBS controller to which they pass the necessary information to the resource configuration itself to be done (i.e. reservation). Up to date, such packet is missing a stable and standard framing format.

In [Farahmand07], the authors present a detailed approach of such problem (Fig.2.7a). Also K. Long suggests in [Long06] that the format of the BCP defined in [Long03] is considered by only replacing the source/destination IPv6 address fields by a label field with 20 bits length. In addition, it includes information such as DiffServ codepoints (DSCP), offset time and burst size in a total of 92 bits.

The BCP processing is a critical thing as congestion in the control-channel is highly undesirable. Consequently, the BCP should contain as limited information as possible to guarantee fast processing at each node as well as be scalable to higher speeds. Note, however, that errors and then losses of such packets will also have a huge impact on the overall burst transmission and then in the burst losses. Thus, the BCP should contain proper mechanisms to deal with errors too.

Departing from the aforementioned proposals, we come out with an even more generic format, which is lighter and more flexible to the control model adopted in the (G)OBS architecture. Both the generic and the detailed BCP formats are illustrated in Fig.2.8. Normally, such packets are of fixed-size but not in our case. For instance, the BCP size may vary from a minimum size of 72 bits up to a maximum of 96 bits plus the variable size of the optional field. The four main blocks of the proposed BCP format are described as follows:

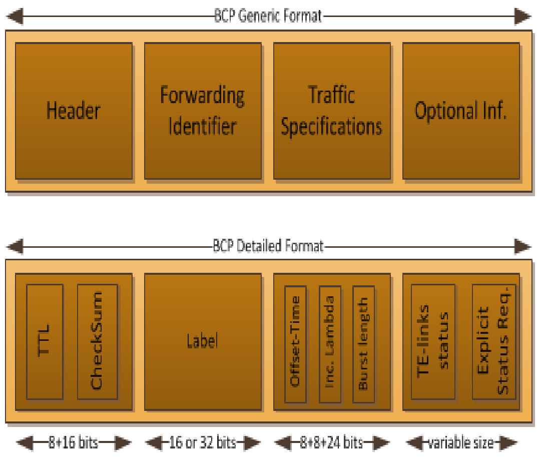


Figure 2.8: BCP format in the (G)OBS network architecture.

1) Header: contains framing sub-fields like the time-to-live (TTL) (8 bits) and the checksum (16 bits) to prevent any error - cyclic redundancy check (CRC) mechanism can be implemented -. Flags to signal the BCP beginning and end are used to determine the BCP length instead of the common type and version sub-fields.

2) Forwarding identifier: contains the label itself (16 or 32 bits) - format discussed in Chapter 4. A look-up at the FT determines the output resources to be configured.

3) Traffic Specifications (T_SPEC): contains all the traffic-specific parameters. This will be the offset-time (if E-OBS architecture is not assumed [Klinkowski09c]) (8 bits), the incoming wavelength (16 bits)- note that port information is not needed as the BCP and burst share the same path -, and the committed burst length duration (24 bits). Note once again that those sub-fields such as QoS (is given by label) and operation and maintenance (O&M) are also removed.

4) Optional information: may contain status information about the TE-links that it transverse (e.g. wavelengths being used by HP traffic per QoS, residual bandwidth) or even contain explicit requests of such information (variable size).

Chapter 3

(G)OBS Control Model

The model is formally defined as a systematic and logical description of complex system by means of a simplified abstract representation. In the present context, the control model comprises the whole set of operations to take place in the aforementioned architecture (functional mode), trying to accomplish both network (GMPLS independence and OBS statistical multiplexing) and traffic (QoS-client) requirements. The rationale here is to design an OBS control layer of low complexity, using simple encoding techniques and short frames lengths with minimum control overhead, while moving the more time consuming and not time-constraint tasks to the GMPLS control layer.

3.1 Overall Behavior

The proposed model comprises the whole set of control operations to be taken in the (G)OBS architecture presented in previous Chapter. The goal is to preserve GMPLS independence (i.e. the extensions to handle OBS will not affect its behavior to other switching technologies it controls) and OBS statistical multiplexing property (i.e. to achieve good network usage), at the same time it is to provide absolute QoS provisioning in OBS networks, which is crucial to Internet applications.

One of the milestones of this model is the co-existence of two different signalling time scales as the two-way GMPLS (referred to as GMPLS signalling time) and the one-way OBS signalling (referred to as OBS signalling time) paradigms are to be maintained. Hence, the control operations are assigned to each control layer on a per time-scale criterion, as explained in the following section.

In the context of our proposal, the LSP concept is to be maintained as to keep GMPLS principles intact. Its removal is therefore out of questions as suggested in [Manolova07]. It would mean to loose TE features from GMPLS or the need to operate several changes in its standard. It would also mean to loose the chance to announce OBS domain as a TE-link to another switching domains and to have e2e LSP crossing OBS (transit) domains. On the other hand, by extending it to be able to operate in OBS environment in a way that the set up of a burst-LSP or b-LSP is performed without any physical reservation binding, we are able to preserve the statistical multiplexing of resources given by OBS (allowing different traffic flows to share resources) and to add higher control in the traffic handling in OBS. The RSVP-TE protocol already covers such case as we will see further. In a nutshell, the proposed control model aims to:

- ✓ Guarantee end-to-end QoS of HP b-LSPs together with optimized resource usage.
- ✓ Allow better BE traffic routing and keep the network overall BLP as low as possible (due to the former HP optimized resource usage).
- ✓ Take advantage of all GMPLS TE features to cope with traffic dynamics, improve network planning and better handle failures through recovery/restoration and protection mechanisms.

There are some drawbacks too and they are related chiefly with the optical switch technology. How fast can the optical switches actually be? Is not a waste of energy if their lasers are due to be turn-on 24/7? These are relevant questions, albeit they are out of the scope of this thesis.

Table 3.1: (G)OBS Control Plane Modules: distribution of the control tasks.

Time Req.	Control Layer	Task Type	Signalling Procedures	Routing Procedures
Low	GMPLS	Background	b-LSP Management	Network Topology Updating
High	OBS	Specific	BCP Management	Resource Availability

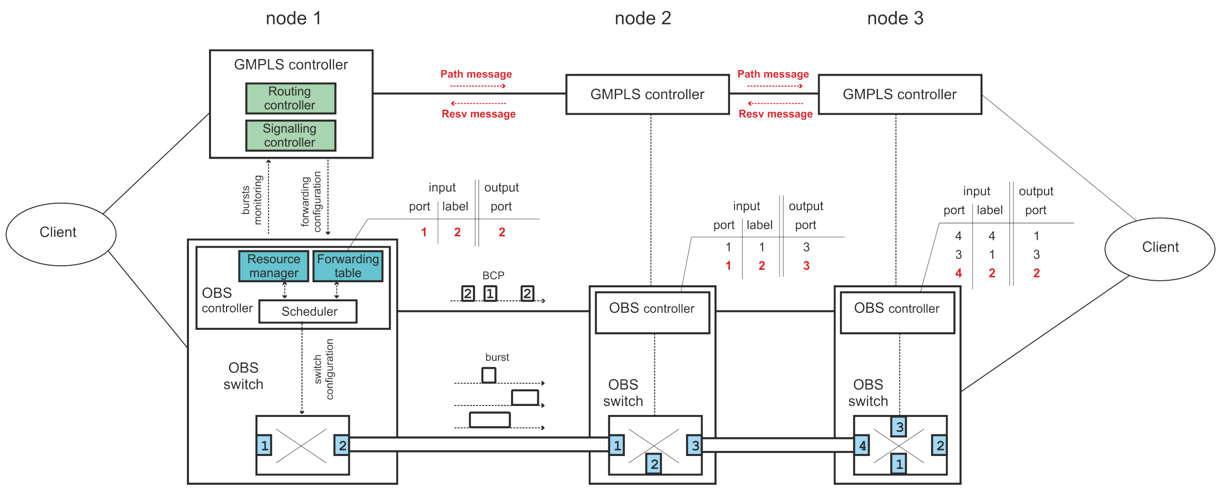


Figure 3.1: Example of a b-LSP establishment in the (G)OBS network architecture.

3.2 Control Operations Division

The control operations are assigned on a per time-scale basis as shown in Table 3.1. Note that not all operations have the same time requirements.

The set of **Background Control Tasks** (BCT) comprises those tasks with lower time requirements with a time scale variation in the order of minutes, hours, days or even longer. These are the responsibility of the GMPLS controller and are related to the management of the b-LSPs and the update of the network topology (addition/removal of a node/link; failures).

The **Specific Control Tasks** (SCT) have a much more restrict time scale, with variations in order of microseconds/milliseconds and are related to the actual resource reservation at the data plane as well as to a fast notification of the network resource status. These are the responsibility of the OBS controller.

These two sets comprise either signalling or routing procedures. Two of the four modules presented, are enhanced with new features to improve (G)OBS performance, in terms of burst loss probability (BLP) guarantees: the b-LSP management module (BCT-Signalling) and the network resource status dissemination module (SCT-Routing). The remaining two modules are left as though as they use the current protocol standards. The route computation is defined by a separated module as to be kept independent of the control model.

3.3 Signalling Procedures

The signalling of a connection defines the way resources are committed to satisfy a certain traffic demand. This commitment is performed at each node along a specific path. By default, both GMPLS - two-way oriented - and OBS - one-way oriented - control layers have their own signalling procedure, which diverge on how to commit the resources. The rationale behind the proposed control model is to maintain both although operating minor changes in order to avoid redundant and waste of committed resources.

The resource commitment (i.e. wavelength allocation or commitment here) performed by the GMPLS controller does not entail any physical resource reservation. This corresponds to the (logical) selection of a set of wavelengths that are accessible within a VT. In fact, the b-LSP is merely a connection representation at the GMPLS control layer, and it does not exist at the OBS control layer. This contrasts to the standard GMPLS signalling paradigm. By the standard, CP and DP commitment of resources means the same, i.e., a wavelength is exclusively and physically reserved for the duration of an LSP (DP level) and at the signalling time (CP level). Applying this to OBS would mean to lose its statistical multiplexing characteristic - which is not desired at any rate (see Section - (resource CP commitment)).

Therefore, the wavelength reservation is maintained at the OBS controller and it represents the actual commitment of a wavelength at the data plane (from the set of allocated wavelengths at the GMPLS signalling time) to transmit a burst (see Section - (resource DP commitment)).

3.3.1 b-LSP Management (BCT-signalling)

An example of b-LSP establishment is shown in Fig.3.1. A new connection is required between two clients attached to node 1 and 3, respectively. Note that the b-LSP signalling should be done per traffic flow and not per single burst, labeling each burst (i.e. BCP) accordingly. Before sending the Path message towards the destination, in order to signal the b-LSP, Node 1 must first define a b-LSP route. Such route should satisfy the traffic and QoS requirements of the connection request.

The path computation problematic is addressed in Section 3.5, where we provide several path computation strategies to achieve absolute QoS support to quality-demanding traffic together with an optimal network usage, under different traffic scenarios. In principle, any path routing can be adopted ranging from simple shortest-path to complex optimization algorithms. Moreover, it can be deployed in a distributed or centralized way (see Section 2.2 of Chapter 2).

The b-LSP signalling proceeds as follows. The signalling Path message follows the computed route and verifies its feasibility at every node along such route (i.e. the node locally determines the resources to commit based in traffic parameters carried in the Path message and according to the active route policy - see Sections 3.5; 3.7; 3.8).

If the Path messages arrives correctly to the destination node, the a signalling Resv message is then sent back. If the b-LSP is feasible, at each node of its back trip to the source node, the signalling Resv message triggers the update of the forwarding table and send it to OBS node controller counterpart. The forwarding table stores an input/output port match associated to the b-LSP (input port,label $\Rightarrow$ output port), that is afterwards looked up for data plane burst forwarding. Once the signalling procedure ends, the bursts can be sent to the destination node along the established b-LSP. The setup latency introduced is relatively low compared with the b-LSP holding time as shown in Chapter 5. Note that only a single GMPLS signalling session is required to transmit all bursts belonging to a b-LSP.

All messages related with this task are exchanged at the GMPLS layer, using an extended version of the GMPLS RSVP-TE signalling protocol (see Chapter 4). The connection time requirements are not as strict as the burst switching ones, which makes the b-LSP management a background task.

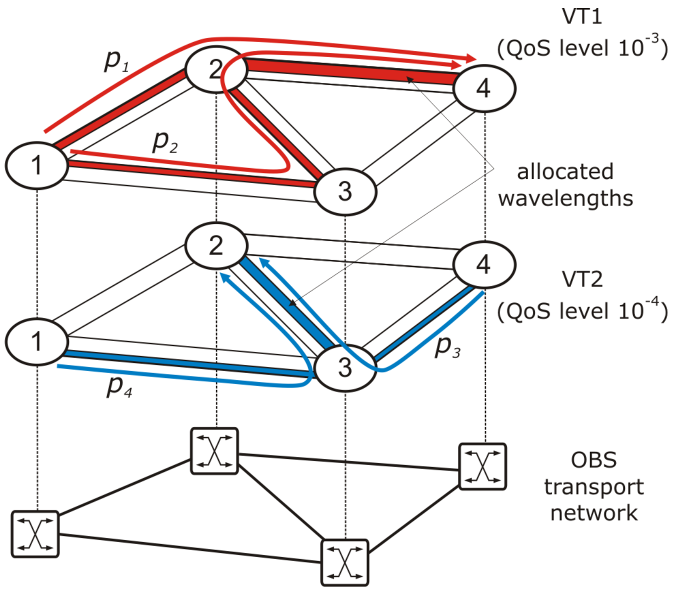


Figure 3.2: Virtual topology in (G)OBS.

Virtual Topology of b-LSPs

The set of several b-LSPs forms a Virtual Topology (VT). We consider that a (limited) number of VTs is maintained on top of the physical OBS transport network and each VT is dedicated to guarantee a given QoS (i.e., a given BLP level). Eventually, we consider the best effort (BE) class of traffic uses the spare network capacity. The set of wavelengths allocated in the links belonging to the VT are appropriately chosen so that to satisfy some (absolute) QoS. We assume that the wavelengths allocated to a VT are accessible for any burst that is carried within the VT, without respect to its origin and destination.

In this work, we assume the network applies the dedicated WA policy so that there is no sharing of wavelengths between VTs. This is motivated by a relatively simple burst loss model that can be derived for such policy which allows to estimate the number of wavelengths to be allocated in the function of offered traffic load and target BLP. Apart from that, we assume the network operates with unsplittable (non-bifurcated) routing. This single-path routing approach avoids the problem of the out-of-order burst arrival [Gunreben07]. Despite such choices, still the proposed framework allows to employ any other WA and routing approach as far as appropriate TE rules for the resource allocation within the VT design problem can be provided.

In Fig.3.2, we can see an example of the (G)OBS network with two VTs established, where two different levels of BLP are guaranteed, respectively, 10^{-3} and 10^{-4}. In this network, the burst contention will arise only within a VT and when two or more paths are routed over the same link. This can happen between paths $p_1 - p_2$ and $p_3 - p_4$ in the links connecting nodes 2-4 and 3-2, respectively. Accordingly, thanks to the dedicated WA for each VT (explained in Section 3.5), the traffic carried over a VT does not affect the traffic carried over other VT in network links.

During the network operation, changes in the VT may occur upon acceptance of new connections by the admission control mechanism. Those changes may concern the increase the capacity of certain b-LSPs (i.e. increase the number of allocated wavelengths in congested links), the change of the routing path, or even either partial or complete reconfiguration of the VT. Similar actions may be taken whenever a connection is terminated. An on-line resource provisioning mechanism which adapts the VT is deployed at the GMPLS control layer (Section 3.8).

Also, the network is equipped with a monitoring function at the network links so that to verify the actual BLP statistics. A proper GMPLS-driven b-LSP capacity reconfiguration mechanism triggers WA procedures whenever unexpected traffic peaks that affect the provided service (i.e., QoS) levels are detected (Section 3.9). In other words, the burst traffic profile is such that the target BLP is not met at a particular link and additional wavelengths are added to such congested links.

3.3.2 BCP Management (SCT-signalling)

The OBS node controller is the one that actually commits data plane resources for the incoming bursts. As in conventional OBS, once a burst is assembled and ready at the edge node, the corresponding BCP is firstly dispatched. Once the data burst is ready, the node releases corresponding BCP that contains the information (e.g. a label) identifying the b-LSP. After the BCP has been transmitted over the control wavelength and the offset time has expired, the data burst is released. Note that here the BCP and the burst are restricted to follow the b-LSP.

At each intermediate node, the BCP is electrically terminated and processed at the OBS controller. Based on the BCP label, it looks up the FT to determine the output port and the set of wavelength by which the burst can be transmitted. From such set, the controller chooses the output wavelength based on the local resource availability in accordance to the implemented active scheduling algorithm. The burst is then switched accordingly. This means the reservation is performed in a one-way mode, hop-by-hop, upon the BCP reception. The control operations at this node are really simple as to avoid switching complexity and perform fast forwarding.

In this context, the OBS signalling protocols are mature. Three well-studied protocols can be used without any modification such as JIT, JET and Horizon. Such committed resources are then released once the burst is completely transmitted (it depends whether JIT or JET is considered). Once again, the statistical multiplexing of OBS is preserved.

3.3.3 Example

In Fig. 3.3, an example of a (G)OBS network is illustrated with two virtual topologies enabling two QoS levels in terms of bursts losses at 10^{-4} and 10^{-6} ratio, respectively. In this example, three HP b-LSPs and two BE b-LSPs crossing node 2 are currently established in the network, so five entries are hence configured in the forwarding table. Note that only the HP b-LSPs are explicitly depicted in Fig. 3.3.

All b-LSPs in the forwarding table are identified by the input port and label. While the allocated wavelengths and the output port are present in the HP b-LSPs entries, only the output port is required to those b-LSPs transporting BE traffic. Indeed, BE bursts can use any of the available wavelengths at the indicated output port. However, in order to assure the HP objective QoS, they can be preempted (and thus dropped) by any HP burst that requires that particular wavelength allocated in its FT.

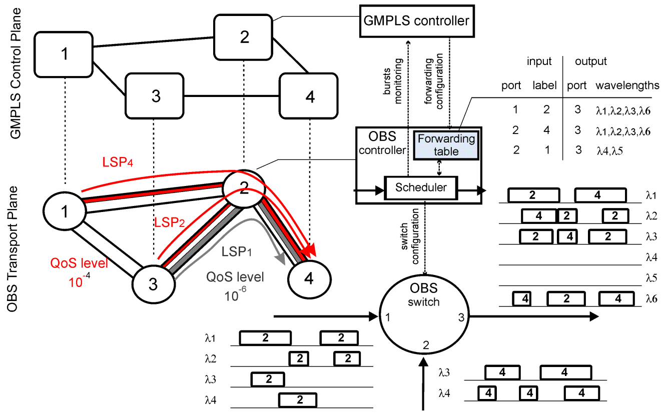


Figure 3.3: The VT and its effect on the burst forwarding.

Although all established b-LSPs use the same output port 3, only those HP b-LSPs with the same QoS share output wavelengths. In the example, bursts belonging to LSP_1 (with QoS at 10^{-6}) can leave node 2 only using either λ_4 or λ_5, while those belonging to LSP_4 and LSP_2 (both with QoS at 10^{-4}) can share λ_1, λ_2, and λ_6. At the top of the figure, an example of some burst arrivals at port 1 and 2 is also depicted: grey bursts for LSP_1, red bursts for LSP_2 and LSP_4, and white bursts for BE traffic labeled LSP_5 and LSP_7.

For bursts sharing the same group of output wavelengths, the OBS scheduler has to find and reserve the proper resources *on-the-fly* (i.e. locally). Although some bursts can still be lost with this approach, the amount of wavelengths assigned by GMPLS to every HP b-LSP ensures the required QoS levels.

3.4 Routing Procedures

The routing procedures have a directed impact on the whole network performance. The way network status information is passed across the network, determines the accuracy of the path routing decisions.

3.4.1 Network Topology (BCT-routing)

This module comprises network topology discovery and resource status dissemination. The GMPLS routing protocol is in charge of the dissemination of physical topology information. In fact, GMPLS routing is not all but distribution of information that will be used as the basis for path computation.

Any major change to the resource status, such as b-LSP set-up or tear-down, should be announced by the GMPLS routing protocol. Thus, some extensions to the GMPLS routing protocol are required in order to disseminate such OBS-related information (see Chapter 4). Those small changes, such as b-LSP capacity reconfiguration, are not disseminated using this mean as they are too dynamic (they are solved locally - see Section 3.9).

The use of GMPLS routing protocol is also important when deploying a PCE node, which "sniffs" these messages to acquire network status information and update its TE-database. The time requirements are low and no modifications are required.

The routing and wavelength assignment (RWA) computation feature, responsible to define the b-LSP's routes and capacities, optimizing the network resource usage, is not part of this module in order to keep it independent of the control model. This gives the proper flexibility to implement whichever RWA policy (see next Section 3.5).

3.4.2 Network Resource Status (SCT-routing)

This module is responsible to gather and disseminate the information of "current" state (i.e. availability) of the network resources at the optical level. A proper and timely performance of those tasks helps to balance the traffic and to reduce the burst-loss probability. GMPLS routing protocols may be too slow to cope with the highly dynamic traffic changes in OBS. However, this is a very hard task in distributed environments.

Following what is being done by the major telecom operators, we consider a centralized node with a PCE. In such a way, we provide more accuracy in b-LSP routing.

In addition, a specific OBS-oriented notification mechanism may be devised to exchange "current" network status information, for which we would suggest to use an extra object like the ADSpec object of the RSVP-TE Path message, in the BCP. Since the b-LSPs are commonly bidirectional, a BCP could collect the status of those nodes along the b-LSP and delivery it to the destination node. Such node is also the source node of a contrary b-LSP; thus it would be aware of its b-LSP e2e status. Although this does not solve the dissemination problem as it does not improve the dissemination speed, it avoids the deployment of novel messages and can reduce the number of GMPLS routing messages traveling around.

3.5 b-LSP: joint Routing and Wavelength Allocation (RWA) Policies

One of the main objective of the proposed (G)OBS control architecture and model is to provide absolute QoS guarantees for quality demanding flows of bursts transmitted between pairs of source-destination nodes in the network. We presume that such e2e QoS can be achieved by means of a TE approach, in particular, by i) an appropriate setup of routing paths (R), i.e. the route of b-LSPs, and ii) an adequate wavelength allocation (WA) on the links belonging to these b-LSPs. We assume two different traffic scenarios:

- ✓ i) static scenario: the matrix of traffic demands does not change along the time. An estimation of the traffic, which is given beforehand, allows to setup an optimized off-line VT dimensioning through a Mixed Linear Integer Programming (MILP) formulation.
- ✓ ii) dynamic scenario: the matrix of traffic demands changes along the time. An online VT maintenance mechanism is deployed, where the VT is augmented with and without changes of routes. Also, an adapted version of the MILP formulation of the off-line case is deployed on-line to optimize a set of connection requests arriving within a certain time window.

The provisioning of absolute QoS is very complicated in OBS networks due to the lack of viable optical buffering technologies. The solutions that allow to achieve absolute QoS are based mainly on the use of two-way signalling [Miguel04], burst preemption [Phuritatkul07], intentional burst dropping [Zhang04], and appropriate wavelength allocation [Yang07a][Yang07b] schemes. For a more thorough discussion on these solutions as well as for other references we refer to the survey of Klinkowski in [Klinkowski09a].

The WA approach to absolute QoS is very attractive since it can be implemented easily in a wavelength conversion-capable switching node. Indeed, the mechanism is based on a logical allocation of a subset of wavelengths, from the entire set of wavelengths in the network link, to be accessible for bursts belonging to a given QoS class. Upon the arrival of a burst, the wavelength reservation decision that is taken by the node controller concerns the selection of a wavelength from the set of allocated wavelengths. Several WA policies have been considered in the literature and they differ in the way the wavelength resources are partitioned [Dolzer04]. In details, wavelengths can be either shared between QoS classes or they are dedicated for each individual QoS class. Moreover, the allocation is either fixed and then it assigns particular wavelengths to a class or elastic and in such case it specifies only the maximal acceptable number of wavelengths that can be occupied by a class simultaneously. Although the WA mechanism is simple, still the key question is how many wavelengths should be allocated in network links so that to provide absolute QoS and, at the same time, use the wavelength resources efficiently.

For the shared-elastic WA policy, the problem of optimized WA was studied in [Yang07a] and [Yang07b]. The authors develop a link loss model that is used to determine the number of wavelengths required to satisfy certain strict BLP for a number of QoS classes. Since the optimization approach is very complex, due to the nonlinearity of both the objective function and model constraints, the authors propose a heuristic algorithm to provide a near-optimal solution to the WA problem. Regarding the dedicated WA policy, it involves a simpler loss model since a QoS class does not share the wavelengths with other classes. In this case, the Erlang B loss formula is frequently used to estimate burst losses in a network link [Yang07a][Dolzer04].

The network-wide QoS should be supported by network routing and a traffic engineering method. Indeed, a properly designed routing protocol may preserve from the selection of overloaded links when applying proper TE rules. In general, in OBS networks either reactive or proactive routing strategies have been considered [Klinkowski10]. In reactive routing, the routing decision is taken on-line, for instance, when burst contention occurs. Proactive routing strategies use either measurement-based or anticipated traffic demands to optimize, usually off-line, routing decisions. In the context of absolute QoS provisioning, proactive routing is a convenient approach since it allows to introduce TE rules easily and, by this means, control the distribution of traffic over the network [Yang07a].

The recent survey on routing methods [Klinkowski10] shows that the problem of routing with QoS guarantees has not been studied widely in OBS networks. The existing solutions belong mainly to the class of alternative (deflection) routing. These reactive-based strategies employ adaptive methods that introduce relative QoS guarantees by the differentiation of routing decisions with respect to the QoS class. Regarding absolute QoS, the common assumption in the literature is of the use of shortest path routing [Yang07a][Phuritatkul07] and network routing has not been explored to provide optimized solutions.

The concept of virtualization and, in particular, the joint problem of RWA with e2e QoS guarantees has not been considered in OBS to the best of our knowledge. Indeed, the common solutions for the e2e QoS provisioning are based on the shortest part routing assumption. Our framework for the absolute QoS provisioning takes several assumptions (as discussed in Section 3.6) which are used to develop a (mathematically) treatable network loss model.

In this case, the proposed optimization algorithms for off-line VT design and on-line VT maintenance are novel. Moreover, the burst loss models that are considered for TE differ from the models usually applied in IP/MPLS networks (as e.g., in [Gopalan03]) due to the buffer-less transmission in OBS. Finally, the proposed framework (deployed at the GMPLS control layer) facilitates the control and dynamic provisioning of resources for quality-demanding traffic in OBS.

3.6 VT Modelling

In this Section, we define a set of TE rules that are based on analytical modelling of the OBS network and that are used in optimizing and maintaining a VT. The network modeling concerns the definition of routing constraints, the estimation of burst losses, the strategy for burst loss guarantees, and the wavelength allocation function. These assumptions result in a set of constraints which are taken into account in the off-line and on-line algorithms presented in Sections 3.7 and 3.8. For sake of simplicity, in this work we focus on the design of a single VT, i.e., on the provisioning of absolute QoS guarantees with one level of BLP in the network only. Since our approach assumes dedicated WA in network links and, in particular, there is no sharing of wavelength resources between QoS classes, the formulation of the VT design problem can be extended straightforwardly to account for multiple VTs.

3.6.1 Notation

We use $G = (V, E)$ to denote the graph of an OBS network, where V and E denote, respectively, the set of nodes and the set of unidirectional links. Link $e \in E$ comprises W_e wavelengths. Let $W = \max\{W_e : e \in \mathcal{E}\}$.

We use the so-called path-link approach [Pioro04] for the network flow representation of the VT model. Let P denote the set of predefined candidate paths between source s and termination t nodes, where $s, t \in V$ and $s \neq t$. Each path $p \in P$ is identified with a subset of network links, i.e., $p \subseteq E$. Adequately, subset $P_e \subseteq P$ identifies all paths that go through link e. Let $\delta = \max\{\delta_p : p \in P\}$ be the length of the longest path in the network, where δ_p is the length (in hops) of path p.

Let D denote the set of demands with QoS guarantees, where each demand corresponds to a pair of source-termination nodes. Let $P_d \subseteq P$ denote the set of candidate paths supporting demand d; $P = \bigcup_{d \in D} P_d$. Each subset P_d comprises a (small) number of paths, e.g., k shortest paths, and a burst belonging to demand d can follow one of them. According to [Izal02], we assume that the traffic is characterized by a Poisson process. Let $h_d = \lambda_d / \mu$ denote the offered traffic load for demand $d \in D$, where λ_d is the arrival rate and μ^{-1} is the mean burst holding time; for convenience, the (constant) offered traffic of each demand is carried throughout a given path and therefore we consider $h_p = h_d$ for $p \in P_d$. Let $\rho_p \in \mathbb{R}_+$ and $\rho_e \in \mathbb{R}_+$ denote the offered load to path $p \in P$ and the offered load to link $e \in E$, respectively.

We maintain the following assignment of indices: $e = 1, \ldots, |\mathcal{E}|$, $p = 1, \ldots, |\mathcal{P}|$, $d = 1, \ldots, |\mathcal{D}|$, which identify, respectively, links, paths, and demands, and w is used to count wavelengths in link e.

3.6.2 Routing

We assume that the network applies source-based routing, i.e., the path to be followed by a burst is determined either by the source node or by the centralized node with a PCE. For all burst belonging to demand d the selection of path p from set P_d is performed according to decision variable x_p, also referred to as the routing variable. In this paper, we consider unsplittable (non-bifurcated/single-path) routing, which allows to avoid out-of-order burst arrivals. Therefore, the routing variables are binary variables and, consequently, a burst flow is routed over path p iff $x_p = 1$ and there is only one path $p \in P_d$ such that $x_p = 1$. These assumptions result in the following routing constraints:

$$\textstyle\sum_{p\in\mathcal{P}_d} x_p = 1, \forall d \in D, \text{ and } x_p \in \{0,1\}, \forall p \in P. \quad (3.1)$$

Note that multi-path routing can be modeled easily by assuming real-valued variables $x_p \in \langle 0,1\rangle, p \in P$. Finally, traffic ρ_p offered to path $p \in P_d$ is calculated as:

$$\rho_p = x_p.h_d \quad (3.2)$$

3.6.3 Burst Losses

To treat the QoS provisioning problem analytically, a burst loss model has to be developed so that to estimate the level of burst losses in network links. A common OBS network loss model is based on the reduced load approximation, which applies the Erlang fixed-point calculation [Rosberg03]. However, due to its computational complexity, we assume a simplified model based on the non-reduced load calculation [Klinkowski09b]. In this model, to estimate traffic load ρ_e offered to link e, we sum up the traffic load ρ_p offered to each path $p \in P$ that crosses this link:

$$\rho_e = \textstyle\sum_{p\in\mathcal{P}:p\ni e} \rho_p = \sum_{p\in\mathcal{P}:p\ni e} x_p h_p, \quad \forall e \in E. \quad (3.3)$$

The use of such approximation is justified by its accuracy, particularly under low overall burst losses (below 10^{-2}) [Klinkowski09b]. The burst loss probability L_p along path $p \in P$ can be calculated as

$$L_p = 1 - \prod_{e\in p} (1 - B_e), \quad (3.4)$$

where we account for blocking probabilities B_e in all links e that belong to path p. This approximation is based on general assumption that burst blocking events occur independently in network links.

The blocking probability Λ_d of a burst belonging to demand $d \in D$ can be calculated as a weighted sum of path loss probabilities. Hence, using (3.2) we have

$$\Lambda_d = \frac{\sum_{p\in\mathcal{P}_d} \rho_p L_p}{h_d} = \textstyle\sum_{p\in\mathcal{P}_d} x_p L_p. \quad (3.5)$$

The main difficulty of the above model is the calculation of losses B_e in network links, which depends highly on the burst traffic model. Under the common in the literature assumption of i.e.d burst arrivals, i.i.d burst durations, together with the assumption of the full wavelength conversion capability in network nodes and the dedicated WA policy, the Erlang B-loss formula can be used to estimate the probability a burst is lost in link $e \in E$:

$$B_e(\rho_e, c_e) = \frac{(\rho_e)^{c_e}/c_e!}{\sum_{k=1}^{c_e} (\rho_e)^k/k!}, \quad (3.6)$$

where B_e is a function of offered traffic load ρ_e and the number of provided (allocated) wavelengths c_e.

Remark: In the optimization problem in Sec. 3.7.2 we rely on numerical approximations of function B_e and its inverse. Therefore, any other dimensioning function that counts for different burst traffic characteristics can be represented straightforwardly in the formulation.

3.6.4 BLP guarantees

We assume that all the bursts routed within VT are delivered with certain absolute BLP guarantees. In particular, for each demand $d \in D$ the following constraints should hold:

$$\Lambda_d \le B^{e2e}, \quad \forall d \in D. \quad (3.7)$$

where B^{e2e} denotes the acceptable *e2e* BLP within the VT.

Constraints (3.7) may bring some difficulties when involved into the optimization problem due to the nonlinearity of Λ_d in the function of x (see (3.5)). In order to simplify the problem, an alternative solution is to replace (3.7) by a set of more restrictive, but treatable inequalities representing constraints on acceptable burst blocking probabilities in network links. A particular, yet convenient, case is when the BLP is kept below certain fixed level B^{link} at each link. In this paper we take such an approach and, similarly as in [Phuritatkul07], we consider B^{link} to be equal to:

$$B^{link} = 1 - \left(1 - B^{e2e}\right)^{1/\delta}. \tag{3.8}$$

It can be shown easily that the burst loss guarantees given by (3.7) are satisfied in OBS with unsplittable routing. Using (3.1)-(3.5), a proof consists in showing that if $B_e \leq B^{link}, \forall e \in E$, then for each $p \in P$ we have $L_p \leq B^{e2e}$ and, since for the active path q (i.e., such that $x_q = 1$) we have $\Lambda_d = \sum_{p \in \mathcal{P}_d} x_p L_p = L_q$, it results in $\Lambda_d \leq B^{e2e}$, what ends the proof. In the reminder of the paper, we assume B^{link} is a fixed value, the same for each link, and determined by the QoS objectives given by B^{e2e} and calculated according to (3.8).

3.6.5 Wavelength Allocation

The last modelling step concerns the definition of the dimensioning function $F(\cdot)$ that for given traffic load ρ_e determines the minimum number of wavelengths to be allocated in link e so that to meet given B^{link} requirements on the BLP. Such an estimation is given by a discontinuous, step-increasing function:

$$F(\rho_e) = \left\lceil \mathcal{B}^{-1}(\rho_e, B^{link}) \right\rceil, \tag{3.9}$$

where $B^{-1}(\rho_e, B^{link})$ is the inverse of the Erlang B Loss formula (3.6) extended to the real domain [Syski60], and $\lceil \cdot \rceil$ is the ceiling function; note that $B^{-1}(\cdot)$ is (strictly) concave. Because B^{link} is a predetermined parameter, for simplicity of presentation we skip it from the list of arguments of function $F(\cdot)$.

It is convenient to define a_w as the maximal load supported by w wavelengths given target blocking probability B^{link}, i.e., $a_w = B^{-1}(w, B^{link})$. Note that the inverse function $B^{-1}(w, B^{link})$ is expressed with respect to w and B^{link}, on the contrary to the inverse function used in (3.9).

Although there is no close formula to calculate the inverse of (3.6), still we can use a line search method (see e.g., [Minoux86]) to find the root ρ^* of function $f(\rho) = B^{link} - B(\rho, w)$ so that to approximate the value of a_w by $a_w = \rho^*$ for each index w, where $0 < w \leq W$; obviously, $a_0 = 0$. Vector $a = (a_0, ..., a_W)$ can be further used to determine $F(\rho_e)$ according to the following simple algorithm:

Algorithm 1 Wavelength Allocation (WA)

```
1: while $a_w < \rho_e$ & $w \leq W_e$ do $w++$
2: if $w \leq W_e$ then return $F \leftarrow w$ else return infeasibility
```

Algorithm 1 is a polynomial time algorithm of complexity $O(W)$. It is applied in the online resource provisioning in Section 3.8.

Eventually, the number of allocated wavelengths in link $e \in E$ must not exceed the total number of available wavelengths W_e. This capacity constraint results in the upper bound on the offered load $\rho_e^{\max} = a_{W_e}$ and can be represented as:

$$\rho_e \leq \rho_e^{\max}, \quad \forall e \in E. \tag{3.10}$$

3.7 OFF-line VT Maintenance

In this Section, we address the off-line VT design problem where a set of traffic demands is given. Such a problem concerns a variety of network scenarios, for instance, whenever the VT has to be rebuilt after a failure or an update in the network, or if there is some information available about admitted, estimated, or long-term traffic demands [Yang07a] [Phuritatkul07] that might be used to establish the VT, or if the VT is already operating in the network but its resource allocation needs to be re-optimized.

3.7.1 VT design optimization problem

In the off-line VT design problem we focus on the optimization of the resource (i.e., wavelength) allocation in the network. The motivation is that when minimizing the resources allocation for the VT, there is more resources left to be used for other classes of traffic. To this end, we define two wavelength usage functions, namely:

1. the overall wavelength usage in the network, given by $U_1(x) = \sum_{e \in \mathcal{E}} F(\rho_e(x))$, and
2. the wavelength usage in the most congested link, $U_2(x) = \max\{F(\rho_e(\boldsymbol{x})) : e \in \mathcal{E}\}$,

where $F(\cdot)$ is the wavelength allocation function defined by (3.9) and $\rho_e(x)$ is the link load function defined by (3.3).

Note that routing vector $x = (x_1, \ldots, x_{|\mathcal{P}|})$ determines the distribution of traffic over the network and thus the traffic load offered to network links. Consequently, in order to minimize the usage of wavelengths in network links, x should be optimized. Taking into account the modelling assumptions introduced in Section 3.6, the off-line VT design problem can be formulated as a non-convex optimization problem:

$$\begin{aligned} &\underset{\boldsymbol{x}}{\textbf{minimize}} && \Phi(\boldsymbol{x}) = f(U_1(\boldsymbol{x}), U_2(\boldsymbol{x})) && \text{(NLP)} \\ &\textbf{subject to} && \text{(3.1) and (3.10),} && \text{(3.11a)} \end{aligned}$$

where $\Phi(x)$ is a function of $U_1(x)$ and $U_2(x)$.

The difficulty of formulation NLP relies in the fact that there is no close formula to express $F(\cdot)$ since no such formula exists for the inverse of the Erlang function $\mathcal{B}^{-1}(\cdot)$. A way to solve the problem is to substitute function $F(\cdot), e \in \mathcal{E}$ with its piecewise linear approximation and reformulate problem NLP as a MILP problem.

3.7.2 MILP problem formulation

For a single link $e \in E$, whenever $\rho_e \leq \rho_e^{\max}$, the piecewise linear approximation of $F(\cdot)$ can be expressed as $F(\rho_e) = \min\{w : a_w \geq \rho_e\}$, or by means of a 0-1 integer linear programming (ILP) formulation:

$$\begin{aligned} &\underset{\boldsymbol{u}}{\textbf{minimize}} && F = \textstyle\sum_{w=1,\ldots,W_e} u_e^w && \text{(ILP1)} \\ &\textbf{subject to} && \textstyle\sum_{w=1,\ldots,W_e} u_e^w b_w \geq \rho_e, && \text{(3.12a)} \\ &&& u_e^w \geq u_e^{w+1},\ w = 1, \ldots, W_e - 1, && \text{(3.12b)} \\ &&& \boldsymbol{u} \in \{0,1\}^{W_e}, && \text{(3.12c)} \end{aligned}$$

where $\boldsymbol{u} = \left(u_{e1}, \ldots, u_e^{W_e}\right)$ are decision (binary) variables and $b_w = a_w - a_{w-1}$ for each $w = 1, \ldots, W_e$.

In ILP1, variable u_e^w is active whenever wavelength w in link e is allocated and, as a result, F is the sum of all active u_e^w. In constraint (3.12a), each active variable u_e^w increases load budget by b_w so that to achieve a value greater or equal to ρ_e. Thanks to ordering constraints (3.12b), first F^* variables u_e^w are active and, therefore, we have $\sum_{w=1,\ldots,W_e} u_e^w b_w = a_{F^*} - a_0 = a_{F^*} \geq \rho_e$, where F^* is the solution to ILP1.

Formulation ILP1 is analogous to formulation (4.3.25) in [Pioro04]. In [Pioro04], there is yet another formulation considered for concave dimensioning functions, namely (4.3.24), which may be used for modelling $F(\cdot)$. Although both formulations might not be computationally equivalent [Minoux01], still they provide the same linear programming relaxations and bounds when used with a MILP solver and the branch-and-bound method [Croxton02]. Indeed, our numerical experiments (not reported here) show there is no particular gain when using one formulation or the other. Hereafter, we make use (arbitrarily) of formulation ILP1.

Eventually, taking into account all network links and introducing routing variables, problem NLP can be reformulated as a MILP problem. Below, we present a MILP formulation with a multi-objective function, where the primary optimization objective is to minimize the overall wavelength usage and the secondary objective is to minimize the wavelength usage in the most congested link, i.e., $\Phi(x) = \alpha U_1(x) + U_2(x)$:

$$\begin{aligned} &\underset{\boldsymbol{x}}{\textbf{minimize}} && \Phi = \alpha \textstyle\sum_{e\in\mathcal{E}} F_e + G && \text{(MILP)} \\ &\textbf{subject to} && \textstyle\sum_{w=1,\ldots,W_e} u_e^w = F_e, \quad \forall e \in \mathcal{E}, && \text{(3.13a)} \\ &&& \textstyle\sum_{p\in\mathcal{P}_d} x_p = 1, \quad \forall d \in \mathcal{D}, && \text{(3.13b)} \\ &&& \textstyle\sum_{p\in\mathcal{P}:p\ni e} h_p x_p = \rho_e, \quad \forall e \in \mathcal{E}, && \text{(3.13c)} \\ &&& \rho_e \leq \rho_e^{\max}, \quad \forall e \in \mathcal{E}, && \text{(3.13d)} \\ &&& \textstyle\sum_{w=1,\ldots,W_e} u_e^w b_w \geq \rho_e, \quad \forall e \in \mathcal{E}, && \text{(3.13e)} \\ &&& u_e^w \geq u_e^{(w+1)}, \forall e \in \mathcal{E}, w = 1, \ldots, W_e - 1, && \text{(3.13f)} \\ &&& F_e \leq G, \quad \forall e \in \mathcal{E}, && \text{(3.13g)} \\ &&& \boldsymbol{u} \in \{0,1\}^{W_e}, F_e \in \mathbb{Z}_+, \quad \forall e \in \mathcal{E}, && \text{(3.13h)} \\ &&& \boldsymbol{x} \in \{0,1\}^{|\mathcal{P}|}, \boldsymbol{\rho} \in \mathbb{R}_+^{|\mathcal{E}|}, G \in \mathbb{Z}_+; && \text{(3.13i)} \end{aligned}$$

where variables F_e and G represent the wavelength usage, respectively, in link e and in the most congested link, $\rho = \left(\rho_1, ..., \rho_{|\mathcal{E}|}\right)$ are auxiliary variables representing the traffic load offered to links, and weighting factor $\alpha = W + 1$ is selected so that to give absolute priority to the overall wavelength usage objective. The introduction of the secondary objective will result in more balanced wavelength allocations.

Constraints (3.13a) count the allocated wavelengths in network links. (3.13b) are the routing constraints. (3.13c) are auxiliary constraints of the non-reduced load calculation. (3.13d) are the link capacity constraints. (3.13e) and (3.13f) result from the 0-1 IP representation of function $F(\cdot)$. In particular, the number of wavelengths allocated in link e should be such that the maximum traffic load it can support (calculated as the sum of active load segments b_w) is greater or equal to offered traffic load ρ_e. Besides, (3.13f) are ordering constraints, i.e., if w wavelengths are utilized so $w-1$ wavelengths are utilized as well. Constraints (3.13g) are used to obtain the wavelength allocation in the most congested link. Finally, (3.13h) and (3.13i) are the variable range constraints.

As discussed in Section 3.6.5, the wavelength allocation function $F(\cdot)$ comes from a concave dimensioning function. Therefore, by applying similar arguments as in Section 4.3.3 in [Pioro04], the optimal routing solutions of MILP will be non-bifurcated with highly unbalanced wavelength allocations in network links. Indeed, the marginal cost of allocating a new wavelength on the already occupied link is lower than on the empty link due to the character of $F(\cdot)$.

Note that MILP is a variant of the well-known discrete cost multicommodity flow (DCMCF) problem [Pioro04], which was shown to be very difficult [Minoux01].

3.8 ON-line VT Maintenance

In this Section, our focus is on the *e2e* QoS provisioning for quality-demanding burst traffic in a dynamic network scenario. Again, we assume that the QoS guarantees are achieved with the aim of a VT, which is maintained in the network. Without loss of generality, we consider both the cases where the VT is already established and operating in the network and the case where it is not and all the b-LSP requests (either set-up or tera-down) arrive after time t_0.

The connection requests of client networks of the (G)OBS network, such as IP networks, arrive through the aforementioned interfaces (see Chapter 2), notifying GMPLS control layer about the volume of quality-demanding data traffic to be transmitted (i.e. offered load) in the network. In order to meet the QoS objectives and satisfy the *e2e* BLP requirements for the traffic supported within the VT, admission control mechanisms are implemented in the network.

Such mechanisms should react both to the changes in the offered traffic load that are notified to the control plane - we will refer to it as the Flow Admission Control (FAC) mechanism - and during the burst assembly process performed at the edge node - referred to as the Admission Control (AC) mechanism.

FAC is responsible for admitting data flows from the clients under the condition there are enough wavelength resources available so that the traffic might be accommodated either within the current VT or after its modification. Concurrently, AC should take care of any excessive traffic which arrives at the OBS ingress node and which does not comply the FAC agreement, to be sent either through the OBS network as best effort traffic or dropped at the ingress node.

3.8.1 VT augmentation without route change

In the following discussion, let $\tilde{x} = (\tilde{x}_1, \ldots, \tilde{x}_{|\mathcal{P}|})$ be the routing vector which determines single routing paths that are used between source-termination nodes in the VT. Also, let $\tilde{h} = (\tilde{h}_1, \ldots, \tilde{h}_{|\mathcal{D}|})$ be the vector of the burst traffic load which is actually admitted to the VT. In particular, the routing vector is obtained either with the assistance of off-line optimization algorithms presented in Section 3.7 or by some other method (e.g., the shortest path algorithm). Eventually, let $\tilde{F} = (\tilde{F}_1, \ldots, \tilde{F}_{|\mathcal{E}|})$ be the wavelength allocation vector which represents the number of wavelengths allocated in network links within the VT. Here, we assume that $\tilde{F}$ is a function of vectors $\tilde{x}$ and $\tilde{h}$ (i.e., $\tilde{F} = f\left(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{h}}\right)$) and is determined using the model presented in Section 3.6.5.

The on-line VT maintenance mechanism, under a request of augmentation of offered burst load for demand d (i.e., between a pair of source-termination nodes) by volume h_d^+, the FAC mechanism performs the following steps:

1. **Let** $\hat{h} = (\tilde{h}_1, \ldots, \tilde{h}_d + h_d^+, \ldots, \tilde{h}_{|\mathcal{D}|})$ be an augmented traffic vector resulting from the actually admitted traffic and the new burst flow request. **Let** $\hat{F} = f\left(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{h}}\right)$ be a wavelength allocation vector required to support the augmented traffic;

2. **If** at least one element of vector $\hat{F}$ exceeds the link capacity (i.e., $\exists \hat{F}_e, e \in E : \hat{F}_e > W_e$) **then** reject the request

3. **Else if** $\hat{F} = \tilde{F}$ **then** accept the request ($\tilde{h} \leftarrow \hat{h}$) and maintain the VT without changes

4. **Else** accept the request ($\tilde{h} \leftarrow \hat{h}$) and increase the allocation of wavelengths in the VT whenever necessary (i.e., $\tilde{F} \leftarrow \hat{F}$).

The control plane should also act whenever it is notified about a diminishment of the burst flow load offered to demand d by volume h_d^-. In this case we consider the following mechanism:

1. **Let** $\check{h} = (\tilde{h}_1, \ldots, \tilde{h}_d - h_d^-, \ldots, \tilde{h}_{|\mathcal{D}|})$ be a diminished traffic vector after the reduction of the actually admitted traffic by h_d^-. **Let** $\check{F} = f\left(\tilde{\boldsymbol{x}}, \check{\boldsymbol{h}}\right)$ be a wavelength allocation vector required to support the diminished traffic;
2. Accept the request ($\tilde{h} \leftarrow \check{h}$);
3. **If** $\check{F} = \tilde{F}$ **then** maintain the VT without changes
4. **Else** reduce the allocation of wavelengths in the VT whenever necessary (i.e., $\tilde{F} \leftarrow \check{F}$).

In Chapter 5, we evaluate this VT maintenance mechanism in a simulation environment of a dynamic (G)OBS network. Apart from modifying the number of wavelengths, more advanced on-line mechanisms shall involve routing decisions and, for instance, the selection of alternative single paths. In this case, it will be probably advantageous, with respect to the wavelength usage, to rely the joint RWA decisions on a modified version of the optimization algorithms discussed in Section 3.7.

3.8.2 VT augmentation with route optimization

The VT augmentation with route optimization can be performed from either a distributed (at every source edge node) or a centralized (at the PCE node) manner. Running it at the source node s, means to perform path computation based on distributed network status information (normally passed by the routing protocol, e.g. OSPF-TE) and optimize the routing of those connection requests starting at the same node (i.e. source s) and terminating either at the same destination node or any other. On the other hand, running it from a centralized PCE node, means to compute requests from one or different source nodes towards the same or different destination nodes but based on a global network view. The frequency of incoming requests is higher at the PCE node than at any single source node.

This being said, let $\tilde{R} = (\tilde{R}_1, \ldots, \tilde{R}_{|\tilde{\mathcal{R}}|})$ be the vector that represents the different b-LSP requests arriving to the network during a certain time frame T, referred to as window. If $|\tilde{R}| = 1$ when T expires ($T = T_{end}$), a common constrained-SP routing can be applied whether centralized or distributed routing is considered. In order to optmized the resource usage, we suggest a k-SP heuristic aiming to minimize the increment of the number of wavelengths (I_w) due to a new b-LSP. From a set of possible paths, P, ($|P| = k$), between source node s and destination node, d, it selects the one that requires less wavelength to be assigned $p*(s,d)$, besides the ones already assigned to other b-LSPs of the same QoS category. This information is extracted from the TED of the GMPLS controller which keeps the number of wavelengths assigned per QoS level and per link, U_w, as well as the current load status of the links, ρ_l. The requested load, ρ_{req}, to be transmitted is extracted from the T_SPEC object of the Path message (as well as the QoS level). Check on Algorithm 2.

Algorithm 2 k-SP Heuristic.

```
Require: $k, W$
Ensure: $p*(s,d)$
 1: $I_{min} = \infty$
 2: $p_{selected} = \emptyset$
 3: $P \leftarrow$ compute K-SP(k)
 4: for $p \in P$ do
 5:    $I_w = 0$
 6:    $L_p \leftarrow$ get links of path p
 7:    for $l \in L_p$ do
 8:       $U_w, \rho_l \leftarrow TED$
 9:       $\rho_{req} = \rho_l + \rho_{offered}$
10:       $w = E_B^{-1}(\rho_{req}, QoS)$
11:       if $w < W$ and $w > 0$ then
12:          $I_w += w - U_w$
13:       else
14:          discard the path p
15:       end if
16:    end for
17:    if !discard then
18:       if $I_w < I_{min}$ then
19:          $I_{min} = I_w$
20:          $p*(s,d) = p$
21:       else
22:          if $I_w \equiv I_{min}$ then
23:             if $|p| < |p_{selected}|$ then
24:                $p*(s,d) = p$
25:             end if
26:          end if
27:       end if
28:    end if
29: end for
30: Return $p*(s,d)$
```

Otherwise, if $|\tilde{R}| > 1$, we achieve better performance if applying optimization techniques to process multiple b-LSP requests at a time. Hence, we suggest a modified version of the previous (off-line) MILP formulation to work in a on-line (i.e. dynamic) scenario. First, we fix the network state, i.e., previous established b-LSPs cannot be moved as it would lead to system instability. To this end, we introduce a new variable, C_e, to the constraint 3.13c:

$$\textstyle\sum_{p\in\mathcal{P}:p\ni e} h_p x_p = \rho_e + C_e, \quad \forall e \in \mathcal{E}, \tag{3.14}$$

Such variable C_e represents the current load crossing link e. This information is gather at the TE-database of the node. Second, the set P should now consist only on the set of candidate paths defined by the requests of the vector $\tilde{R}$. The procedure is described in Algorithm 3.

After every period the window time is renewed and $\tilde{R} = \emptyset$.

Algorithm 3 on-line b-LSP dimensioning using the modified MILP formulation.

```
Require: $T_{end}, |T|, k$
Ensure: $\Omega \neq null$
 1: $\tilde{R} = \emptyset$
 2: while $t < T_{end}$ do
 3:     $\tilde{R} \leftarrow \tilde{R} \cup \{\tilde{R}_t\}$
 4: end while
 5:
 6: if $|\tilde{R}| > 1$ then
 7:     for $\tilde{R}_t \in \tilde{R}$ do
 8:         $p = computeSpatialRoute(k, \tilde{R}_t)$
 9:         $P \leftarrow P \cup \{p\}$
10:     end for
11:     $\Omega = RunMILP(P)$
12: end if
13: update $T_{end} = T_{end} + |T|$
14: $\tilde{R} = \emptyset$
15: Return to Point 2.
```

3.9 GMPLS-driven b-LSP capacity reconfiguration

In highly dynamic networks, such as OBS networks, the volume of offered traffic may change abruptly and unexpectedly. Therefore, an auxiliary mechanism to locally handle short-duration peaks of traffic is deployed. This allows to maintain the QoS-levels not only statistically but at any situation during the entire lifetime of a b-LSP, as well as, avoid the entire VT reconfiguration.

The devised mechanism is triggered and operated from the GMPLS control layer. It acts in a proactive manner avoiding that the b-LSPs reach whether full capacity, which would increase the number of dropped bursts, or an inefficient use of wavelengths. The decision to increase/decrease the number of wavelengths associated to a b-LSP can be taken either on a local or end-to-end basis approach, spanning *n-hops*. Here, only locally based decisions spanning one single hop are considered, leaving end-to-end alternatives for further work. The dynamic reconfiguration of the b-LSP capacity works as follows.

Each OBS node $n \in \mathcal{V}$ is responsible for monitoring the HP traffic being offered by all b-LSPs supported on any of its output links i, $i = \{1, ..deg(n)\}$, over a sliding temporal window T with duration $|T|$. Note that by considering only one high-priority class, the b-LSPs share the same set of wavelengths at each output link, facing the same wavelength occupancy. Therefore, the offered high-priority traffic load to an output link i in the node can be expressed as:

$$\rho_{(i)} = \frac{\sum\limits_{b\in B} t_b}{|T|} \tag{3.15}$$

where t_b is the duration of the incoming burst $b \in B$, B denotes all the incoming HP bursts to be switched at node n within $|T|$. At every interval, the OBS controller sends a SNMP *trap* message to its respective GMPLS controller reporting the current HP traffic being offered to its output links.

Upon reception, the GMPLS controller is then responsible for detecting sudden traffic changes and to trigger b-LSP reconfiguration, if required. Given $\rho_{(i)}$, it verifies whether the b-LSPs' size at link i, L_i (i.e. the number of wavelengths), is still appropriated. To this end, it estimates the current HP traffic BLP using the Erlang-B loss formula ($\mathcal{B}$) and checks if the value remains below the demanded QoS threshold Ω. For this, the following assessment condition is verified:

$$eBLP_{HP} = \mathcal{B}(\rho_{(i)}, L_i) < \Omega \quad \forall_i \tag{3.16}$$

This mechanism does not trigger the reconfiguration request before W consecutive windows with $eBLP_{HP} \geq \Omega$ (per each output link i). Such decision helps to maintain the stability of the system by remaining insensitive to short-term traffic changes. If W is reached, however, the traffic peak is considered as significant and a 1-hop b-LSP expansion is triggered. To this goal, GMPLS computes the new set of wavelengths for the b-LSPs on output link i to properly face the measured HP traffic load. Such set of additional wavelengths is given by:

$$\phi(i) = \mathcal{B}^{-1}(\rho_{(i)}, \Omega) - L_i \tag{3.17}$$

where $\mathcal{B}^{-1}(\rho_{(i)}, \Omega)$ returns the number of wavelengths needed to satisfy Ω for the new estimated traffic.

On the other hand, GMPLS can verify that the capacity of the b-LSP is over-dimensioned (i.e. $\phi(i) < 0$). In such a case, a given number of wavelengths may be released. From the GMPLS control plane perspective, the rearrangement of those wavelengths assigned to the downstream link of a given b-LSP is quite straightforward. We point the reader to Section 4.3.5.

Chapter 4

(G)OBS Protocol Extensions

4.1 Impact on GMPLS standards

The GMPLS standard makes use of a single set of protocols to handle multiple switching domains [RFC3945]. It is in principle capable of controlling any technology due to its generalized nature, it is well studied and standardized. Yet, it does not cope with (optical) burst switching (i.e. OBS) domain [Qiao99]. From the few published works found in the literature related directly with the GMPLS and OBS integration, there are even fewer detailing on protocol extensions. In fact, there was one OBS-specific attempt (as an IETF draft) only to formalize such protocol extensions [LongDraft05], albeit no specific extensions were proposed.

In this Chapter, we aim to a well-defined GMPLS protocol framework with the necessary extensions to support the proposed (G)OBS control architecture and model defined in previous Chapters. To this end, we study the impact of burst switching type on the GMPLS protocol standards and consequently elaborate on the necessary protocol extensions. For instance, we cover GMPLS signalling (i.e. RSVP-TE) and routing (i.e. OSPF-TE) protocols. Protection and restoration mechanisms and the link management protocol (i.e. LMP) are out of the scope of this work.

Note that every extension proposal is defined in accordance to ITU-t (i.e. ASON) and IETF (i.e. GMPLS) requirements. The goal is to maintain GMPLS technology-independent as much as possible, i.e., the interoperability and further extensions must not compromise the overall GMPLS applicability to other switching technologies.

The remainder of this Chapter continues as follows. Section 4.2 introduces the switching and forwarding hierarchies currently defined in the GMPLS standards, and points out the possible changes on the LSP hierarchy to cope with the (G)OBS architecture. An OBS-specific switching interface is proposed, as well as, a new LSP region. Section 4.3 and 4.4 describe the signalling and routing aspects, respectively, and the several protocol extension alternatives.

4.2 Switching and Forwarding Hierarchies

The introduction of burst switching in GMPLS impacts on its standards and chiefly on its switching and forwarding hierarchies. Its correct integration might dictate the future of OBS in multi-region/multi-layer networks controlled by GMPLS [RFC5212]. However, where to insert it can be a tricky task and is dependent of the architecture and model assumed. The following scenarios may be formulated:

- ✓ Sub-division of the lambda-switching capable interface (LSC) according to granularity levels (i.e. circuit, burst and packet);
- ✓ "Encapsulation" of the burst switching capability into the LSC interface;
- ✓ Re-definition of the waveband-ISC;
- ✓ Definition of a novel switching type interface to OBS from scratch.

Nevertheless, there are countless evidences, as enumerated in Chapter 1, "demanding" the definition of a specific interface to accommodate burst switching (under OBS) in the GMPLS context. In fact, ignoring them, may lead to protocol incongruence. GMPLS must be aware of the presence of the OBS domain in order that the MRN/MLN concept [RFC5212] could be accomplished. This new interface must detail the proper procedures to handle an LSP in such domain like its signalling and forwarding aspects. The primordial steps in this area were done by K. Long through the submission of an IETF draft in 2005 [LongDraft05], yet it lacked on important details and specifications and thus failed to become an RFC.

Our strategy consists in:

1) to define the meaning of burst switching in the GMPLS context.

2) to elaborate on the possible alternatives to accommodate the burst switching concept in the switching and forwarding hierarchy of GMPLS and therefore propose a novel burst switching interface and burst LSP entity.

3) to propose protocol extensions to GMPLS handle OBS network.

4.2.1 Burst switching meaning in GMPLS

Even though OBS is a quite well-defined architecture, its meaning in the GMPLS context is still not defined. In the context of our proposed (G)OBS architecture, we define it as follows:

The burst switching type is the capability of a network node (i.e. its controller unit) to extract and process routing control information from an incoming (burst) control packet (i.e. BCP). The BCP is sent in advanced from the data payload itself for dynamic provision of sub-wavelength granularity, being electronic processed while the (data) burst travels all-optically. The following information must be read from the BCP fields:

- ✓ i) burst duration/length
- ✓ ii) incoming data wavelength
- ✓ iii) type switching to be performed (in hybrid scenarios).

This information allows the OBS controller to configure the switch matrix only for the burst duration. It also requires fast switching and control technology. Buffering is not required at any rate. Besides the buffer-less nature, another distinguishing characteristics of OBS is its statistical multiplexing, improving the network resources usage.

At GMPLS level, it means that the switching is not performed anymore at the lambda level and the switch unit is now a burst and not an entire wavelength. The switching is performed based on the incoming port/incoming wavelength association (if LSC interface), where a label identifies a specific wavelength. On the contrary, several bursts from different flows may travel over the same wavelength in a burst switching context, and therefore the sharing of the same wavelength (sub-wavelength granularity) by different LSPs/flows must be considered.

In such a way, a label must be associated to the entire LSP, which may consist in more than one wavelength per hop/link, and not to a specific wavelength. In addition, the wavelengths should not be reserved to preserve the statistical multiplexing of OBS, being only committed from a control plane level and not at the data plane - burst switching type. The effective reservation of resource is done after the LSP signalling and it is performed by the OBS control layer (upon the BCP reception). In contrast, if the type is circuit switching, the wavelengths are reserved immediately at the control plane level and during the signalling of the LSP (at GMPLS signalling time).

Thus, a proper switching interface may be advised to cope with OBS technology under a single GMPLS control instance. This will require an extension of the switching and forwarding hierarchy as well as of the LSP region of the GMPLS standard.

4.2.2 New Switching Capability

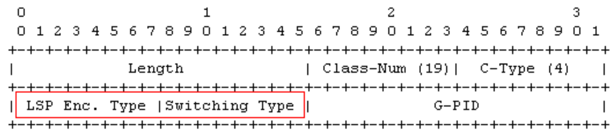
Motivations

The Interface Switching Capability (ISC) is introduced in GMPLS to support various types of switching technology (an ISC is identified via a switching type). Currently, GMPLS supports switching at the packet (PSC), frame (L2SC), time-slot (TDM), frequency/wavelength (LSC), and fiber (FSC) granularities as defined in [RFC3945] and [RFC3471]. A novel switching type, Data Channel Switching Capable (DCSC), was recently introduced in [RFC6002]. Parallel definitions for these switching types are also found in [RFC4202],[RFC4203] and [RFC5307].

In the previous subsection, we define the meaning of burst switching and highlight its differences to wavelength switching in GMPLS. Standard GMPLS does not handle OBS technology and therefore a proper OBS-related switching type should be introduced in GMPLS according to the former.

The definition of this novel switching type capability will guarantee the correct behavior of the whole hierarchy of GMPLS procedures. Below, we list some of the reasons that justify it:

1. MRN/MRLN the multi-domain control requirements, specified in [RFC5212][RFC6001], will be accomplished: end-to-end path computation and forwarding adjacency (FA) LSP connectivity among different regions and layers will be possible even when OBS domain is present. To this end, a new LSP region [RFC4206], and a new LSP type, must be created (see subsection 4.2.3).



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Length              | Class-Num (19)|  C-Type (4)   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| LSP Enc. Type |Switching Type |            G-PID              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.1: Generalized Label Request object (in Path message).

2. Signalling Protocol two parameters in the Generalized Label Request object (see Fig.4.1) of GMPLS RSVP-TE signalling PATH message are crucial to support the LSP being requested. One of them is the *encoding type* parameter, which indicates the encoding of the LSP being requested. It represents the nature of the LSP (i.e. burst), and not the nature of the links the LSP transverse (e.g. (D)WDM). The other one is the *Switching type* parameter, which indicates the type of switching that should be performed on a particular link. It is needed for links advertising more than one type of switching capability (e.g. MRN/MLN scenarios). A new value must be defined to advertise this new switching type for the corresponding link in the routing Switching Capability Descriptor (see point 3).

See [RFC3471] for a description of parameters and [RFC3473] for a description of procedures. As defined in [RFC3473], if the type indicated in the Switching Type parameter cannot be supported, on the corresponding incoming interface, the node must generate a PathErr message with a "Routing problem/Switching Type" indication.

3. Routing protocol another important requirement comes from the GMPLS routing protocol [RFC4202]. The link state advertisements (LSAs) must carry an ISC descriptor in accordance to the switching type supported by the TE-link. This is required to have a correct and feasible path computation. In addition, this also implies on the TE-link announcement of the burst LSP to lower regions (see subsection 4.2.3).

So, it is clear that either due to LSP encoding type, switching type or LSAs content, they all must be in agreement with the node/link interface type. Although the differences between burst and wavelength switching, the adaptation of GMPLS Lambda-ISC to cope with burst switching was formalized as an initial hypothesis. In the following, we discuss the various scenarios that were under study and point out the reasons why they are not the best solution and in some cases not doable.

A. Lambda-switching capable (LSC) Interface

The rationale behind this scenario is to maintain the LSC interface as the switching domain at control plane while extending data plane granularity beyond circuit switching (OCS), namely to burst (OBS) and packet switching (OPS) layers. This would require several changes, specifically at the LSC label meaning. The [RFC3945] defines the following: "LSC are interfaces that switch data based on the wavelength on which the data is received (...)". On the contrary, a single wavelength in OBS or OPS might carry traffic from different flows (sub-wavelength granularity), which means it is impossible to switch OBS or OPS-traffic based only on the incoming wavelength (the label itself).

In the work published in [Pedroso08], we suggested the "encapsulation" of burst switching into this interface. This approach is effortless and requires few protocol extensions. In fact, it is more an optimization extension (reduction in the number of FT entries) than actually a crucial extension contributing for the overall interoperability process. For such purpose, we made use of the end-to-end TE-Tunnel connection object. The Te-Tunnel would be a set of LSC-LSPs. Therefore, the *TE-Tunnel ID* would be used as the label to be included in the BCP in order to manage a set of wavelengths under the same connection instance (i.e. for the same e2e traffic flow).

However, the impact of these modifications at ISC level are significant. The switching type announced would be LSC while the LSP encoding type would be OBS. Even considering independent e2e LSC LSP establishment, the problem of the label meaning remains. If a GMPLS controller is not configured with the proper switching interface, the GMPLS signalling and routing would not work.

The mentioned label meaning mismatch between burst and wavelength switching also excludes the waveband hypothesis [RFC3945]. Waveband is a special case of wavelength switching. It represents a set of contiguous wavelengths, being switched as a unit block, w_1, where $w_1 < W$ and W the maximum number of wavelengths in a link. This hypothesis would also be rejected because our (G)OBS node switches a burst in one wavelength at a time and not as a block.

B. Burst Switching Capable (BSC) Interface

According to [RFC3945] and [RFC4202], and take into an account the previous enumerated advantages, the most correct way to integrate OBS in GMPLS is due to a new switching interface. An ISC is identified via a switching type. The new kind of switching interface will support burst-LSPs. The interfaces sending, receiving and treating GMPLS signalling messages with burst switching encoding type will be called Burst Switch Capable (BSC) interfaces.

Definition of BSC These interfaces at the GMPLS controller recognize the GMPLS signalling message boundaries, which carry the proper traffic information, and perform the assignment of resources at the control plane without physical reserve them. At each GMPLS controller a new forwarding entry is created and inserted at the forwarding table upon the reception of a Resv message. This entry, which is passed to the OBS controller counterpart, will allow it to then forward (all-optically) the optical signal (data burst) from an incoming wavelength to an outgoing wavelength based on the content of OBS signalling message, i.e., BCP (label-based forwarding).

Compatibility Transit and egress nodes that do not support the BSC Switching Type, when receiving with a Path message with a Label Request containing the BSC Switching Type, will behave in the same way nodes generally handle the case of an unsupported Switching Type.Such nodes are required to generate a PathErr message, with a "Routing problem/Unspported Encoding" in accordance to [RFC3473]. The ingress nodes initiating a Path message (containing such label request), will notify the requesting application user as appropriate if they receive such PathErr message.

Table 4.1: Altered GMPLS ISC Stack

Value	Type	
1	Packet-Switch Capable-1	(PSC-1)
2	Packet-Switch Capable-2	(PSC-2)
3	Packet-Switch Capable-3	(PSC-3)
4	Packet-Switch Capable-4	(PSC-4)
51	Layer-2 Switch Capable	(L2SC)
100	Time-Division-Multiplex Capable	(TDM)
???	**Burst-Switch Capable**	**(BSC)**
150	Lambda-Switch Capable	(LSC)
200	Fiber-Switch Capable	(FSC)

4.2.3 GMPLS LSP Hierarchy: new Region

The basic service abstraction in GMPLS is an LSP. The LSP is an end-to-end network connection between two interfaces of the same type. The [RFC4206] defines the concept of LSP region as a network switching domain where all interfaces are of the same type. The creation/construction of this region and its boundaries are defined by the information carried in the ISC descriptor.

As discussed before in subsection 4.2.2, a new LSP region must be considered in the GMPLS LSP hierarchy: BSC LSP region. This allows the correct establishment of an LSP crossing different LSP regions in an heterogenous network, where OBS is one of the switching domain. This new LSP can now be announced as a TE-link i.e. a forwarding adjacency (FA) LSP would be established to lower order LSP regions (e.g. TDM, L2SC, PSC), as illustrated in Fig.4.2. This is also important for the LSP route computation because it will take region boundaries into account (more details in [RFC3945]). This will also have implication on every sort of GMPLS procedures: bundling of FA-LSPs, signalling, routing, protection and recovery. Note, however, that this is only applicable when consider the same instance of GMPLS control plane. The definition of an LSP in the BSC context is described below.

Regarding the LSP/ISC hierarchical structure, the novel BSC

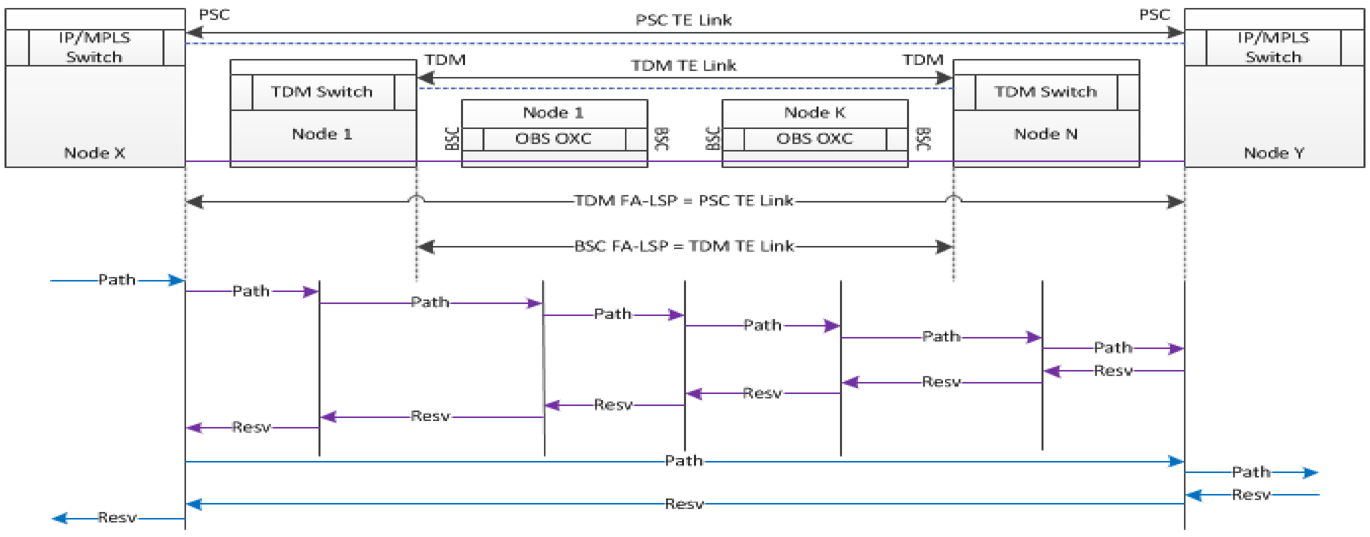


Figure 4.2: Example of a FA-LSP signalling. For instance, the BSC FA-LSP (or b-LSP) can be announced as a TE-link to lower TDM region.

interface places itself between TDM and LSC interfaces, as shown in Table 4.1. This means PSC, L2SC and TDM LSPs can be nested into one BSC and a BSC LSP can be nested into LSC LSP. The reverse is not possible, i.e., to nest a LSC LSP into a BSC LSP. The ISC value must be assigned by IANA and it should range between 125 and 150 to preserve the switching capability ordering and LSP region definitions specified in [RFC4206].

What is a b-LSP?

The burst-LSP is a label-switched path that may bundle one or more wavelengths per hop, unlike an LSC-LSP. Henceforth, we will use b-LSP to refer to a burst LSP, however, burst-LSP and BSC-LSP are equally correct. In the (G)OBS architecture, the b-LSP is merely a connection representation at the control plane only and does not entail a reservation of data plane resources (e.g., wavelengths in the OBS data layer). In fact, a b-LSP, established by the GMPLS signalling protocol, determines the end-to-end path that must be followed by all bursts belonging to it and the set of wavelengths that may be used at each hop. In this way, multiple b-LSPs can share the same network resources (i.e. wavelengths), preserving the statistical multiplexing capabilities of OBS. Note that only a single GMPLS signalling session is required to transmit all bursts of a b-LSP, i.e., it is per-demand and not per-burst basis. Hence, each GMPLS controller of each of the network nodes along such path must be configured with BSC interfaces in order to be able to perform burst switching configuration.

The bursts belonging to a b-LSP are forwarded all-optically using one of the wavelengths at a time per each link crossed, from the set stored at the FT and passed to the scheduler according to the label contained in the BCP. So, the outgoing wavelength is locally reserved at the burst time scale (i.e. upon the reception of the BCP at the OBS controller), but previously committed at the control plane at the b-LSP time scale (i.e. during b-LSP signalling at the GMPLS controller). It is worth to mention that is the OBS controller the one that commits data plane resources for the incoming bursts and not the GMPLS controller counterpart. Note also that the main difference here lies on the fact that BCP and burst are restricted to follow this same b-LSP. A possible BCP format is described in Section 2.4 of Chapter 2.

Label Meaning The b-LSP label has end-to-end meaning, and so no label swapping operation is performed. This label represents an input/output port match associated to the b-LSP (input port,label $\Rightarrow$ output port). It is looked up afterwards for data plane burst forwarding, given the set of wavelengths that can be selected (one per incoming burst).

The b-LSP label may be assigned either by the source or the destination node. If source-defined, we propose that it should be carried in the $<< suggested\_label >>$ object of the GMPLS signalling RSVP-TE Path message during the b-LSP signaling at the expense of some changes in such object. The drawback of such option is that by the standards, the downstream node can choose a complete different label which would oblige us to force the downstream to select the given suggested label. On the other hand, if destination-defined, no changes are required as the GENERALIZED_LABEL object of the Resv message can be used (see Fig.4.3).

Label Format The format of the label is as simple as an integer value. It can be a 16-bit or 32-bit label (see BCP format in Section 2.4 of Chapter 2), varying according to the size of the network. In order to guarantee the unique identification of the b-LSP inside the network, we propose the following data encoded structure: $< bits_x, bits_y >$, where the **x** most significant bits on the left represent the node id and the **y** most significant bits on the right, the number of b-LSPs accepted by such node (which is kept locally).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Label                             |
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.3: Generalized Label object (in Resv message).

4.3 Signalling extensions

As has been repeatedly said along Part I, the GMPLS-OBS signalling is the most sensitive aspect in the whole interoperability process. Besides the previous modifications in the switching and forwarding hierarchies of GMPLS standard, namely the novel LSP meaning and LSP region, specific protocol extensions are also required. The GMPLS signalling [RFC3471] makes use of two types of signalling protocols. In the context of our work, we refer only to the commonly deployed RSVP-TE protocol [RFC3473].

4.3.1 ASON Call and Connection

In accordance to the ASON definition of a control plane: "the control plane performs the call control and connection control functions through signalling , the CP sets up and releases connections and may restore a connection i case of failure (...)"; the GMPLS signalling Path message considered in our model also assumes the GMPLS extensions to support call and connection control [RFC4974]. We reuse the terms call and connection as follows:

i) a Call as an association between endpoints and possibly between key transit points (such as network boundaries) in support of an instance of a OBS service, building a relationship by which subsequent connections may be made; the Call (call_ID) is the logic association, an agreement between endpoints (source, destination), used to facilitate and manage a set of b-LSPs.

ii) a Connection as a b-LSP. A b-LSP may exist without a Call. The term TE-tunnel is used here as a set of multiple b-LSPs (different from what is proposed in [Pedroso08]). The goal of this TE-Tunnel object is to increase the scalability of the proposed architecture by reducing the number of entries at the forwarding tables.

4.3.2 b-LSP Set-Up

The GMPLS signalling of a b-LSP is triggered upon the reception of a client request (arriving either through UNI, E-NNI or NMS interface) demanding the transport of a certain data amount to a certain destination. The b-LSP signalling is done per traffic flow, i.e., same source-destination pair, and same QoS requirements. The definition of this process follows the [RFC3945], [RFC3471] and [RFC3473] standards.

Remember that the b-LSP signalling without actually committing resources in the data plane (i.e., cross-connection is not performed at this phase) is already contemplated in the RSVP-TE framework for GMPLS multi-layer/multi-region networks, as detailed in [RFC6001].

The signalling process is kept in two-way mode in order to preserve the GMPLS interoperability with other switching domains: from the source towards destination node (downstream) and from destination towards the source node (upstream). During the downstream, a temporary forwarding entry (FE) is kept at each GMPLS controller along the path, without interfering with the normal FT. If more than one b-LSP signalling is in place and some resources are to be shared among them, then those temporary FEs are updated accordingly. The effective FE is then created upstream, after the correct reception of the Resv message. Such FE is passed by SNMP messages to the OBS controller counterpart. Once the signalling is completed, the BCPs of such b-LSP can be labeled at the OBS control level.

Going DOWNstream (source ⇒ destination)

The b-LSP signalling starts by sending a PATH/Label request message towards the destination node. Among the objects contained in this message, we highlight the following ones:

i) GENERALIZED_LABEL_REQUEST: three major parameters must be specified to support the LSP being requested, namely encoding and switching type of an LSP and its payload type called Generalized PID (GPID). Only one label request object is carried per message, so a single LSP can be requested at a time per signalling message. It is technology generic but it requires the proper LSP switching (e.g. burst) and encoding type (e.g. OBS), which is now possible due to the new BSC interface.

ii) EXPLICIT_ROUTE (ERO): source-routing is considered as part of the (G)OBS model. It defines the nodes along the end-to-end path (it can be loosely or strictly define). This path is given by a RWA algorithm either computed in a centralized or distributed manner using the PCE. Note that all nodes along such path must satisfy the traffic requirements and support the requested switching interface.

iii) RSVP-TE SENDER_TSPEC/FLOW_SPEC: all the specific parameters other the ones specified in the GENERALIZED_LABEL_REQUEST are transported as technology-specific traffic parameters. In fact, in [RFC3945] is said that it is expected then specific parameters to be defined in the future for photonic (all optical) switching - circuit, burst and packet switching. The expired IETF draft from K. Long also mention it, suggesting the addition of new Type-Length-Value (TLV) objects (Fig.4.4), albeit without detailing them. The FLOW_SPEC object received in a Resv message should be identical to the content of the SENDER_TSPEC of the corresponding Path message. In other words, the receiver is normally not allowed to change the values of the traffic parameters. These traffic parameters are defined further (subsection 4.3.3).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Type              |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                             Value                             ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.4: Type-Length-Value (TLV) format.

Going UPstream (destination ⇒ source)

As previously defined in subsection 4.2.3, the format of the GENERALIZED_LABEL object traveling in RESV message is as simple as an integer value. If it is correct, it means the resource commitment can be accomplished. The confirmation of the b-LSP establishment is done upon the reception of this message at each GMPLS controller. This means that the temporary FE kept at the controller can be added to the normal FT of the GMPLS controller. A SNMP configuration message is sent to the OBS counterpart controller, which updates its own FT. The protocol proposed for the vertical communication is detailed in Section 2.3 of Chapter 2.

If the signalling fails, a PathErr message is sent, according to [RFC3473]. As soon as an upstream node receives this message, it removes the temporary FE (created during the downstream direction). Note that if there is more than one entry sharing the same resources, such remaining entries must be updated accordingly.

4.3.3 The New Traffic Parameters

According to the proposed control model (see Chapter 3), each downstream node along the b-LSP route must be informed of the set of resources to commit to such b-LSP. A logical way is to pass the traffic requirements and let each node to locally compute such set according to the implemented RWA policy. Thus, OBS-specific traffic parameters must be defined in order to include the requested bandwidth (load) and other traffic specifications, such as the QoS level. Note that there is no need to define discrete bandwidth values to a b-LSP as for a TDM or LSC LSPs.

To this end, we suggest to extend the SENDER_TSPEC object as to advertise such traffic parameters. The SENDER_TSPEC object (Class-Num = 12, Class-Type = 6) should then include the QoS level and the maximum transfer unit (MTU) values for the requested b-LSP. Its format is depicted in Fig.4.5. The novel TLV to be incorporated in the SENDER_TSPEC object is illustrated in Fig.4.6 and specifies the b-LSP traffic requirements, setting an upper bound on the volume of the expected data bursts belonging to a particular OBS service instance (committed rate + excess rate) - expressed in Erlangs. The source-computed set of wavelengths per hop along the path can be optionally inserted in this object. The FLOWSPEC object (Class-Num = 9, Class-Type =

6) has the same format as the SENDER_TSPEC object. No ADSPEC object is associated with it.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             | Class-Num (12)|   C-Type (6)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Quality of Service (QoS)  |              MTU              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                              TLVs                             ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.5: Sender_TSPEC object format (option 2).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Committed Rate   (load)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Excess Rate                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ -
|  Hop: 1 - |set of wavelenghts| |              ...             | o
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ p
|             ...               | Hop: n - |set of wavelenghts| | t
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ i
                                                                  o
                                                                  n
                                                                  a
                                                                  l
```

Figure 4.6: TLV defining the bandwidth profile and optionally the set of wavelengths to be assigned at each hop of the b-LSP route.

4.3.4 b-LSP Tear-Down

The tear-down of a b-LSP is done exactly in the same modes that defined in [RFC3473]. Here, we presume an upstream release of resources (i.e. from source towards the destination node), saving in the number of messages exchanged.

4.3.5 b-LSP Modification and Re-Routing

The modification and re-routing of a b-LSP is done as in normal LSP (see [RFC3945]). The re-routing follows the concept of "make-before-break", whereby an old path is still used while a new path is set up by avoiding duplication of resources. Then, the node performing re-routing can swap on the new path and close the old one.

In general, the LSP modification consists in changing some LSP parameters. In particular, the RSVP-TE module of the GMPLS controller records the information of all configured b-LSPs (being initiated at this node) following the Path State Block (PSB) structure defined in [RFC2209][RFC3209]. For each b-LSP, the information contained in the PSB describes a similar structure than the Path message that originally signaled it, namely, a Session, a Sender Template and an ERO object. In this way, assuming that the b-LSP would have to be modified from the source, that information in the PSB would be used to allocate/deallocate the resources supporting it.

4.4 Routing extensions

The GMPLS routing [RFC4202] also makes use of two types of routing protocols. In the context of our work, we refer only to the commonly deployed OSPF-TE protocol [RFC4203]. The OSPF-TE protocol is a link-state class of routing protocol, which dynamically adapts itself to the changes in the network. This protocol defines five type of messages, namely hello message, database description message, link-state request message, link-state update message and the link-state acknowledgement message. Despite of it, the proposed (G)OBS architecture and model requires extensions on the TE Link-State Update (TE-LSU) message only (OSPF packet type 4).

The information carried in this type of message gives a global network view and what is happening in it to the GMPLS controllers. At the same time, such information is also the basis to compute the b-LSP routes in a distributed scenario. These messages implement the flooding of Link State Advertisements (LSAs) and each one of them carries a collection of LSAs one hop further from their origin.

Anyway, OBS-related LSA objects should be added to the TE-LSU messages in order to disseminate information whenever a b-LSP is setup or tear-down, its capacity is readjusted, or even to announce a b-LSP as a TE-link to other switching domains. Opaque LSAs are used, once they provide a generalized mechanism to allow future extensibility of OSPF [RFC2328], and some specific TLVs added to it.

Table 4.2: Wavelength TE Status

Status Description	Bit Code	Switching Technology
Free	0	OCS/OBS
Reserved/Shared	1	OCS/OBS

4.4.1 New LSAs

Unlike the an LSC-LSP, the b-LSP does not have any more a 1:1 LSP/wavelength correlation per link. Instead, the resources of one b-LSP are also shared by other b-LSPs of the same QoS demand (i.e. traffic flows). This introduces the notion of *sharing degree* of the resources of a TE-link, which will be helpful to better define the b-LSPs (both route and allocated resources). At IETF, efforts are being made to define a wavelength status bitmap scheme [RWAinfo11] in order to provide detailed control over wavelength usage, which is illustrated in Table 4.2. The bit code 0 represents a free wavelength while the bit code 1 a reserved one. In the context of our work, we extend the bit code 1 meaning to represent also a shared status (i.e. a wavelength being assigned to more than one b-LSP and thus carrying traffic from different flows at a time). Note that no modification on the standard is required. The Interface Switching Capability Descriptor [RFC4202], carried in the OSPF-TE message, indicates which switching capability is active on a certain GMPLS interface, if BSC (i.e. OBS) or LSC (i.e. OCS). The reserved status is assigned exclusively to OCS, where the resources are reserved at the GMPLS signalling time.

In addition, a *Sharing Factor* (SF) metric is also introduced, which represents the number of flows (i.e. b-LSPs) crossing a TE-link. Such SF metric, which can be included in any path computation strategy, will allow to fix a maximum sharing degree of the wavelengths of a certain TE-link crossed by multiple b-LSPs, and contribute to improve the load balancing in the network and to decrease the burst loss probability, i.e., avoiding the overload of some wavelengths over others less used. Although in an indirect way, we make use of it in the RWA policies defined in the previous Chapter 3. It is also useful if wavelength continuity or conversion restriction policies are applied.

Figure 4.7 shows the Opaque LSA format which is suggested to be incorporated in the TE-LSU messages. This new LSA will carry those OBS-related traffic parameters with respect to a TE-link. A novel TLV object per QoS traffic-class is then added to it, as illustrated in Fig.4.8. Such TLV consists of three different sub-TLVs such as QoS, Load (in Erlangs) and SF metric. This information will be then kept at the TE-database of each GMPLS controller and will allow each network node to properly run the RWA policies.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            LS age             |     Options   |  9, 10, or 11 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Opaque Type  |               Opaque ID                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Advertising Router                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      LS Sequence Number                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         LS checksum           |           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                      Opaque Information                       |
+                                                               +
|                              ...                              |
```

Figure 4.7: Opaque LSA.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Quality of Service (QoS) sub-TLV                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Load in Erlangs  sub-TLV                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Sharing Factor (SF) sub-TLV                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.8: New TLVs and sub-TLVs to be carried by the OBS-related LSAs of the TE-LSU messages.

Chapter 5

Results and Analysis

In this Chapter, we present an extensive simulation campaign where different network topologies and traffic scenarios (static and dynamic traffic demands) are exhaustively tested.

The objective is to show both the reliability and feasibility of the proposed (G)OBS control architecture and model, as well as, the effectiveness of the b-LSP dimensioning model aiming at absolute QoS provisioning for HP traffic demands, as agreed with potential OBS network clients through Service Level Agreements (SLAs). In addition, a dynamic, GMPLS-driven mechanism reacting to unexpected traffic variations is also shown.

Table 5.1: Messages travelling in the JA(G)OBS simulator.

Layer	Message	
	Type	**Name**
GMPLS	Signalling	RSVP-TE Path RSVP-TE Resv RSVP-TE PathErr
	Routing	OSPF-TE LSU
OBS	Signalling	BCP
	Notification	SNMP config. SNMP trap
PCE	Signalling	PCEP request PCEP answer

5.1 Simulation Environment

The simulations are executed on the ad-hoc, event-driven JA(G)OBS simulator reported in [Pedroso11a], which was designed in accordance to the architecture and model described in Chapters 2 and 3. Its architecture is depicted in Fig. 5.1.

Each (G)OBS node entity is composed by two dedicated controllers, one for OBS and another for GMPLS. In this simulation campaign, we assume they are physically co-located in a 1:1 correlation (i.e. 1 GMPLS controller per 1 OBS controller). However, a generic out-of-fiber GMPLS configuration is assumed.

The implemented GMPLS controllers generate, transmit and process GMPLS-related messages, namely RSVP-TE Path and Resv messages and OSPF-TE Link State Update (LSU) messages, in order to manage the b-LSPs.

On the other hand, the implemented OBS controllers are responsible for the burst generation and transmission, BCP processing and consequent reservation of resources in the data plane (BCP congestion is neglected). In this work, we assume one-way signalling protocol [Chen04], just-in-time (JIT) resource reservation [Wei00] and first fit unused channel (FFUC) scheduling policy, yet other OBS resource reservation protocol and scheduling policy can be used.

Each network node is both an edge node and a core switching node capable of generating bursts destined to any other nodes. It is equipped with full wavelength conversion [1] to exploit the maximum of the statistical multiplexing property of OBS, a non-blocking switching matrix, enough number of add/drop ports, one FT and one b-LSP occupancy reporting system (i.e. an O-DLRM component with SNMP Agent software).

[1]Note that it is applied to all schemes used in the simulation campaign. Nevertheless, there is available literature showing that the use of partial conversion would reduce network cost while achieving the same performance [Gauger02].

We assume source nodes do not buffer the bursts after assembly and the delay for BCP processing (i.e. offset-time) is compensated by a short extra fibre delay coil (FDC) of appropriate length at the input port of the node (E-OBS architecture is assumed - more details are found in [Klinkowski09c]). Albeit nodes are not enhanced with fiber delay lines (FDL) for buffering purposes.

The communication between both controllers is done through the exchange of SNMP messages to keep track of the system's conditions as explained in Section 2.3 of Chapter 2: OBS sends SNMP trap messages with resource usage information (e.g. output link usage), while GMPLS sends SNMP configuration messages (e.g. forwarding table updates).

A centralized node deploying a PCE is also considered in the dynamic scenarios. In such a way, it is connected to every (G)OBS node through the GMPLS controller, being equally distant. In this case, a TCP session is established per GMPLS controller upon the first b-LSP request and kept open during the entire simulation.

5.1.1 Network Topologies

Two reference network topologies are used in order to claim for topology independence, namely, the NSF network [Claffy93] with $|\mathcal{N}| = 14$ nodes and $|\mathcal{L}| = 21$ links, and the EON network [Maesschalck03] with $|\mathcal{N}| = 28$ nodes and $|\mathcal{L}| = 41$ links (see Appendix).

Each network link supports $W = 32$ bidirectional wavelengths with a transmission rate of $C_\lambda = 10Gbps$ and each network node has only 1 input/output fiber.

The propagation time is defined as $t_{prop} = \frac{d(l)}{c}$, where $d(l)$ is the distance (km) of link $l \in |\mathcal{L}|$ and c the light speed (in a fiber environment is said that it is 30% less the speed of the light in the vacuum space). The transmission time as well as the OBS controller processing and switching time are neglected as they are strictly small.

5.1.2 Traffic Model

Let the total network load (E_T), expressed in Erlangs, be defined as the total traffic load that is injected by all nodes into the network. Each node is offered with a traffic load that occupies ρ percents of the link capacity, and thus $E_T = \rho \cdot W \cdot |\mathcal{N}|$. The offered traffic is normalized to the transmission bit-rate and expressed in Erlang. In our context, an Erlang corresponds to an amount of traffic that occupies an entire data wavelength, for example, 51.2 Erlang indicate that each edge node generates 512 Gbps (i.e. $1E = 1\lambda = 10Gbps$).

Two classes of service are defined to the traffic being offered: one high-priority (HP) class, representing quality-demanding traffic, and a low-priority (LP) class, serving traffic not demanding any specific QoS and also referred to as best effort (BE) traffic. The HP traffic (E_{HP}) can be seen as a premium service providing different QoS levels, where the b-LSPs are dimensioned accordingly to the route policies described in Chapter 3. On the other hand, the BE traffic (E_{BE}) is always on transit across the network, limited to use the spare network capacity. Different HP-BE network traffic ratios are defined during the simulations. We can defined the correspondent traffic loads as follows:

$$E_T = E_{HP} + E_{BE}, \quad E_{HP} \ll E_{BE} \tag{5.1}$$

$$E_{HP} = \alpha \cdot E_T \quad (Erlang) \tag{5.2a}$$

$$E_{BE} = (1 - \alpha) \cdot E_T \quad (Erlang) \tag{5.3a}$$

where α is the percentage of HP traffic being offered to the network. Usually, the HP load is lower than the BE load.

Burst Generation

There is a long lasting discussion about which distribution type to apply to burst arrivals. Pareto distribution is the long-tail distribution that better represents a self-similar traffic generation, however there is who states that the traffic inside the OBS network does not need to be necessarily "bursty". It all depends on how the burst assembly and buffering are done. In this study, as in many others in this research area, we assume the bursts are generated according to a Poisson arrival process and have exponentially distributed lengths. The mean burst duration is set to $27.7\mu s$ (still we have observed that the model behavior does not change with other burst duration values) and it is given by a exponential distribution

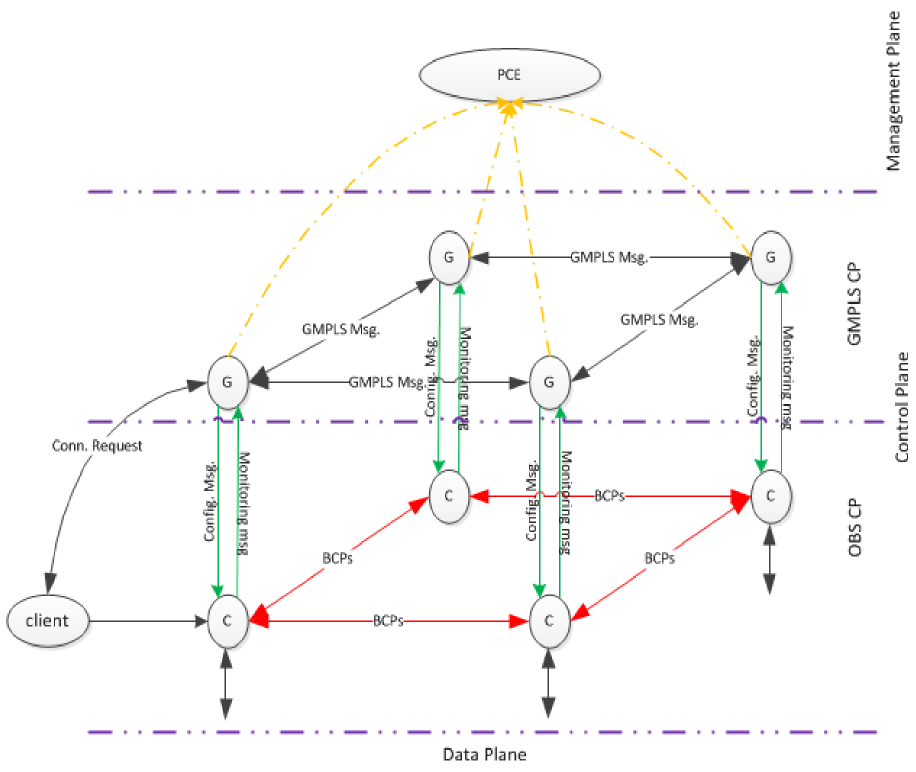


Figure 5.1: The Architecture of the JA(G)OBS Simulator.

function with rate:

$$\lambda = \frac{\rho}{\mu} \quad (bursts/sec) \tag{5.4}$$

where ρ is the load in Gbps and μ is the service time given by the mean burst duration in terms of bits (i.e. length).

The traffic is uniformly distributed between the network nodes. We assume that each edge node offers the same amount of burst traffic to the network. Apart from that, a non-uniform traffic pattern is equally considered in some cases, e.g. the b-LSP dimensioning validation in the static scenario section.

Connection (b-LSP) Generation

In a dynamic scenario, the matrix of traffic demands varies during the simulation period. Note, however, that in this study only HP traffic is assumed to vary and consequently, the BE traffic is always constant. In such a way, the number of requested b-LSPs to HP traffic transmission, as well as the duration of each one of them should reflect the overall network traffic occupancy desired to the simulation run. From the total HP offered load to the network (E_{HP}), we can deduce the HP offered load to each edge node ($E_E = \eta * \frac{E_{HP}}{N}$): $\eta = 1$ if an uniform traffic distribution is assumed; $\eta > 0$, otherwise. Such offered load must be then transmitted through several b-LSPs established between such edge node and $N-1$ possible destination nodes of the network, along the time:

$$\overline{E}_E = |\overline{LSPs}| \cdot \overline{\phi} \tag{5.5}$$

where $0 < \phi \leq 1$ is the desired average occupancy of the b-LSP capacity. The average number of b-LSPs established by each edge node ($|\overline{LSPs}|$) is given by the ratio between the average b-LSP duration, referred to as Holding Time (HT), and the average frequency that each b-LSP is requested, referred to as Inter-Arrival Time (IAT):

$$|\overline{LSPs}| = \frac{\overline{HT_E}}{\overline{IAT_E}} \Rightarrow \overline{E}_E = \frac{\overline{HT_E}}{\overline{IAT_E}} \cdot \overline{\phi} \tag{5.6}$$

The $\overline{HT}$ and $\overline{IAT}$ values follow an exponential distribution with rate $\lambda_{HT} = \frac{1}{\overline{HT}}$ and $\lambda_{IAT} = \frac{1}{\overline{IAT}}$, respectively. In turn, the average HP offered load to be exchanged between each pair of edge nodes (sd) - which corresponds to the average load of a b-LSP, $\overline{\rho}_{LSP}$ -, is given by:

$$\overline{E}_{sd} = \frac{\overline{E}_E}{(N-1)} = \overline{\rho}_{LSP} \tag{5.7}$$

Table 5.2: Simulation Configuration Parameters.

Parameters	**Value**	**Parameters**	**Value**
PCE	E/D	b-LSP ϕ	1
Preemption	E	b-LSP IAT	80 s
λ conversion	E	b-LSP HT	variable (f(load)) s
Burst Size (mean)	$27.7\mu s$	OBS processing time	neglected (too small)
BLP (i.e. QoS level)	$\{10^{-3}, 10^{-4}, 10^{-5}\}$	GMPLS processing time	1 ms

E=enable D=disable

which makes the $\overline{IAT}$ per edge pair bigger:

$$\overline{IAT}_{sd} = \overline{IAT}_E \cdot (N-1) \tag{5.8}$$

In Exp.5.6, four variables must be defined. In our simulation runs (see Table 5.2), we settle $\overline{IAT_E}$ and $\phi =]0,1]$, while ($\overline{E}_E$) varies according to the execution iteration. The $\overline{HT}$ is set then to achieve the desired $\overline{E}_E$ at each edge node,

$$\overline{HT_E} = \overline{IAT_E} \cdot \overline{E}_E \tag{5.9}$$

Whenever an overlap of HP traffic transmission requests occurs between the same s-d pair, i.e., $HT > IAT$, they are handled as independent b-LSP requests. In a nutshell, a b-LSP is requested at every $\overline{IAT_E}$ and is characterized by:

- ✓ i) source-destination (s-d) pair
- ✓ ii) load and QoS requirements
- ✓ iii) HT, i.e. the b-LSP lifetime;

The generation of HP traffic to a certain b-LSP starts only after the b-LSP itself has been established, i.e., as soon as the GMPLS RSVP-TE Resv message is received at the GMPLS controller of the source edge node, it announces it to the OBS controller counterpart, which immediately starts the transmission. Conversely, the generation of BE traffic starts at the beginning of the simulation as it will be always traveling in the network.

5.1.3 Performance Metrics

The burst loss probability (BLP) is calculated in a cumulative way, i.e., taking into account all bursts offered and lost in the network until a given instant of time. The results are obtained with a full burst preemption mechanism implemented. In this mechanism, we consider that the BE bursts are allowed to use unoccupied wavelengths that are allocated to the VT HP and be preempted whenever a HP burst needs these resources. The requested QoS constraints for HP traffic demands are set in terms of BLP.

In the dynamic scenario, we observe the usage of wavelengths and the number of active b-LSPs as function of time and assess the setup latency of the two-way GMPLS signalling process. Some BLP figures are shown as function of time samples.

5.2 Static scenario

In a static traffic scenario, the traffic demands are not known beforehand and thus do not change during a simulation.

A full-mesh topology of b-LSPs to route HP-class traffic is set-up according to the off-line MILP dimensioning model defined in Section 3.7 of Chapter 3. There are $N*(N-1)$ b-LSPs in each network topology. Conversely, the BE-class traffic is routed using the constrained-shortest path heuristic algorithm described below. The HP-BE traffic ratio is set to $40\% - 60\%$.

5.2.1 Constrained routing for BE b-LSP

The establishment of the VT for HP traffic may increase the burst loss figures experienced by the BE traffic. For this reason, we consider a constrained shortest path (SP) routing being computed at the GMPLS routing engine of each source node in order to determine the b-LSP routes for BE traffic. In fact, the BE routing can be highly improved by being HP-class b-LSP aware. Thereby, by knowing the wavelengths committed at each links for the HP b-LSP routes, a constrained SP heuristic is devised. Specifically, in addition to the common hop count metric, we take also into account the number of committed HP-class wavelengths of the bottleneck

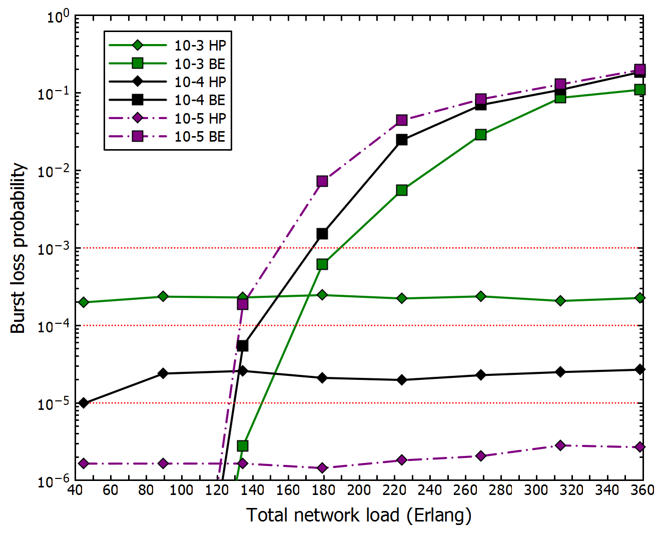


Figure 5.2: (G)OBS control model: validation of the b-LSP dimensioning model with respect to 3 diff. QoS demands 10^{-3}, 10^{-4}, 10^{-5} in NSF network.

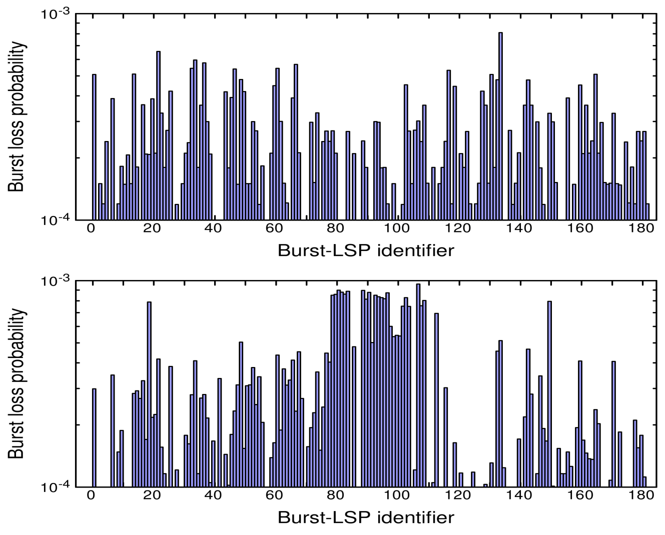


Figure 5.3: HP burst losses with uniform (top) and non-uniform (bottom) traffic. QoS threshold = 10^{-3}.

link of the computed spatial route,$\lambda_{committed}$. k-SP routes are computed and the least cost path between the source-destination pair is selected. The cost function is defined as follows:

$$c = \frac{H}{D} + \frac{\lambda_{committed}}{\lambda_{total}} \tag{5.10}$$

where H is the hop-count value for the k-computed shortest path route, D is the highest hop-count shortest path in the network, and $\lambda_{total}$is the maximum wavelengths per link. To prevent the overloading of BE routes over the same links, the route is selected only if the cost gain compared to the shortest hop-count path is above a given threshold (e.g. 20%).

5.2.2 Analysis of Results

1) Validation of the b-LSP dimensioning model

The first objective is to validate the proposed dimensioning model of HP-class b-LSPs, described in Section 3.7 of Chapter 3, on real network topologies. Figure 5.2 illustrates the overall behavior of the model for different QoS, namely 10^{-3}, 10^{-4} and 10^{-5}, in the 14-node NSF network. As expected, the proposed model accomplishes successfully for all the three HP-class traffic thresholds and under any network load scenario. In fact, we can state that the proposed model is independent from the network load and QoS demands. As regards the BE-class traffic performance, we would like to highlight that only in exceptionally high loads, the 10^{-1} threshold is overcome. The performance of the BE-class is further improved by applying the intelligent constrained SP heuristic and burst preemption.

Besides the optimization of the overall network resource usage, the model also does each individual b-LSP to strictly meet the required QoS demands. Figure 5.3 is focuses on the previous 10^{-3} QoS network scenario, considering a network offered load of 224 Erlang. As can be seen, the experienced BLP in all individual b-LSPs is below the demanded BLP, for both uniform (top) and non-uniform (bottom) traffic demands. These results not only validate the proposed dimensioning model, yet show that it behaves independently of the traffic volume and distribution in the network.

2) Benchmarking of (G)OBS

In order to position our model among other state-of-the-art proposals, comparisons with some of the current best practices are performed. We compare the performance of the proposed (G)OBS architecture against conventional OBS architectures either augmented with the classic Burst Preemption mechanism (relative QoS techniques) or the Dynamic Wavelength Grouping (DWG) policy [Zhang04] (absolute QoS techniques). Both techniques route their traffic (HP and BE) over shortest path routes. The numerical results are extracted from both 14-node NSF and 28-node EON networks. Only the uniform matrix of traffic demands is considered here.

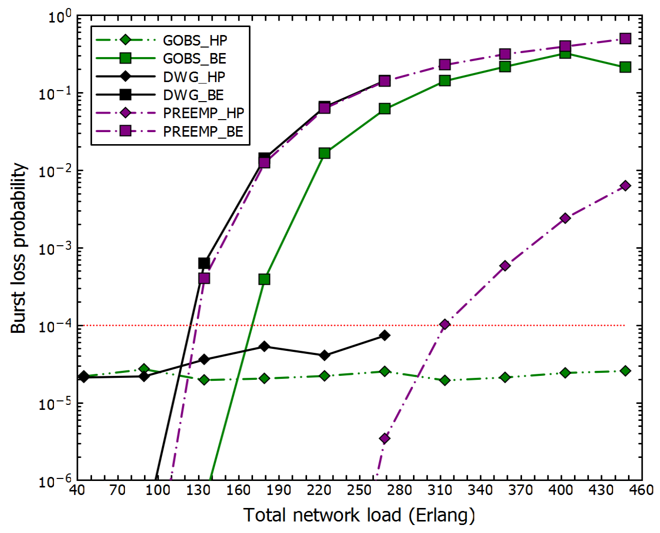


Figure 5.4: NSF network: HP-BE traffic ratio of 40%-60% for the 3 QoS techniques, i) (G)OBS, ii) DWG and iii) Preemption.

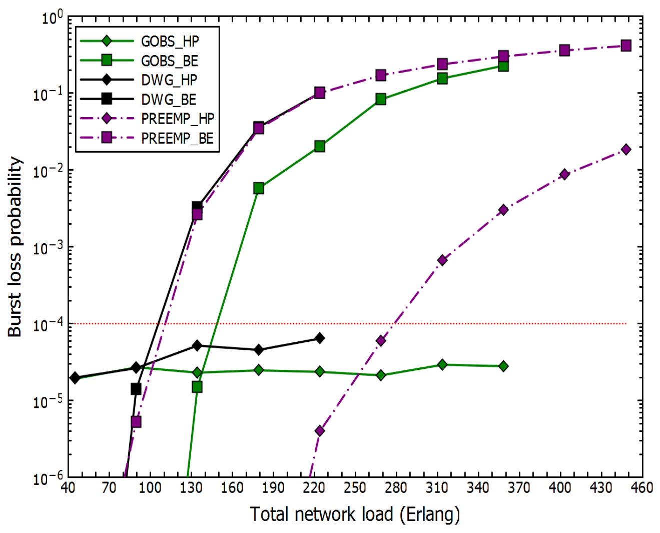


Figure 5.5: EON network: HP-BE traffic ratio of 40%-60% for the 3 QoS techniques, i) (G)OBS, ii) DWG and iii) Preemption.

Figures 5.4 and 5.5 illustrate the comparison of the proposed model against those two benchmark techniques. Quite similar behavior is observed in both network scenarios. From both figures, it can be easily observed the outstanding performance of the proposed model over those two QoS techniques in any network scenario (either topological or load), even for BE-class traffic.

Regarding the HP-class traffic performance, it remains constant and considerably below the demanded QoS (10^{-4}) for the (G)OBS case; it grows (almost linearly) with the offered load in the DWG case, although never crossing the QoS threshold; and in the Preemption case, it grows exponentially with the offered load (common OBS behavior), crossing the demanded QoS threshold at approx. 310 Er. in the NSF network and at approx. 280 Er. in the EON. Note, however, that the DWG policy cuts off (i.e. it can not cope with the traffic demands) at approx. 268 Er. and 313 Er. in the NSF and EON networks, respectively. This means that there are one or more links without enough wavelengths to satisfy such offered load. Nothing is suggested in [Zhang04] to overcome such constraint. In contrast, the (G)OBS model goes much further due to a better and more intelligent resource usage by its b-LSPs, which allows it to support higher loads (approx. 40% more). Although it cuts off for an approx. load of 400 Er. in the EON network, it equally supports a 37.5% higher load than DWG. Both models run out of channels to commit to HP-class traffic owing to EON topology intrinsic properties, which has some nodes concentrating a lot of traffic and consequently routes. In the Preemption case, there is no limiting load theoretically, although in practice both HP and BE performances become even worst as the load increases, being impractical after a certain value. In fact, the HP-class BLP in the Preemption case is already one order of magnitude above the demanded threshold at approx. 300 Er. in the EON network and 380 Er. in NSF network. Note also that the break-even point is achieved much before in the EON than in the NSF network.

On the other hand, the BE-class traffic performance of the GOBS performs again better that the DWG policy and at least as good as the Preemption technique (for higher loads), in both network scenarios. However, we shall reinforce that the primary goal is to demonstrate that our model guarantees the QoS for HP traffic. BE burst preemption is applied both in the (G)OBS and the DWG techniques, which explains why DWG and Preemption BE BLP values go side by side along the load range. On the other hand, the (G)OBS performs even better due to its resource usage optimization and constraint-SP heuristic for BE traffic. It is worth to mention that the previous HP BLP values for the Preemption case are achieved at expense of worst BE performance. Note that in lower loaded scenarios, the differences between (G)OBS and Preemption BE BLP values span from one up to two orders of magnitude (while preemption has not any HP losses - too far from the demanded QoS). In Fig. 5.4, an unexpected decrease of the BLP between 430 and 460 Er. is observed. This means that for this particular case, the model finds a distribution to

Table 5.3: % of Resource Usage by (G)OBS and DWG QoS Techniques in NSF and EON networks to allocate HP traffic demands.

Load (Erlang)		44.8	89.6	134.4	179.2	224.0	268.8	313.6	358.4	403.2	448.0
NSF	GOBS	23.21	31.37	38.99	45.39	50.89	56.70	62.43	67.56	72.62	77.75
	DWG	25.89	33.78	40.48	46.43	52.53	57.29	/	/	/	/
EON	GOBS	22.79	30.41	36.93	42.42	47.52	52.93	63.64	67.84	/	/
	DWG	23.86	30.56	36.13	41.62	46.42	/	/	/	/	/

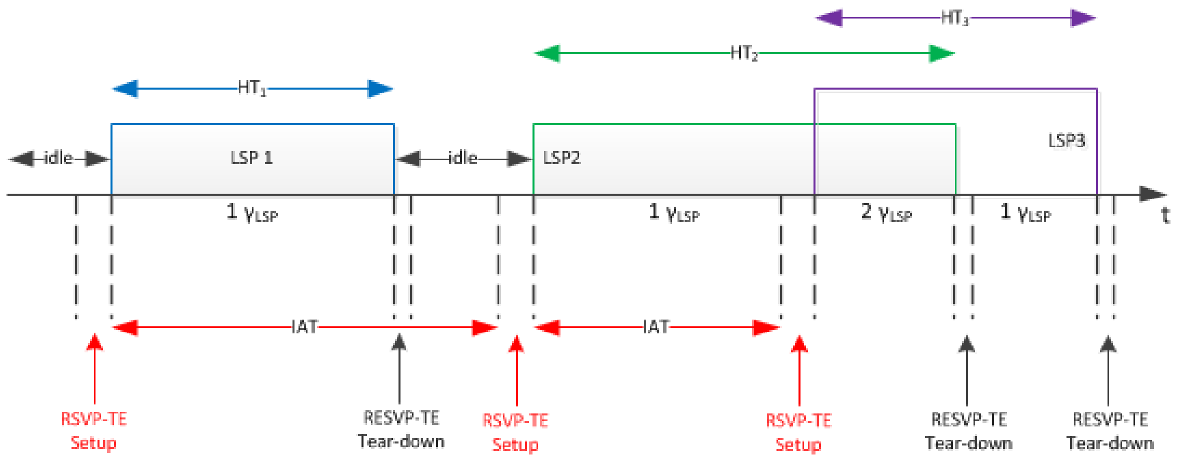


Figure 5.6: Set-up and tear-down of b-LSPs. An overlap of b-LSP requests is also shown (between LSP_2 and LSP_3).

the b-LSPs such that it "releases" more resources to the BE traffic.

Regarding the resources (i.e. wavelength) usage, it is interesting to see how the three approaches make use of them to satisfy the same requirements (either load or QoS). The (G)OBS and the OBS augmented with the DWG models share the same principle, yet (G)OBS provides it in an optimized manner (i.e. better resource distribution) and most importantly, provides it from a (GMPLS) control plane perspective. In table 5.3, one can see that there are sight differences for highly loaded scenarios. However, better results in terms of BLP, both for HP and BE, are achieved by GMPLS/OBS model.

For instance, more wavelengths are used by DWG than by the (G)OBS to provide the same BLP order of magnitude, 10^{-4}, in the NSF network. It starts with a 2,68% usage gain to end up with a 0,60% gain to a load of 268.8Er. (note that GMPLS/OBS continues while DWG cuts off). In the EON network, the differences are even smaller. In fact, after 134.4 Er., the OBS with DWG uses less total wavelengths than GMPLS/OBS. Nevertheless, due to an optimized distribution of the the b-LSPs and its capacity, the (G)OBS model can cope with much more load than the DWG model.

Moreover, the HP-class BLP behavior of the proposed model is more stable along the load range (due to a better wavelength assignment per b-LSP). We remind that the classic Preemption model makes use of the entire set of wavelengths in the each link (i.e. 100% of usage), much more than the other two models.

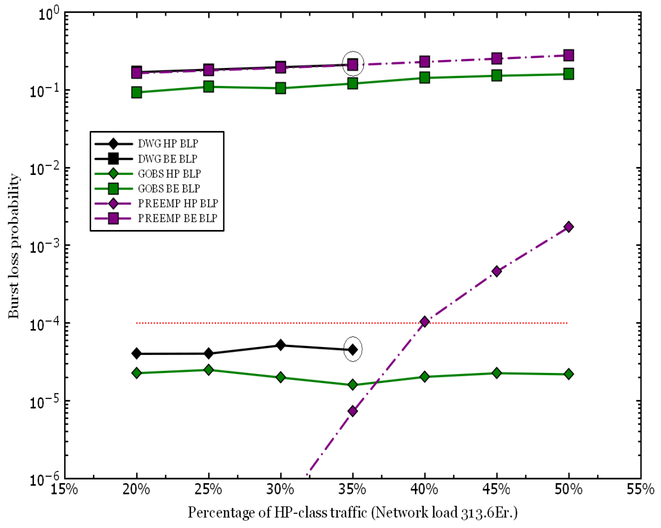


Figure 5.7: BLP behavior according to the % of HP-class traffic in the NSF network.

In Fig.5.7, we fix the network load (313.6Er.) and vary the percentage of HP-class traffic in the network. Once again, the proposed model outperforms compared to the other two. As regards the classical burst preemption, the break-even point occurs at 40% to the HP-BLP curve with a 10^{-4} QoS threshold, while the BE-BLP curve is always above the one of the proposed model. The DWG model suffers the same problem than before. It does not support a percentage of HP-class traffic higher than 35%.

5.3 Dynamic scenario

In this Section, we evaluate a fully dynamic (G)OBS network scenario, i.e., traffic demands are not known beforehand and thus no VT is pre-configured. Moreover, both GMPLS signalling and routing mechanisms act dynamically to setup and tear-down the b-LSPs and to disseminate network status information for further path computation, respectively. The HP-BE traffic ratio is set to $25\% - 75\%$, yet the model copes with any traffic ratio, as shown before.

Figure 5.6 illustrates a sequence of b-LSP requests as function of time, and some of its properties (e.g. b-LSP HT and node IAT) and some of the signalling actions that can be taken (RSVP-TE setup and tear-down). For instance, let us assume a source edge node s which receives several requests to transmit HP traffic to different destination nodes along the time. Whenever a new request arrives to the GMPLS controller of such node s, the GMPLS RSVP-TE signalling is initiated (by sending a Path message).

In turn, the reverse way is done by the RSVP-TE Resv message. Each intermediate node confirms the previous assigned resources upon its reception, and a SNMP configuration message is sent to the OBS controller counterpart with the updated FT. In the same way, OSPF-TE routing messages are generated and disseminated through the network, containing the new LSA objects as defined in Section 4.4 of Chapter 4. Each GMPLS controller updates then its TE-database which will allow later on to compute the path in a total distributed way according to the "current" network status. When the Resv message finally arrives to the node s, the HP traffic transmission itself can start. Remember that the RSVP-TE signalling is a two-way process.

In this study, we assume the RSVP-TE setup time comprises the time taken by each intermediate node to locally compute and assign the proper resources to such request plus the two-way propagation time of the signalling messages in each link of the path, t_{prop}. The processing time of each signalling message (t_{proc}) at each GMPLS controller is divided in two categories whether it is going on the upstream or downstream direction. Hence, the t_{proc}^{UP} is set to $1ms$ as more operations are to be executed and the t_{proc}^{DOWN} is set to $0.5ms$ as nodes should only confirm or deny the previous assigned resources (i.e., FT updates - no physical reservation). The transmission time (t_{tx}) is neglected due to the small size of the messages. The b-LSP route computation time, t_{RWA}, is equally excluded as it highly depends of the routing strategy adopted. The setup time (t_{setup}) is then given by:

$$t_{setup} = (N_p - 1) * t_{proc}^{UP} + N_p * t_{proc}^{DOWN} + (N_p - 1) * 2 * t_{prop} \tag{5.11}$$

where N_p is the number of nodes part of the b-LSP route. In the case where a centralized PCE node is considered, the t_{setup} increases as it must take into account the time taken between the (two-way) (G)OBS-PCE communication, t_{PCE}, as well as the inherent processing time at the PCE node (note that the t_{RWA} is not considered here either). In addition, the first request from a given (G)OBS source node to the PCE requires the establishment of a TCP session ($3 * t_{prop}$).

Each network node will receive a b-LSP request every $\overline{IAT} = 80s$. Its duration is given by the Exp.5.9 and depends on the network load scenario being run. As the $\overline{HT} > \overline{IAT}$, the overlap of requests occurs, as can be seen in Fig.5.6 (e.g. between LSP_2 and LSP_3). If not, an idle period follows the end of any b-LSP transmission. The RSVP-TE tear-down process (one-way) is executed as soon as the HT of the b-LSP ends.

This being said, a VT augmentation with routing optimization scenario is considered. The RWA policies described in Section 3.8 of Chapter 3 are used, namely the on-line MILP dimensioning (serving multiple requests) deployed at a single centralized node (or PCE node). An average distance of $1000km$ is assumed between itself and each (G)OBS node of the network. In this case, multiple b-LSP requests from different source nodes arriving within a time frame (w_{PCE}) of $6ms$ are processed together. As an alternative, the k-SP heuristic enhanced with route optimization (serving single requests) is also used. It is considered to be deployed both from the source and the PCE node. It aims to minimize the increment of the number of total wavelengths (I_w) used to allocate HP traffic upon the arrival of a new b-LSP request. From a set of paths P, ($|P| = k$), it selects the one that requires less wavelength to be assigned besides the ones already assigned to other b-LSPs of the same QoS category. This information is extracted from the TE-database of the GMPLS controller which keeps both the number of wavelenghts assigned per QoS level and per link, U_w, as well as the current load status of the links, ρ_l. The requested load, ρ_{req}, to be transmitted is extracted from the T_SPEC object of the Path message (as well as the QoS level).

Last but not least, the SP algorithm is used as the reference. If not mentioned otherwise, the e2e BLP of the HP traffic supported by the VT is on the level of 10^{-4} and the network load is equal to $E = 224$ Erlangs in those figures where the x's axis refers to time.

5.3.1 Analysis of Results

The NSF ($|\mathcal{V}| = 14$ nodes) network topology is used to assess the performance of the different routing strategies. We observed (1) the BLP as function of the network load, (2) the BLP of a specific e2e pair of source-destination nodes (with several b-LSPs) along the time, (3) the implications of the setup latency of the two-way GMPLS signalling procedure and (4) the usage of wavelengths in VT according to the number of active b-LSPs in the network at a given moment. We also analyzed the usage of the PCE deployed at a single centralized network node, as well as, the influence of the on-line MILP window time in the different BLP figures (HP, BE and overall).

The three different routing approaches are depicted in all figures, namely, the SP algorithm, the k-SP heuristic (run at the source) and the on-line MILP formulation (run at the PCE node). To the best of our knowledge, there are no performance studies available in the literature regarding absolute QoS provisioning in a dynamic GMPLS-OBS scenario.

BLP as function of network load and time Figure 5.8 illustrates both the achieved BLP (left y'axis) and the number of simulated bursts (right y's axis) in function of the network load (E_T) for the NSF network.

As in the static case scenario, the model shows a constant behavior for the HP BLP curve, which remains independent of the network load and regardless the routing approach itself. The BE BLP curve maintains the classical shape of an OBS network without any contention resolution technique, although the $= 10^{-1}$ BLP threshold is only reached in exceptional highly loaded scenarios. Quite similar performances are achieved among the different routing approaches. The variations observed in the HP BLP curve for the MILP case are due to the use of the SP algorithm to route the requests individually whenever the MILP can not find a proper solution.

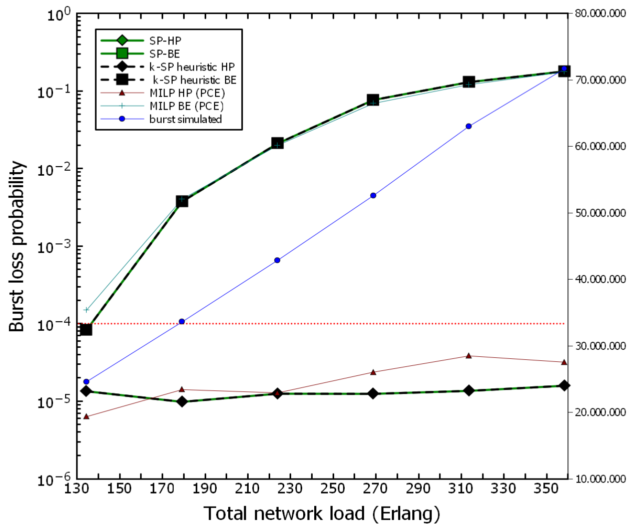


Figure 5.8: BLP figures in function of the total network load. HP-BE ratio of 25%-75%.

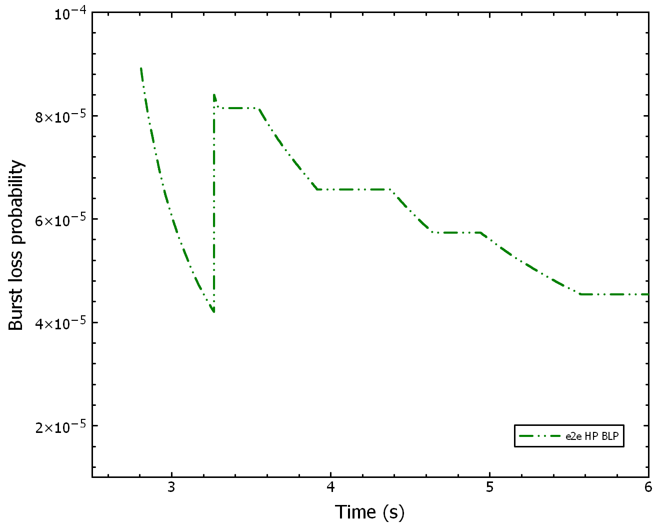


Figure 5.9: Time sample (from 3 to 6 seconds) of the HP BLP for a e2e pair of (s-d) nodes.

Table 5.4: Blocking % of quality-demanding burst flow requests for the 3 diff. routing approaches.

Load (E_T)	224	268.8	313.6	358.4	403.2	448
SP	0%	0.11%	0.79%	2.50%	4.54%	5.45%
k-SP Heuristic	0%	0%	0.11%	0.23%	1.36%	2.04%
on-line MILP	0%	0.23%	1.14%	1.70%	4.42%	5.33%

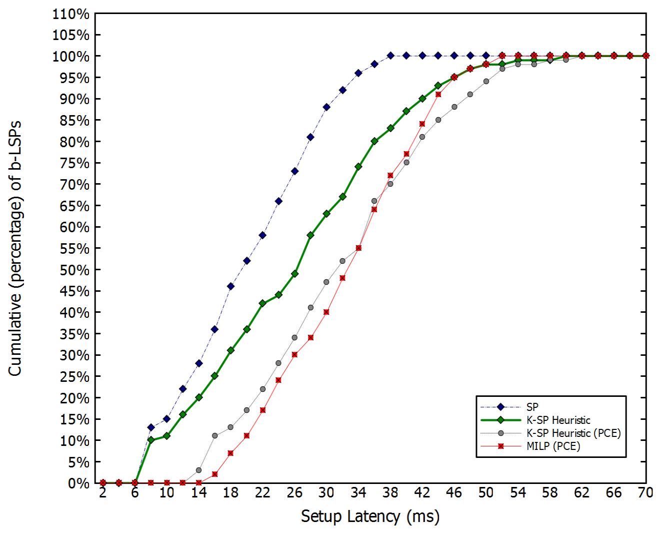


Figure 5.10: Setup latency of the GMPLS signalling two-way procedure in the 14-nodes NSF network. $E_T = 224$ Erlangs.

In fact, the main difference among the three routing approaches is found at the percentage of blocked b-LSP requests. For traffic loads above 268 Erlangs, we notice the blocking of some requests of dynamic burst flows due to the lack of wavelength resources in the network. Table 5.5 contains the blocking percentage for the different traffic loads (from 268 to 448 Erlangs).

The on-line MILP blocking probabilities are somehow unexpected. This might be related with the fact that the in-place set-up b-LSP can not be changed and the optimization is though limited to the new set of b-LSP requests arriving within the time window. Moreover, SP is the alternative routing if the MILP does not find a proper solution which is a limited approach. The k-SP heuristic will be used in the future.

In Fig.5.9, we illustrate the HP BLP of a specific e2e pair of nodes in function of the time (note that the accomplishment of guaranteed e2e BLP per b-LSP was already shown in the static scenario and it is observed here again). The time sample is taken for a period between 3 and 6 seconds. From the figure, one can confirm that the QoS threshold (10^{-4}) is not crossed at any rate. In addition, one can observe a sudden variation in the BLP curve as a result of a burst loss around the 3.2 seconds.

Setup Latency Figure 5.10 shows the cumulative percentage of b-LSPs that accomplish a certain setup latency as a result of the two-way GMPLS signalling procedure in the NSF network. The setup latency is given by Exp.5.11, using the parameters defined in Table 5.2. Slight differences are visible.

The use of the PCE node (either running the k-SP heuristic or the on-line MILP routing algorithm) results in worst setup latency times due to the extra two-way communication between the (G)OBS and the PCE nodes and the PCE processing time (the t_{RWA} is not considered), $t_{PCE} = 2 * t_{prop} + t_{proc}^{PCE}$ - ($t_{proc}^{PCE} = 1ms$). It is even worse when the online MILP is applied as it only computes after T time (i.e. temporal window) in order to collect the arriving b-LSP requests to be processed together. The crossing between the two curves regarding the MILP and the PCE k-SP heuristic means that the latter computes longer paths (in a way to minimize the number of used wavelengths). If only one request arrives during such period, the SP algorithm is run instead of the MILP in order to reduce the t_{RWA}. Note, that the minimum setup time observed is never inferior to $6ms$ when the PCE is not used, and $14ms$ otherwise (achieved for 1-hop long b-LSP).

Regarding the k-SP heuristic, it shows a slightly longer setup latency times than the SP (lower latency algorithm) due to the fact that sometimes the selected paths are not the first shortest path, i.e., $k > 1$, yet the ones which allow a lower increment of used wavelengths in the network. A degradation of those times is observed when the same algorithm (k-SP heuristic) is run from the PCE (due to extra two-way communication to the PCE). Excluding the PCE cases, we conclude that no more than 50% of the b-LSPs are setup within a time higher than approx. $25ms$ and we can fix a maximum of $< 40ms$

to SP and $< 60ms$ to k-SP heuristic, and a minimum of $6ms$ (in both cases) to the b-LSP setup time in the 14-nodes NSF network.

Accordingly to such data, we can say that the impact of the GMPLS signalling setup latency is residual compared to the average HT of the b-LSPs in the case of distributed source-routing computation. In our simulation runs, the $\overline{HT}$ varies from $192s$ ($E_T = 134.4Er.$) up to $512s$ ($E_T = 358.4Er.$). For instance, the impact of the maximum b-LSP setup latency considering an average HT of $320s$ ($E_T = 224Er.$) is approx. 0.0125% for the SP case and 0.01875% k-SP heuristic case. Even if the setup latency increases tenfold or hundredfold (due to different configuration parameters of the switching and processing times at the controllers and even including the route computation time), the impact would remain low.

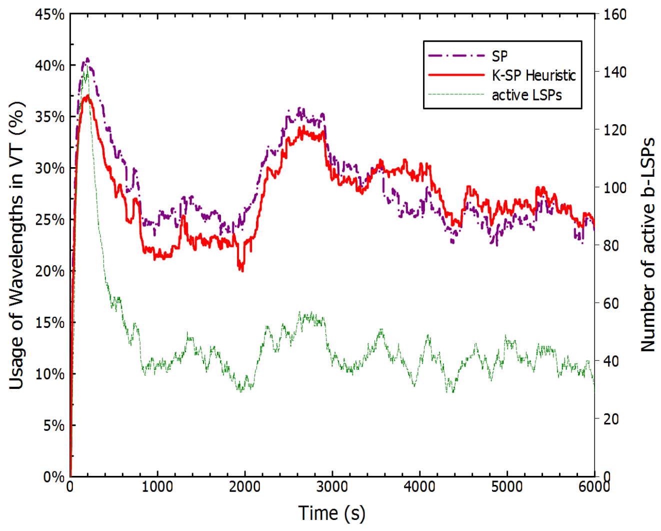


Figure 5.11: % of Wavelength Usage and the number of active b-LSPS in function of the time, $E_T = 224$ Erlangs.

Usage of wavelengths in the VT according to the number of active b-LSPs The usage of wavelengths in the VT is also an interesting metric to be observed in such a dynamic scenario. In Fig. 5.11, we depict it together with the number of active b-LSPs regarding a time sample of more than 1 hour and a network load of $E_T = 224$ Erlangs. After an initial peak of 140 active b-LSPs, it decreases and stabilizes around the 40 active b-LSPs. In turn, the number of wavelengths in the VT does not cross the 40% of the total network capacity, varying according to the arrival and departure of b-LSPs along the time.

In what regards the usage of wavelengths by the different algorithms, one can see that k-SP heuristic routing approach performs better in the first half of this time sample but not after the $3600s$. The situation reverses as we probably reach a point where we have longer paths that are not more optimizing the resources usage due to the tear-down of several b-LSPs, while the SP is in that sense oblivious (the setup of b-LSPs are completely independent). This shows that a full reconfiguration of the VT would probably be a good decision at this point.

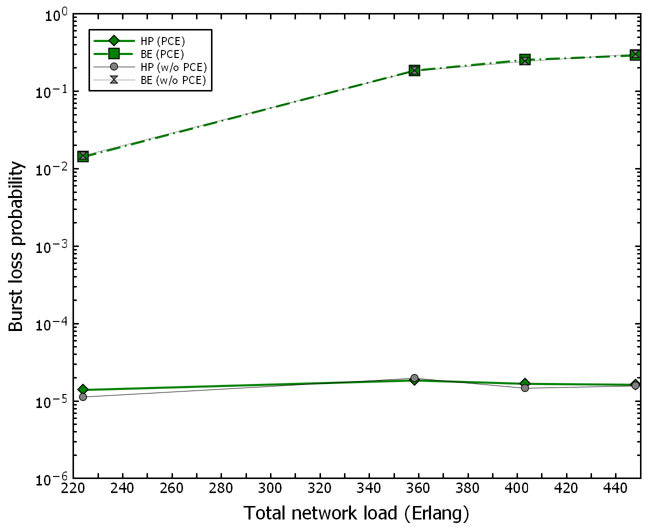


Figure 5.12: PCE influence: running the k-SP heuristic either at the PCE node or at the source node (NSF network).

With and without centralized PCE The idea here is to assess the use of the single centralized node with a PCE. Hence, we fix the routing algorithm to be used (the k-SP heuristic (k=6) in this case) and define two scenarios: i) Scenario A where we perform the b-LSP route computation at this centralized node. This node posses an unique TE-database with full knowledge about the network status; and ii) Scenario B where the b-LSP route computation is performed at the b-LSP source node based on information distributed by a link-state protocol (e.g. OSPF-TE) and kept at its local TE-database. As can be seen from Fig.5.12, the differences in terms of BLP figures are almost none. The BLP curve trends are quite similar either with or without using the PCE, albeit some really small gains are observed for the PCE case in some network load scenarios.

The main advantage of using the centralized PCE however is in terms of b-LSP request blocking probability. In this case, the use of the PCE contributes to improve the number of b-

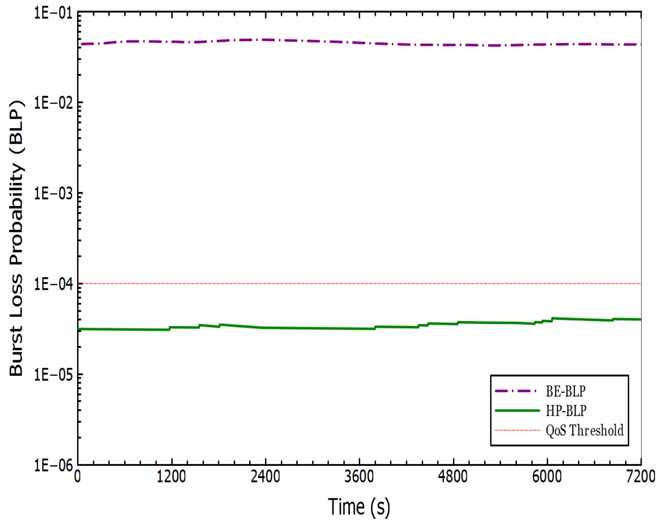


Figure 5.13: Burst Loss Probabilities for a given demand vs. Time; $E = 224$ Erlangs.

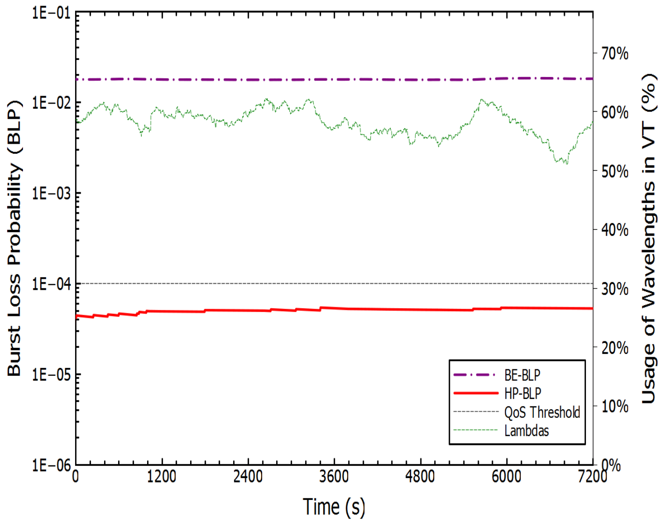


Figure 5.14: Maximum and Average Burst Loss Probabilities vs. Time; $E = 224$ Erlangs.

LSPs accepted chiefly in highly loaded scenarios ($E_T = 403$ and 448 Er.) as shown in Table 5.5. We have also observed that a bigger time window will reduce this value at the expense of longer setup latency.

Table 5.5: Blocking Probability of HP b-LSP requests using the k-SP Heuristic.

Load (ET)	224	358.4	403.2	448
without PCE	0%	0.23%	1.36%	2.04%
with PCE	0%	0.45%	1.02%	1.48%

5.4 Mixed scenario

In this Section, we evaluate the performance of the on-line VT maintenance mechanism without route changes. The results are obtained for NSF ($|\mathcal{V}| = 14$ nodes, $|\mathcal{E}| = 21$ links) network topology.

The VT is initially dimensioned, using the off-line MILP formulation presented in Section 3.7 of Chapter 3, to accommodate 70% of the HP traffic, which comes from a static matrix of uniformly distributed traffic demands that does not change during the simulation. The remaining 30% corresponds to the requests of dynamic traffic flows offered to the network. Such burst flow requests arrive in average at every $\overline{IAT} = 500$ seconds and have a mean duration of $\overline{HT} = 600$ seconds for each pair of source destination nodes. Accordingly, the average inter-arrival time of the burst flows offered to the network is equal to $\overline{IAT}/\left(|\mathcal{V}|\left(|\mathcal{V}|-1\right)\right) \approx 2.74$ seconds.

The source and destination nodes for arriving burst flows are selected according to a uniform distribution. The network either admits or rejects the burst flow requests with the assistance of the VT maintenance mechanism presented in Section 3.8. The *e2e* BLP of the HP traffic supported by the VT is on the level of 10^{-4}. If not mentioned differently, the network load is equal to $E = 224$ Erlangs.

In Figure 5.13, we present BLP results for the HP and BE traffic obtained as a function of time. These temporal samples are taken for a period of 7200 seconds for an (arbitrary) pair of source-destination nodes (i.e., demand) in the network. The VT is adapted whenever a new traffic flow arrives such that it cannot be satisfied using the current allocated resources. The BLP threshold is accomplished with respect to the HP traffic.

They show some small variations in HP BLP that take place in the network due to the arrivals and departures of burst traffic flows. We observe that the changes in the offered HP traffic load do not have a significant impact on the BE BLP results, which vary between $4.2 \cdot 10^{-2}$ and $4.9 \cdot 10^{-2}$. This is achieved thanks to the burst preemption mechanism which allows to make use of the VT resources for the BE bursts.

In Figure 5.14, we present the BLP results and the overall wavelength usage results obtained as a function of time. The samples are taken again for a period of 7200 seconds. In the Figure, the HP BLP metric represents the largest observed *e2e* BLP among all the demands at each time step. Since the

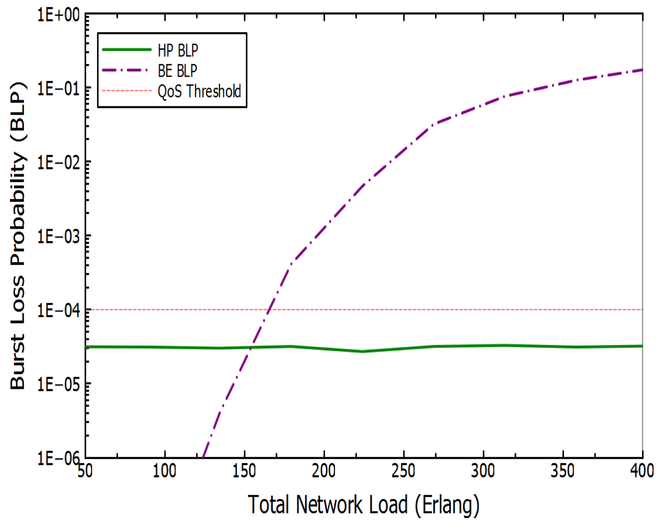


Figure 5.15: Burst loss probability vs. Total Network Load.

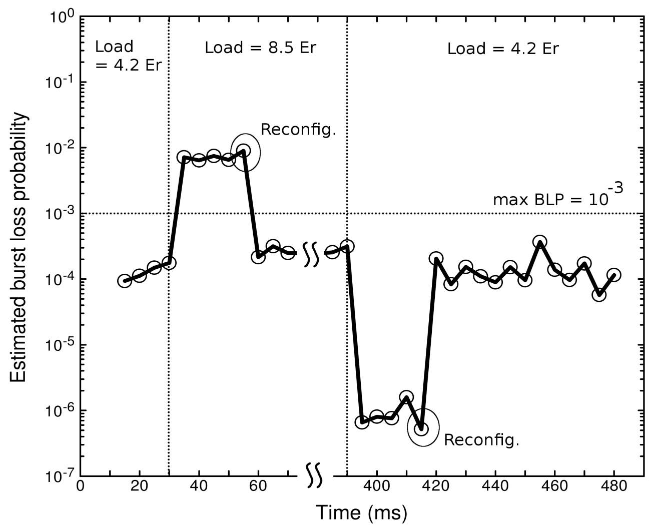


Figure 5.16: GMPLS-driven b-LSP reconfiguration.

observed results are below the level of 10^{-4}, it shows that for each demand the *e2e* guarantees are satisfied. The BE BLP represents the average *e2e* BLP with respect to the BE traffic.

Although it can be hardly observed in the logarithmic scale, the BE BLP results vary slightly. The overall wavelength usage, which is expressed as a percentage of wavelengths allocated to VT to the overall number of wavelengths in the network, varies with the arrivals of quality-demanding burst flows which cannot be served with the already allocated VT resources. Similarly, every time a burst flow is terminated, any excessive VT resources are released.

Table 5.6: Blocking Probability of the quality-demanding burst flow requests.

Load (E)	224	268.8	313.6	358.4	403.2
Blocking	0%	1.6%	4.23%	7.5%	12.42%

In Figure 5.15, we show the overall BLP results for the HP and BE traffic after the performance of dynamic network simulations. The results are obtained for different loads (E) offered to the network. We can see that the quality guarantees are provided for the HP traffic independently of the traffic load. As regards the BE traffic, the BLP increases as the load increases.

It is also worth to mention that for traffic loads above 268 Erlangs we notice the blocking of some requests of dynamic burst flows due to the lack of wavelength resources in the network. The results for different traffic loads are presented in Table 5.6.

5.5 GMPLS-driven b-LSP Reconfiguration

In order to evaluate the GMPLS-driven b-LSP reconfiguration mechanism proposed in Section 3.9, we enforce a peak of HP traffic of short duration in one of the b-LSPs of the network. This leads to the following two main operations: i) expansion of the b-LSP capacity to accommodate the extra traffic when the peak-increment is detected and on the other hand ii) return to the original b-LSP configuration after the peak-decrement time.

Figure 5.16 shows a sample of the execution time during which we monitor the traffic occupancy of an output link in the NSF network. The QoS control is performed in terms of HP burst loss probability (eBLP), which is estimated every T window with $|T| = 5ms$. From the figure, we observe that as soon as the traffic peak occurs, around $t = 30ms$ (the offered load goes from 4.2 to 8.5 Er.), the GMPLS instance detects it and remains on a holding stage during $W = 5$ consecutive T windows (i.e. 25ms) with an eBLP above the QoS level, to avoid any system instability.

Following the process as described in Section 3.9, those interface IDs of each additional wavelength are included in the specific Unnumbered interface ID sub-object maintained in the PSB at the RSVP-TE module of the GMPLS controller. Once the PSB is updated, the OBS controller is notified in order to update its forwarding table and the b-LSPs acquire extended properties. As a result, the eBLP returns to values below the QoS threshold. At approx. $400ms$, however, the offered load is reduced back to its previous value of 4.2Er (the traffic peak ends), so that the b-LSP becomes now over-dimensioned for the current offered load. Therefore, in order to achieve good resource usage, a counterpart process is triggered. After $W = 5$ consecutive T windows with extra allocated resources (i.e. wavelengths), the b-LSPs return to their initial configuration.

This reconfiguration mechanism allows the proposed architecture to properly handle unexpected traffic demands that may arise besides the VT dimensioning. It is worth mentioning, however, that long-term changes on the traffic matrix would eventually require the reconfiguration of the VT.

Summary

Part I introduces the proposed GMPLS-controlled OBS network architecture and model that leverages on the GMPLS interoperability to enable seamless vertical and horizontal OBS integration with different switching layers under a common control plane.

The outcome of this work can be summarized in three main contributions, namely i) a functional architecture, ii) a control model with RWA policies providing absolute QoS guarantees to quality-demanding (HP) traffic while achieving optimized resource usage, and iii) GMPLS protocol extensions to come with such architecture.

The b-LSP entity is here introduced as a mean to provide QoS-aware burst transport services for HP-class traffic, besides the end-to-end connectivity among different domains. In a way to optimize the network resource usage, a MILP formulation is presented to compute an optimal virtual topology of b-LSPs over the OBS data plane, defining their routes and capacities.

Two types of network scenarios are considered: a static scenario with a two QoS levels of traffic (i.e. HP and BE) and benchmark it with reference literature proposals as a first attempt to validate the model; and a dynamic scenario with fully dynamic set-up and reconfiguration of the VT of b-LSPs involving both GMPLS signalling and routing mechanisms for which specific protocol extensions were also proposed.

Also, proper RWA algorithms and policies were designed to deal with the concept of b-LSP and its computation, as well as aiming at absolute QoS guarantees. A dynamic GMPLS-driven b-LSP reconfiguration mechanism yields a successful adaptation to those unexpected and short duration variations in the agreed traffic demands, keeping the burst loss probability of the HP traffic below the requested maximum values. Extensive simulation results highlight the effectiveness of the proposed architecture and model which allows to guarantee absolute BLP figures (at the same time it improves BE traffic performance up to 2 orders of magnitude), even in high load situations compared to other benchmark QoS techniques (up to 40% more).

These contributions can be a strong step in the direction of the standardization and development of OBS networks. The Control Plane Tasks Assignment helps to achieve a sustainable network with a reasonable and acceptable performance, as shown in Chapter 5. The extensions proposed will fulfil the RFC gaps in the GMPLS/OBS interoperability process, not compromising the overall GMPLS applicability to other switching technologies, which is crucial for our model to be successful. Moreover, this work contributes to collect, organize and clarify the whole set of spread ideas related with GMPLS-based OBS Control Plane design.

In a nutshell, the proposed GMPLS-controlled OBS architecture and model contribute effectively with a solution addressing a considerable percentage of the requirements defined in Chapter 1. We assess our outcome according to the following methodology:

Strengths

· Use of GMPLS as a mature control framework to define OBS network control operations.

· Fostering GMPLS-control in multi-domain environments (MRN/MLN) even when OBS is present due to a set of protocol extensions and concepts such as b-LSP and new GMPLS switching interface to accommodate OBS.

· RWA strategies providing absolute QoS figures for quality-demanding traffic in OBS networks while optimizing the resource usage. An optimized VT modelling independent of the RWA strategy.

· Acceptable BE losses using burst preemption.

Weakness

· Model dependency of wavelength conversion (due to optical buffering technology unavailability).

· Different QoS classes can not share resources (it requires complex scheduling algorithms to be further addressed).

Opportunities

· Mid-term deployment. OBS and GMPLS technologies are pretty mature nowadays. First commercial devices.

· Minor extensions to GMPLS standards.

· Proper answer to future network demands (in terms of high and efficient network usage, QoS figures, low operational cost, unified and automatic network control).

Threats

· OBS: fail to deploy it due to the still expensive devices, or the overcome by OPS technology.

· GMPLS: late deployment due to the impasse by operators and vendors to thrust in an automatic control of their networks.

· IETF conservative approach as it aims at really short-term deployment technologies and thus to standardize solutions to current, in-place problems.

· Considerable investment to implement a GMPLS-controlled OBS network.

Part II

Routing Scalability in Internet

Chapter 6

Overview

Internet, which is supported by an IP packet-switched architecture, is facing scalability issues as an enormous quantity of service instances and devices must be managed nowadays. Current Internet features are not prepared to accommodate either such high volume or its dynamics. In particular, the scalability, convergence, and stability properties of Internet inter-domain routing and addressing systems are being questioned (as discussed in the IAB workshop in 2006 [RFC4984]). It is believed that such features and systems cannot cope with the foreseen size of the Future Internet. For instance, the current size and growth rate of the routing tables (RT) are reaching unbearable values. The memory-space consumption is already huge. Neither full information about all possible unicast/multicast sets, nor global topology information (proportional to the network size) can be stored anymore in order to cope with the aforementioned growth of the Internet and, in particular, with the boom of multimedia applications.

The main objective of Part II is to investigate new routing paradigms so as to design, develop, and validate distributed and dynamic routing schemes suitable for the future Internet and its evolution. The resulting routing schemes are intended to address the fundamental limits of current stretch-1 shortest-path routing in terms of RT scalability but also topology and policy dynamics (perform efficiently under dynamic network conditions). In such a way, we also investigate the trade-offs between RT size (to enhance scalability), routing scheme stretch (to ensure routing quality) and communication cost (to efficiently and timely react to various failures). The rationale behind it is to address directly the root causes (disruptive attitude) instead of short-term incremental fixes to attenuate symptoms (evolutionary attitude), which has been the current practice to deal with the "inter-domain" routing system problem of the Internet. Today's features have limited elasticity to satisfy the following triple trade-off: stretch x RT size scale x dynamics.

Two breakthrough paradigms on the design of a routing system scalable with the Future Internet requirements are introduced and discussed in the following Chapter 7 and Chapter 8, namely:

- ✓ AnyTraffic Labeled Routing: it is a new concept introduced by this work. A specific heuristic algorithm is devised to forward unicast and multicast traffic over the same source-initiated network entity - AnyTraffic tree - envisioning RT size reduction. In other words, a single entry is shared to forward both type of traffic, attempting to reduce specifically on the forwarding entries.

- ✓ Compact routing is well positioned as one of the main alternative to overcome the poor scaling properties of the current Internet routing system. Although "static" compact routing works fine (i.e. source-routed routing with topology-dependent addressing), scaling logarithmically on the number of nodes even in scale-free graphs such Internet, it does not handle dynamic graphs. Multi-homing and mobility require topology-independence node identifiers as any topology-driven, policy-driven or protocol-driven change event must trigger the exchange of routing messages to timely update the non-local topology information of each node in the network and the renumbering of IP devices. Hence, we propose a name-independent, leaf-initiated, dynamic compact multicast routing relying on a complete unawareness of the topology.

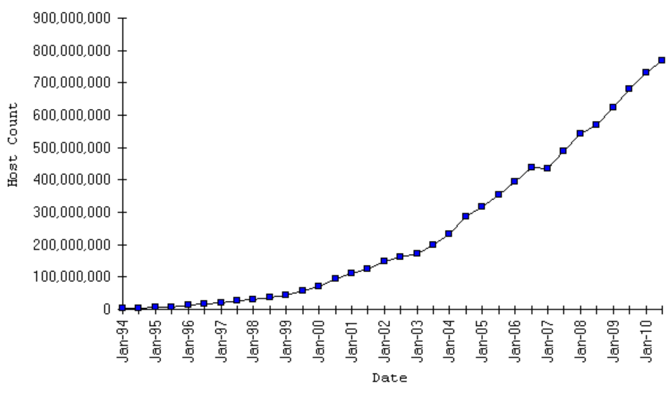


Figure 6.1: Growth in the number of Internet hosts (from Jan.'94 to Jan.'10) [BGPReports].

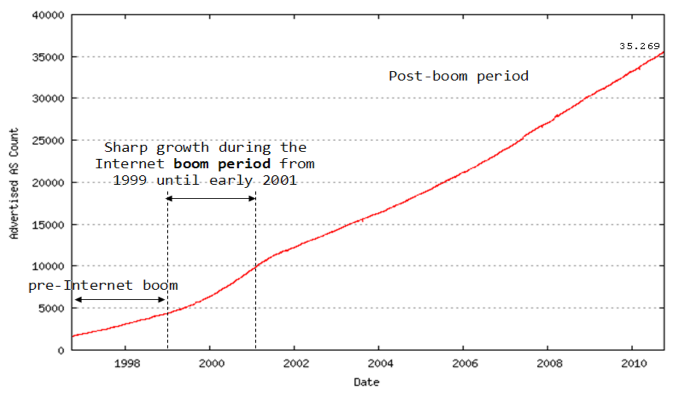


Figure 6.2: Growth in the AS number advertised in BGP RT (from 1997 to 2010) [BGPReports].

6.1 Drivers

Originally, host IP addresses were Provider Allocated (PA) and assigned based on network topological location. In the mid 90's, Classless Inter-Domain Routing (CIDR) [RFC4632] emerged to perform address aggregation which seemed sufficient to handle the address scaling problem by then. Today, however, conditions to achieve efficient address aggregation and relatively small routing tables are not met anymore [RFC4984].

New "network" requirements such as host mobility, site multi-homing (25% of sites) and traffic-engineering (prefix de-aggregation), as well as, the Regional Internet Registries (RIR) policy to allocate Provider Independent (PI) address space (not topologically aggregable), are making CIDR ineffective (as re-computation of RT is often requested).

This scenario is contributing to the current super-linear growth of the RTs of the Internet inter-domain routing system. The current inter-domain routing protocol, Border Gateway Protocol (BGP), cannot cope with these demands: it does not scale even if network size is not growing. In order to better understand the problems we have on hands, let us first analyze the major factors that are driving RT growth, and the limitations of today's Internet addressing architecture [RFC4984]. The analyzed period ranges from 1989 to 2010. The data is extracted from BGP Routing Table Analysis Reports available at [BGPReports].

What is happening?

- ✓ Growth in number of Internet hosts.
- ✓ Growth in AS number advertised in BGP Routing table.
- ✓ Growth of active BGP IPv4 and IPv6 entries.
- ✓ Multicast traffic is surpassing P2P traffic.

Each one of these items are correlated among themselves. Figure 6.1 shows the growth in the number of Internet hosts over time. The growth curve has clearly a linear crescent slope, as we are reaching the 900 millions hosts. In terms of the number of users, this is translated to a number around the two billions (as of Jun.2010). As a consequence, the traffic volume is standing around 8 to 9 Exabytes with a growth rate of 50% (+/- 5%) per year.

Figure 6.2 shows the growth in the number of Autonomous System (AS) advertised in BGP RT over time. The number of advertised AS is now at the $35k$ (Sep.2010) with a growth rate of 10% per year, but $100k$ are expected to be reached in the upcoming years. The AS-path length, which is the median of the means of shortest path lengths connecting each node to all other nodes, is steady at approx. 3.7.

Figure 6.3 illustrates the growth curve in terms of active BGP IPv4 entries in the RT (in particular, in the Forwarding Information Base (FIB)) over time. The number of active RT entries is currently on the $345k$ (Sep.2010) with a growing rate of $15\% - 25\%$ per year.

As this would not be enough, the advent of IPv6 is impacting also in the RT sizes. The IPv6 RT has grown by 50% during 2008 and 2009, and it is expected to double the current $3.5k$ entries in the next five years. Note that the IPv6

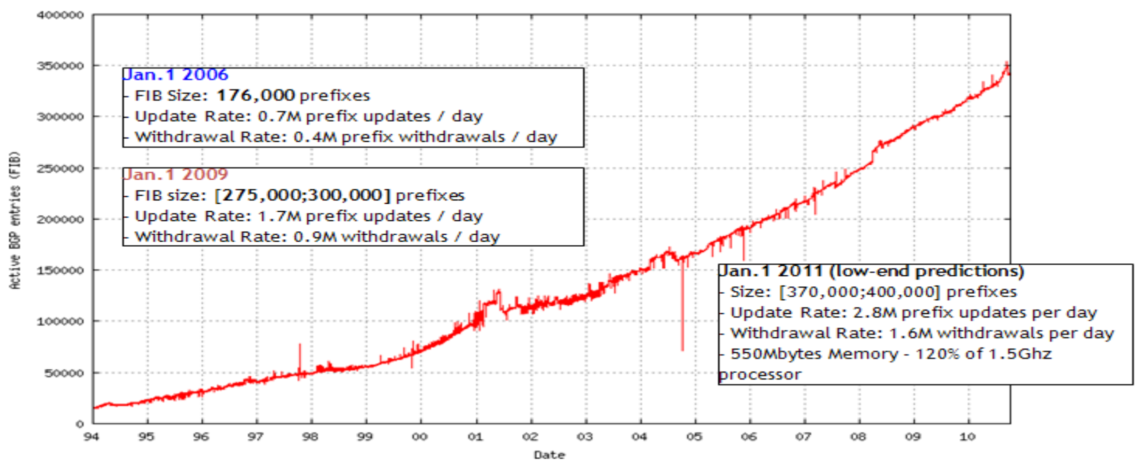


Figure 6.3: Growth of Active BGP Entries (from Jan'89 to Sep'10) [BGPReports].

requires more bits to be stored. In addition to the RT sizes, an equal important problem is the routing convergence times. The dynamics of the BGP updates due to topological failures and traffic engineering (prefix de-aggregation) is affecting its convergence time (huge update rates, which in turn increases load on routers). The BGP's path vector also amplifies these problems (path exploration).

Another interesting point is the multicast content distribution (point-to-multipoint (P2MP) or multipoint-to-multipoint (MP2MP)). The advent of multimedia stream/content has brought the multicast topic again to the spotlight as it is a bandwidth saving technique competing with or complementing cached content distribution. In fact, multicast (video-multimedia) traffic has surpassed the unicast (i.e. point-to-point (P2P)) traffic in Internet according to Cisco [CISCO10]. The result is a strong competition for resources either by unicast and multicast traffic.

Nevertheless, the scaling problems faced in the 90's when multicast received main attention from the research community remain unaddressed since so far. Indeed, routing protocol dependent multicast routing schemes (such as Distance Vector Multicast Routing Protocol and Multicast Open Shortest Path First) have been replaced by routing protocol independent routing schemes such as Protocol Independent Multicast (PIM) and Core Base Trees (CBT). However, overlaying multicast routing on top of unicast suffers from the same scaling limitations as current unicast routing with the addition of the level of indirection added by the multicast routing application. Thus, the number of entries in RTs (states) and their maintenance remains a major problem.

6.2 Problem Statement and Literature Review

6.2.1 Requirements

Today's Internet routing lays on poor scaling properties. As new routing paradigms are desired, the main challenge is to address new routing schemes meeting the following distinctive characteristics:

Scalability (memory) The RT size must scale better than $\Omega(nlog(n))$, i.e., a sub-linear RT size growth is desired (n is the number of nodes).

Quality (stretch) The stretch is a quality measure given the increase of the path length (cost) produced by the routing scheme compared to the minimum path length (cost). Such minimum cost, which is incompressible, is given by the shortest path routing (stretch = 1) in the unicast (p2p) routing case and by the Steiner tree routing [Hwang92] in the multicast (p2mp) case. We must find a bound length increase that does not grow with the network size.

Reliability Fast convergence upon topology changes is required. In such a way, the routing can be *updatefull*: route with non-local knowledge about network topology. In this case, information exchanges result from network topology and routing updates. These routing updates lead to the re-computation of RT entries that in turn may lead to convergence delays, instabilities and processing overhead. Thus, a small number of routing update messages (referred to as communication cost) is also required to maintain non-local knowledge about the network topology; or it can be *updateless*, which is the contrary. This neither requires the exchange of routing updates nor maintain non-local knowledge about the observed network topology, which should be achieved by other means that are being currently studied.

Topology independent routing A routing scheme can be either name-dependent or name-independent. The former means topology-aware addressing where the node addresses (or labels) encode some topological information useful for routing. Thus, labels cannot be arbitrary. The main drawback is if the topology changes, the node label has to change too (renaming). In turn, the later means topology-unaware addressing. Here the node addresses are assigned independently of the topology (arbitrary addressing space). Note that scalable name-independent schemes are highly desirable for Internet routing due to site multi-homing, mobility, etc..

Specialization The Internet topology is not a tree or a grid but approximates scale-free graphs class. The node degree (k) distribution follows a power-law distribution $P(k) \sim k^{-\gamma}$ with scaling index $\gamma = 2.254$. In other words, fewer nodes have large degree while large number of nodes have low degree. The routing scheme should take advantage of these properties.

6.2.2 Strategy

Nowadays, a router maintains independent unicast and multicast forwarding states in its RT regardless of the underlying forwarding paradigm. Both membership states are stored as entries in the RT, that is subsequently used to derive a FT. The latter determines the actual forwarding of an incoming packet to a router's outgoing interfaces. A multicast routing protocol enables routers to build a (logical) delivery tree between the sender(s) and receivers of a multicast group. In such a way, a Multicast RT includes the Multicast Routing Information Base (MRIB) and the multicast Tree Information Base (TIB). The MRIB is the topology table, typically derived from the unicast routing table, which carries multicast-specific topology information. The TIB is the collection of routing state created from the exchange of join/prune messages. This table stores the state of all multicast distribution trees at that node. On the other hand, a unicast RT includes the Unicast Routing Information Base (URIB) from which the FIB is derived.

The maintenance of these entries is one of the major problems that limits the scalability of any multicast deployment when dealing with bandwidth intensive multimedia applications as well as topologically scattered but large multicast groups. Unlike unicast routing, which "uses" a clever hierarchical label assignment that lead to significant reduction in its FT size (taking advantage from aggregation towards common destination), there is no natural aggregation in multicast forwarding states. Thus, the multicast states cannot be reduced easily - in particular for selective trees - and the scalability of the multicast routing protocol becomes a very critical issue. In addition, a node/router may take a long time to look up the forwarding state for each arriving packet when there are a large number of multicast group [Wong00]. Several research contributions proposed methods to reduce the multicast forwarding state through aggregation [Thaler00][Radoslavov99], tunneling [Blazevic99], and no-branching state elimination [Tian98][Yang08].

The problem becomes even more complex when considering cases for which the set of receiver nodes changes dynamically during the multicast content distribution session. In such a case, a node can join or leave a multicast session at any time which generally implies the addition (grafting) or removal (pruning) of a path from the multicast tree. The dynamic process of grafting and pruning membership nodes can lead to suboptimal configuration of the multicast trees over time. For this reason, readjustment mechanisms must be included to prevent too much divergence from the optimal distribution. Such mechanisms can be based on periodic readjustment, cost threshold policy or specific criteria.

Even if some of these techniques might be considered as acceptable, all of them rely on the current Internet principles limiting their scalability in the future. The following proposed concepts are totally new and try to improve such existing algorithms by i) achieving lower state consumption (memory - RT size) at small stretch (or low bandwidth consumption increment) and ii) cope efficiently with topology changes (dynamics).

First, we defy the current forwarding paradigms used in current traffic-engineered, packet-switched networks, by introducing the so-called AnyTraffic Labeled routing scheme. According to this routing scheme, both unicast and multicast traffic demands are forwarded sharing a single routing entity

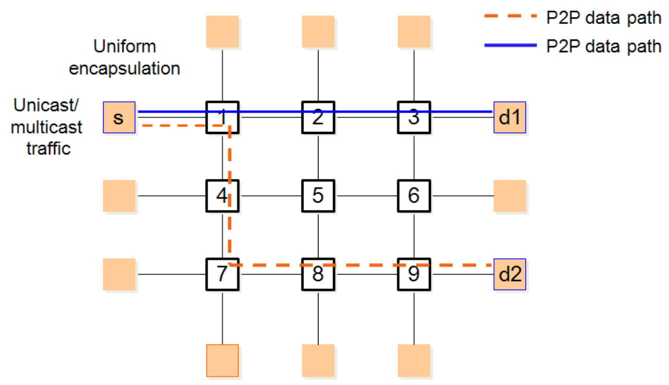


Figure 6.4: Approach 1 : P2P (unicast+multicast).

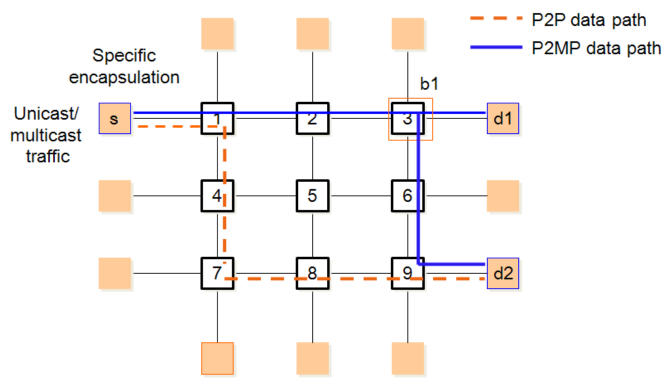


Figure 6.5: Approach 2 : P2P (unicast) + P2MP (multicast).

(i.e. tree) and thus saving on routing (i.e. forwarding) entries (memory). The main objective is to assess the reliability and feasibility of the concept in terms of stretch and RT size in order to be applied later to any other routing scheme. The proposed scheme is root-initiated as in source-specific multicast.

Second, we aim to construct, in polynomial time, a name-independent dynamic compact multicast routing scheme and later on a name-independent dynamic compact AnyTraffic routing scheme. In both cases, such routing schemes should minimize the stretch bound for Internet-like graphs while requiring at most o(n) bits per RT (n is the number of nodes of the network). In this case, the proposed schemes are leaf-initiated as in any-source multicast. A full study in terms of RT size, communication cost and stretch is performed in synthetic Internet environments.

6.2.3 AnyTraffic Routing

In current traffic-engineered, packet-switched networks, we have both unicast and multicast traffic demands competing for shared resources. In such a way, two different forwarding approaches are commonly assumed, namely:

1) Forwarding on a set of P2P switched paths to encapsulate either unicast or multicast traffic. Constraint-based shortest path algorithm is commonly used to compute the corresponding path across the network topology. This implies that both unicast and multicast traffic are carried over p2p data paths (referred to as network trunks) between each edge-node pair. It also implies that the multicast traffic must be replicated as many times as the number of edge nodes receiving (and processing) multicast traffic. This approach is clearly a suboptimal solution and entails high bandwidth consumption as well as high forwarding (state) consumption (see Fig.6.4).

2) Forwarding on a set of dedicated P2P paths for unicast traffic and dedicated P2MP paths for multicast traffic. The latter is the most referenced approach to forward multicast traffic and consists in setting up dedicated p2mp data paths in addition to the p2p data paths for unicast traffic. Steiner Tree heuristics (STH) [Hwang92] are commonly used to construct the minimum cost tree dedicated to multicast traffic. Edge nodes in this case have to provide for differentiated treatment of incoming native multicast (from p2mp data paths) vs. unicast traffic (from p2p data paths) in order to forward it in the outgoing p2mp and p2p data path respectively. The p2mp data paths can be either inclusive or selective. Inclusive implies that a single p2mp data path is setup for the entire set of multicast groups to a set of edge nodes. Note that the set of edge nodes may be greater than the number of edge nodes of each individual multicast group resulting thus in bandwidth waste. Selective p2mp implies that each multicast group is mapped into a dedicated p2mp data path, resulting thus in saving bandwidth at the expense of system resource needed for additional p2mp state maintenance. This approach also implies that dedicated p2p data paths must be provisioned for unicast traffic with their corresponding states in the FT (see Fig.6.5).

Therefore, the idea is to use only one type of entry for both traffic to save on routing entries. The problem is that applying any of the routing algorithms used by traditional approaches (i.e. 1) and 2)) to both type of traffic results in a suboptimal solutions. For instance, if we apply the Shortest Path (SP) algorithm, we would be underperforming in terms of multicast routing. On the other hand, if we apply Steiner tree (ST) algorithm, we would be underperforming in terms of unicast routing because the path to carry the unicast traffic - one of the leaves of the multicast tree - would be too long when compared to the shortest path, requiring considerable additional (unicast) bandwidth.

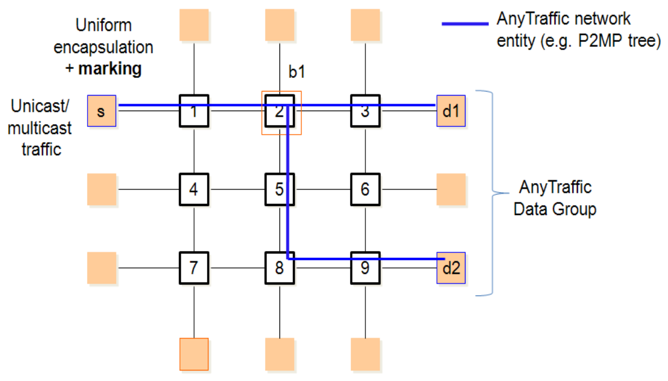

Uniform encapsulation + marking
Unicast/ multicast traffic
s
b1
1
2
3
4
5
6
7
8
9
d1
d2
AnyTraffic network entity (e.g. P2MP tree)
AnyTraffic Data Group

Figure 6.6: Approach 3 : AnyTraffic (unicast+multicast).

3) The concept of AnyTraffic data group is therefore introduced to define a group of destination nodes receiving unicast and multicast traffic over the same source-initiated network entity (see Fig.6.6 - Approach 3) with the goal of save on state consumption (i.e. forwarding entries). To this end, a specific heuristic algorithm is required, attempting to construct such single network entity per each AnyTraffic data group. This is a breakthrough approach compared to the literature work found in this topic.

The aim of the heuristic algorithm is to find, at the minimum-cost, a set of branch nodes that takes into account unicast and multicast traffic constraints. At these designated branch nodes, several p2p data path segments are appended to reach the destination nodes that belong to a given set of one or more AnyTraffic data groups. The branch node selection is done according to a given pruning condition in order to guarantee a low increase of bandwidth consumption and consequently the length of unicast path. In fact, the created network entity is a root-initiated p2mp tree which is signaled using the technique described in [RFC4875]. Note that this single scheme for AnyTraffic data simplifies the management of the p2mp tree when considering dynamic multicast sessions.

It is then expected that our heuristic places itself between the SP and the ST algorithms, achieving better overall performance to forward an offered load consisting of combined unicast and multicast traffic in packet-switched meshed networks, as shown in Chapter 7.

6.2.4 Compact Routing

A routing scheme is said to be compact if node names (labels) and header sizes scales logarithmically; if the RT size scales sub-linearly; and the stretch is bounded by a constant (independent of the network size). There is the unavoidable trade-off between stretch of routing algorithm and RT size it produces (and the messages exchanged (communication cost) if a dynamic scenario is considered).

The problem of designing routing schemes with small RT size has been introduced at the very beginning of the Internet, in the 1970's by Kleinroch and Kamoun in their seminal work "Hierarchical Routing for Large Networks; Performance Evaluation and Optimization, Computer Networks" [Kleinroch77], and the theoretical aspects of compact routing have been mainly developed in late 1980's by the work of Peleg and Upfall [Peleg89].

What we learn from these works, and from several subsequent ones, is that every weighted network with n nodes has a routing scheme with RT of approximately $n^{1/k}$ memory space per router such that the length of any route of the scheme is no more than $O(k)$ times the optimal length (i.e., the distance), where k is any integral parameter > 0. Recent results lead to some significant improvement on the stretch as produced by routing scheme, i.e., the factor $O(k)$, and on the capability of the scheme. [Thorup01] demonstrated for general weighted undirected networks that a name-dependent handshaking-based routing scheme that uses $O(n^{1/k})$ bits memory at each router has stretch $2k - 1$ (for every integer $k > 2$). The authors also demonstrated that without handshaking, the stretch of the routing scheme increases to $4k-5$. In name-dependent (or labeled) routing schemes, addresses (or labels) encode some topological information. As labels cannot be arbitrary, any topological change implies node address change (renaming).

Lots of attention has been captured by "name-independent" compact routing schemes. [Abraham08] presents the first name-independent compact unicast routing scheme for arbitrary undirected weighted graphs. The routing scheme has stretch 3, and requires poly-logarithmic-bit headers and

$O(n^{1/2})$ bits of routing information per node. When routing along its stretch 3 paths, each routing decision is performed in constant time. The fundamental result is that with $O(n^{1/2})$ bits of routing information per node, topology-dependent node labels does not improve the stretch factor compared to topology-independent naming of nodes. Indeed, in name-independent routing schemes, the addressing space is arbitrary and topology-unaware, i.e., independent of the topology, thus highly desirable for Internet routing system supporting dynamic terminal multi-homing, mobility, etc.

Recent schemes developed in [Abraham06] [Abraham08] can support "name-independence" with a stretch in $O(k)$, for $O(n^{1/k})$ memory space, where the hidden constant is about hundred. Recent studies have applied compact routing schemes on Internet-like topologies by taking advantage of their topological properties (network diameter growing logarithmically in the number of nodes and node degree distribution following power law). [Krioukov04] showed that the average performance of the stretch-3 compact routing scheme of [Thorup01] on Internet-like topologies is much better than its worst case, it achieves an average stretch = 1.1 (up to 70% of all pair-wise paths being stretch-1 shortest paths).

What does remain unsolved? Nevertheless, drawbacks resulting from the name-dependence of the routing scheme remain unaddressed and limit their applicability to static topologies (and thus inapplicable for dynamic and evolutive topologies such as the Internet). Application of the name-independent general scheme of [Abraham08] to Internet-like topologies has thus been trialed but leads to an average stretch of 1.5 (thus worse on average than its name-dependent counterpart).

Despite the amount of work dedicated to deal with properties of large-scale networks, most of the previous results achieved with compact routing schemes consider only static networks. Unfortunately, existing compact routing schemes neither handle node/link insertion/removal (characteristic of the network evolution) nor failures (intermittent or permanent).

Hence, compact routing schemes can not currently cope with network topology dynamics. In [Krioukov07], it is said that we need radically new ideas that would allow us to construct convergence-free, "updateless" routing, requiring no full view of the network topologies. We are at an apparent impasse as current static, name-dependent (i.e. topology aware) compact routing algorithm does not cope with the desired level of scalability to future Internet and name-independent compact routing algorithms do not perform better than name-dependent in Internet-like topologies so far. Moreover, as it concerns to multicast context, there is only one attempt so far and it is name-dependent and source-routed [Abraham09].

It is our objective threfore to contribute with a leaf-initiated, dynamic and name-independent compact multicast routing.

Chapter 7

AnyTraffic Labeled Routing

In this Chapter, we investigate a routing algorithm that computes paths along which combined unicast and multicast traffic can be forwarded altogether, i.e., over the same path. For this purpose, the concept of AnyTraffic group is introduced that defines a set of nodes capable to process both unicast and multicast traffic received from the same (AnyTraffic) tree. The resulting scheme is referred to as AnyTraffic routing.

This research work comprises the definition of a heuristic algorithm to accommodate the AnyTraffic group and to find the proper set of branch nodes of the tree. The algorithm supports dynamic changes of the leaf node set during multicast session lifetime by adapting the corresponding tree upon deterioration threshold detection. Studies are performed for both static and dynamic traffic scenarios to i) determine the dependencies of the algorithm (node degree, clustering coefficient and group size); and ii) evaluate its performance under dynamic conditions. Results show that the AnyTraffic algorithm can successfully handle dynamic requests while achieving considerable reduction of forwarding state consumption with small increase in bandwidth utilization compared to the Steiner Tree algorithm.

7.1 AnyTraffic Routing Concept

In this section, we propose a traffic routing approach, whose computed paths enable forwarding of both unicast and multicast traffic together, i.e., over the same path. For this purpose, the concept of AnyTraffic group is introduced that defines a set of nodes capable to process both unicast and multicast traffic received from the same distribution tree, the AnyTraffic tree. The resulting routing scheme is referred to as AnyTraffic routing. This Chapter presents a heuristic algorithm to accommodate the AnyTraffic group and to find the proper set of branch nodes of the AnyTraffic tree. It also provides for a performance evaluation of the proposed scheme against two commonly used approaches.

Introducing an AnyTraffic distribution tree to a group aims at reducing the total number of forwarding states by maintaining (as much as possible) a single path for both unicast and multicast traffic forwarding altogether. In other terms, a single state allows for both unicast and multicast traffic forwarding. The idea is to perform label-based forwarding (where labels encode topological information) using a single forwarding table entry for both unicast and multicast labeled traffic directed toward the same "label". Network nodes are named with destination labels that encode topological information. These labels are used in the forwarding decision process: each datagram carries the chosen destination in its header. At intermediate nodes, a discriminator (set at network ingress node) allows selecting either a unicast (one-to-one) or a multicast (one-to-many) forwarding entry associated to the same label entry.

Differentiation between incoming unicast and multicast traffic is simply performed by means of a (single-bit) discriminator. Thus, the proposed routing scheme is applicable to any label-based forwarding technology as long as the following conditions are met: i) capability to distinguish multicast from unicast traffic by inspecting other header information than the destination address (e.g. label flag to discriminate between unicast and multicast traffic following the same path); and ii) de-multiplexing of traffic at destination nodes relies on the information encoded as part of other header information not processed by each network node. This scheme can be seen as a unification of the locator/identifier split concept where the locator value space names topological end-points that are able to terminate any traffic and the identifier value space allows distinction (at the edges) between unicast and multicast traffic. Ingress edge nodes upon multicast traffic identification tag this traffic as part of the label. Based on this indication, branch nodes along the AnyTraffic tree replicate the multicast traffic onto outgoing interfaces towards edge nodes registered for the corresponding multicast group(s). On the other hand, the unicast traffic directed to these edge nodes is not replicated at branch nodes but follows "as short as possible" paths. The salient feature of the proposed scheme is that the multicast traffic does not require any additional forwarding entry on intermediate network node to reach the topological location where the traffic is then natively processed.

The aim of the proposed routing scheme is to achieve better system resource consumption (for state maintenance) while limiting the network resource consumption, i.e., mitigate the state vs. bandwidth resource tradeoff by increasing the "common path" stretch. The proposed approach keeps the forwarding state maintenance overhead as low as possible while avoiding bandwidth waste by i) avoiding replication of multicast traffic at branch nodes, and ii) keeping unicast traffic forwarding over "as short as possible" paths. To meet this objective combined with the decrease in hop count of p2p paths, a deficit factor and an adaptive threshold function for the selection of the branch nodes are specified to decide where to separate the unicast from the multicast forwarding path (i.e., the placement of a branch node). This algorithm is also designed so as to efficiently operate in a dynamic environment where receivers are joining and releasing multicast sessions. Beside the reduction of the number of states, the AnyTraffic routing scheme can also handle more efficiently join and leave requests. Here, as both types of requests may deteriorate the system resource performance compared to the optimal case, a readjustment mechanism is designed so as to accommodate actual receivers' dynamics by adapting the multicast tree. Finally, resulting from the type of routing information it processes, the proposed routing scheme applies typically within network partitions (intra-domain).

7.2 AnyTraffic Routing Algorithm

Consider a network modeled by a directed graph $G = (N, L)$, where N represents the finite set of nodes, and L represents the finite set of links. Let $s, d \in N$ denote a source and a destination node, respectively. Each link $l \in L$ might have an associated capacity $b(l)$, and cost $c(l)$. In section 7.3, we present the method we use to calculate the cost $c(l)$ of link $l \in L$.

Let $p_{i,j}$ and $p_{i,k,j}$ both denote a path from node i to node j, where k is an intermediate node with $i \neq j \neq k$. Let $T_{s,M} = (N_T, L_T)$ be a connected sub-graph without cycles (i.e., a tree) of G, source-initiated at s, and with the set of destination nodes $M \subseteq N_T \setminus \{s\}, M \neq \emptyset$. Hereafter, M is referred to as the AnyTraffic group, and Ts, M as the AnyTraffic tree.

7.2.1 Static AnyTraffic Heuristic Algorithm

Let $\phi_{s,M}$ denote a traffic request between source s and an AnyTraffic group M, where $M \subseteq N \setminus \{s\}, M \neq \varnothing$. If $|M| = 1$, $\phi_{s,M}$ is a request for unicast traffic, i.e., a P2P data path to unicast (one-to-one) traffic forwarding. Otherwise, it is a multicast request with multiple destination nodes $d_1, ..., d_{|M|}$, i.e., a P2MP data path to multicast (one-to-many) traffic forwarding.

The objective of the AnyTraffic routing algorithm is to construct a graph $T_{s,M}$ for a given source s and AnyTraffic group M, such that $T_{s,M}$ supports both unicast and multicast traffic requests. The graph $T_{s,M}$ is constructed by successive selection of branch node, $n* \in N$. At a given source node s, the processing of the request $\phi_{s,M}$ depends on its nature, i.e., it is either a unicast or multicast traffic request. We have the following alternatives:

i) if a multicast traffic request $\phi_{s,M}$ arrives and an AnyTraffic tree $T_{s,M}$ is available, then the request is supported by $T_{s,M}$; otherwise, the AnyTraffic routing algorithm is executed (see Section 7.2.1) to establish a new AnyTraffic tree;

ii) if a unicast traffic $\phi_{s,d}$ request arrives, three situations can occur: (a) $d \in M$ and $T_{s,M}$ (with $|M| > 1$) is available and $\phi_{s,M}$ is supported by $T_{s,M}$; (b) $d \in M$ but $T_{s,M}$ is not yet created and thus a shortest path must be setup; or (c) $d \notin M$ and thus a shortest path must be setup.

The AnyTraffic routing algorithm comprises two phases, namely, the initialization and tree computation phase. The

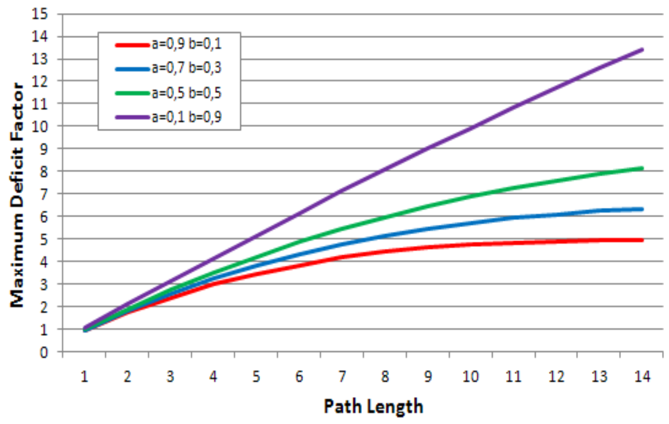


Figure 7.1: Maximum Deficit Factor function ($\Delta^{s,d}_{max}$).

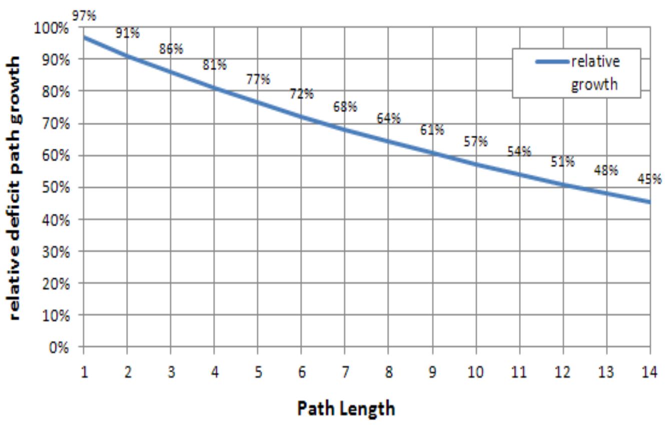


Figure 7.2: Relative Growth of the Path Deficit (a=0.7, b=0.3).

initialization phase assigns initial values to algorithm attributes. The tree computation phase specifies the set of iterative processes that occur during the actual execution of the algorithm.

1) Algorithm Initialization Phase

The initialization phase of the algorithm consists in defining attributes and assigning initial values to them. Since these attributes depend only on the network topology, this phase is performed once (off-line) and their values can be stored in the memory of network elements.

Let $x_{i,j}$ denote the cost of the shortest path from node i to node j, $i \neq j$, as computed by the Dijkstra algorithm on the (positive integer) link cost $c(l), l \in L$. The hop count is used as tie-breaker. Methods for computing the cost c(l) of each link $l \in L$ can be found in section 7.3. Accordingly, $x_{s,d}, d \in M$, denotes the cost of the shortest path from the source s to the destination $d \in M$. Among all path costs, we can find $c_{\max} = \max\{x_{i,j} : i, j \in N, i \neq j\}$, which corresponds to the shortest path of maximal cost in the network. Let the function $F(x) : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ be a function defined as follows:

$$\Psi\left(x\right) = x\left(1 + e^{-\frac{f(x)}{d_{\max}}}\right), \tag{7.1}$$

where function $f(x) : \mathbb{R}^+ \rightarrow \mathbb{R}$ is defined as

$$f\left(x\right) = \alpha x - \beta, \tag{7.2}$$

The function $F\left(x\right)$ specifies the threshold for the maximum cost of an alternative path to the path of cost x. In particular, $F\left(x_{s,d}\right)$ limits the acceptable cost deviation of an alternative path $p_{s,d}$, $d \in M$, from the path given by the Dijkstra algorithm (optimal solution unicast (one-to-one) traffic forwarding). This admissible cost deviation depends on a linear function $f(x)$ and d_{max}. The parameters α,$\beta \in [0, 1[$ of $f(x)$ define the shape of the threshold function. The maximum growth is achieved at α approaching zero and β approaching one. In this study, after performing a number of experiments, we believe that $\alpha = 0.7$ and $\beta = 0.3$, are good values to initiate the algorithm so that the state consumption would be the lowest possible. High values for α and β would make the function too restrictive, allowing only for low path stretch. On the other hand, low values would result into too high path stretch. In Fig. , one can see the different curves of the maximum deficit function by changing the a and b parameters. In turn, the figure on the right (7.2) illustrates the percentage of path growth allowed in function of the original path length (i.e. the shortest path).

Function $F(x)$ is applied (indirectly) as a decision criterion by the AnyTraffic routing algorithm in order to decide whether an alternative path between two nodes is acceptable or not. Closer to the tree-root/source, selection as part of the AnyTraffic subgraph of alternative paths (deviating from the shortest path) among all possible alternative paths is desirable up to a certain limit for unicast traffic. Indeed, for link cost metrics as defined in Section 7.3, increasing the path cost for such traffic results in additional state consumption that increases inversely to the "distance" from the tree-root: branching unicast traffic closer to the source consumes more states (than branching closer to one of the tree-leaves) at the expense of a slightly higher bandwidth consumption. The initial linear-growth for lower x is smoothly vanished into a curvature as x becomes larger, given the former idea. Having defined Ψ, we can calculate a *maximum deficit factor* $\Delta_{\max}^{s,d}$ for each each shortest path $p_{s,d}$, given by:

$$\Delta_{\max}^{s,d} = F(x_{s,d}) - x_{s,d} = x_{s,d}e^{-\frac{f\left(x_{s,d}\right)}{d_{\max}}}. \qquad (7.3)$$

This factor determines the acceptable cost increment(s) of an alternative path against the shortest path, i.e., it quantifies the tolerable cost deviation when forwarding both multicast and unicast traffic on that path without incurring too much damage compared to unicast traffic forwarding along the shortest path. Being only dependent on the topology, $x_{s,d}$, c_max, $f(x_{s,d})$, and $\Delta_{\max}^{s,d}$ can be computed during the initialization phase (i.e. off-line) for any pair $(s,d) \in N$ (values remain constant during the tree computation phase).

2) Tree Computation Phase

This phase of the algorithm consists in the tree computation itself, which is a progressive and iterative process. In order to make easier the understanding of the following description an illustrative example is depicted in Fig.**??**. A more detailed explanation of such example is presented in Subsection C.

Let's define a leaf as the tuple $\omega_{v,\Lambda} = \{v, \Lambda\}$, where $v \in N$ is a leaf seed and $\Lambda \subseteq M$ is a subset of the AnyTraffic group. We define Ω as the set of leaves remaining to be processed. At the beginning, this set comprises only the initial leaf, $\Omega = \{\omega_{s,M}\}$, where s is the seed root from where the computation is initiated and which comprises all destination nodes M. We also define the initial graph $T_{s,M} = (\{s\}, \varnothing)$.

The algorithm terminates when there is no leaves left in Ω and all destinations $d \in M$ can be reached from s with the graph $T_{s,M}$. At each iteration step, an arbitrary leaf $\omega_{v,\Lambda}$ is pull out from Ω and the algorithm searches for a branch node $n^* \in N$, to be included in $T_{s,M}$, such that source s is connected through n^* to a subset of nodes comprised in $\Lambda^* \in \Lambda$. For this leaf $\omega_{v,\Lambda} \in \Omega$, a set of candidate branch nodes A_ω is found. The set A_ω is restricted to unvisited nodes in previous iterations that are adjacent to v and have a node degree equal or greater than three, i.e., the nodes that have at least two outgoing links, apart from the outgoing link to node v. In case the node degree of an adjacent node a is two, the first node with node degree equal or greater than three and laying on a path going from v through a is included into A_ω. For instance, at the example depicted in Fig.7.3a), $A_\omega = \{1, 2, 3\}$.

At each candidate branch node $n \in A_\omega$ being evaluated (i.e. one of the adjacent nodes of v), one alternative paths $p_{v,d}$ per destination $d \in \Lambda$, starting at v but forced to pass through n, is computed. A pruning condition determines if the set of alternative paths from v to each $d \in \Lambda$ and passing through n could be accepted. Indeed, each alternative path $p_{v,n,d}$ may introduce higher cost $(x_{v,n}+x_{n,d})$ when compared to the cost $x_{v,d}$ of the previous path $p_{v,d}$ (in the first iteration this is the shortest path). Therefore, a local deficit $\Delta_{local}^{v,n,d}$ is computed for each $d \in \Lambda$ by means of:

$$\Delta_{local}^{v,n,d} = (x_{v,n} + x_{n,d}) - x_{v,d}. \qquad (7.4)$$

For each $d \in \Lambda$, a cumulative path deficit $\Delta_{path}^{s,d}$, sums up, at node n, the local deficits produced by the alternative path $p_{s,d}$ passing by already accepted branch nodes $n_0 (= s), n_1, , n_u$ of $T_{s,M}$ and the candidate branch node $n_{u+1} (= n)$:

$$\Delta_{path}^{s,d} = \sum\nolimits_{i=0}^{u-1} \Delta_{local}^{i,j+1,d} = \sum\nolimits_{i=0}^{u-1} (x_{i,i+1}) + x_{n,d} - x_{s,d}, \qquad (7.5)$$

Then, and still for each $d \in \Lambda$, a comparison between the cumulative path deficit (7.5) and the maximum Deficit Factor (7.3) is performed. If the maximum deficit constraint is verified

$$\Delta_{path}^{s,d} <= \Delta_{\max}^{s,d} \qquad (7.6)$$

, i.e., if the cumulative deficit of an alternative path $p_{s,d}$ does not exceed the maximum deficit, the alternative path can be accepted. Otherwise, the algorithm removes node d from $\omega_{v,\Lambda}$ and creates a new leaf for further optimization.

When all candidate branch nodes have been evaluated, branch node selection can be performed by running the pruning condition for each $d \in \Lambda$. The decision is taken by considering the minimum total deficit among all candidate branch nodes $n \in A_\omega$. However, to reach decision fairness, considering the deficit based on the cost metric only is insufficient. Hence, we further ponder the deficit of each candidate branch node n by

i) summing a fraction γ of the local deficit (7.4) to a fraction $(1-\gamma)$ of a local deficit $\Delta_{local}^{v,n,d*}$ defined as (7.4) but using the hop count instead of the cost metric (in order that more hops represents more state records at the nodes);

ii) multiplying by a factor σ the ratio r, defined as the number of alternative paths meeting the pruning condition divided by the total number of paths that can reach all destinations, i.e., $|\Lambda|$), via $n \in A_\omega$. The parameter γ governs the selection

trade-off between longer but lower cost paths among the successful alternative paths meeting the pruning condition (high γ values), and shorter paths thus, lower probability to aggregate paths (low γ values). The parameter σ is a weight factor that favors candidate branch nodes with a higher number of successful alternative paths. After performing a number of experiments, we select $\gamma = 0,5$ and $\sigma = 2$. However, this variables can be tuned in function of the bandwidth and state consumption objectives. For each candidate branch node, a *candidate deficit* $\Delta^{n,\omega}_{candidate}$ is computed as:

$$\Delta^{n,\omega}_{candidate} = \sum[\gamma\Delta^{v,n,d}_{local} + (1-\gamma)\Delta^{v,n,d*}_{local}] - \sigma P_j \quad (7.7)$$

The candidate branch node n with the lowest deficit is se-

lected as a branch node $n^* = \min\{\Delta^{n,\omega}_{candidate} : n \in A_\omega\}$. Accordingly, $T_{s,M}$ is updated with all links and nodes that lay on the path from v, the seed of the currently processed leaf $\omega_{v,\Lambda}$ to n^*. Note that the resulting graph is different from that given by the conventional Steiner tree algorithm.

Then, two new leaves may be created, $\omega_1 = \{n^*, \Lambda_{n^*}\}$ (leaf with the subset of destination nodes ?n* that accepted n* as branch node), and $\omega_2 = \{v, \Lambda \setminus \Lambda_{n^*}\}$ (leaf with the destination nodes removed by the pruning condition). Leaves ω_1 and ω_2 are conditionally added to the set Ω for further processing, respectively, if $|\Lambda_{n^*}| > 1$ and $|\Lambda \setminus \Lambda_{n^*}| > 1$. If either $|\Lambda_{n^*}| = 1$ or $|\Lambda \setminus \Lambda_{n^*}| = 1$,, $T_{s,M}$ is updated with all links and nodes along the shortest path, respectively, n^* to $d \in \Lambda_{n^*}$ and from v to $d \in \Lambda \setminus \Lambda_{n^*}$.

Branch node n^* is excluded from the set of adjacencies of v, i.e., $A_{\omega_2} = A_\omega \setminus \{n^*\}$. Branch selection is repeated for each leaf left in Ω. The pseudo-code of the AnyTraffic routing algorithm is given below.

Algorithm 4 AnyTraffic Heuristic Algorithm.

```
Require: $G = (N, L)$,$c(l)$,$s$,$M$
Ensure: $T_{s,M} = (N^*, L^*), N^*?N, L^*?L$
1:
2: INIT{
3: $\omega_{s,M} = s, M$
4: $\Omega = \omega_{s,M}$
5: $p_{s,d} \leftarrow SP(s,d) \forall d \in M$
6: $\Delta_{max}(p_{s,d}) \forall d \in M$
7: }
8:
9: ANYTRAFFIC_HEURISTIC{
10: while $|\Omega| \neq \emptyset$ do
11:     $\Delta_{candidate} = 0$
12:     $\omega_{v,\Lambda} \leftarrow \Omega, \Omega = \Omega\{\omega_{v,\Lambda}\}$
13:     $A_w = Adjacencynodes(v)$
14:     for $n \in \Lambda_\omega$ do
15:         $\Lambda_n \leftarrow$ Pruning Policy($n, \Lambda$)
16:         $\Delta^{n,\omega}_{candidate} \leftarrow Pdeficit$
17:     end for
18:     $n* = min\{\Delta^{n,\omega}_{candidate} : n \in A_w\}$
19:     $\Omega = \Omega \cup \{w_1, w_2\}$
20:     $T_{HEUR} = T_{HEUR} \cup p_{v,n*}$
21: end while
22: }
23:
24: Function Pruning Policy($n, \Lambda$){
25: for $d \in \Lambda$ do
26:     $p_{n,d} \leftarrow SP(n,d)$
27:     $\Delta_{path}(d)_k = \Delta_{path}(d)_{k-1} + \Delta_{local}(d)_k$
28:     if $\Delta_{path}(d) > \Delta_{max}$ then
29:         $\Lambda_n = \Lambda_n \setminus \{d\}$
30:     end if
31: end for
32: return $\Lambda_n$
33: }
```

4) Complexity Analysis

The complexity of this algorithm is $O(|M| \cdot A \cdot H)$, where $|M|$ is the size of the AnyTraffic group, A is the maximum node degree, and H is the hop distance between the source and the most distant destination node. This bound comes from the fact that at each hop towards the destination all adjacent nodes are checked as candidate branch node for the destination nodes belonging to M. In a regular connected network ($A << |N|$), the complexity is low and it may be further reduced by limiting the set of adjacent nodes that are within a given perimeter with respect to the next node along the shortest path to each destination node. Results achieved by applying this method show no performance degradation while significant reduction of running time.

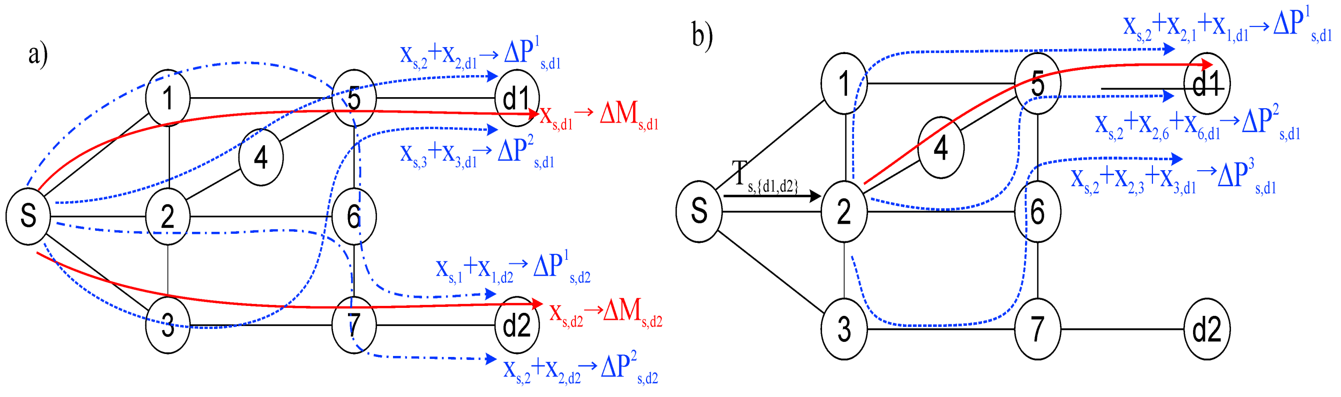

a)
b)
S
1
2
3
4
5
6
7
d1
d2
$x_{s,2}+x_{2,d1} \rightarrow \Delta P^1_{s,d1}$
$x_{s,d1} \rightarrow \Delta M_{s,d1}$
$x_{s,3}+x_{3,d1} \rightarrow \Delta P^2_{s,d1}$
$x_{s,1}+x_{1,d2} \rightarrow \Delta P^1_{s,d2}$
$x_{s,d2} \rightarrow \Delta M_{s,d2}$
$x_{s,2}+x_{2,d2} \rightarrow \Delta P^2_{s,d2}$
$T_{s,\{d1,d2\}}$
$x_{s,2}+x_{2,1}+x_{1,d1} \rightarrow \Delta P^1_{s,d1}$
$x_{s,2}+x_{2,6}+x_{6,d1} \rightarrow \Delta P^2_{s,d1}$
$x_{s,2}+x_{2,3}+x_{3,d1} \rightarrow \Delta P^3_{s,d1}$

Figure 7.3: Two consecutive steps of the branch node evaluation mechanism. a) initial leaf $\omega_{s,M}$, $M = \{d1, d2, d3\}$; first step of the algorithm; b) if previous pruning condition is satisfied and node 2 is selected as branch node, the process is repeat from the node 2 which is now the leaf seed of the leaf $\omega_{v,\Lambda}$, $\Lambda = \{d1, d2, d3\}$.

7.2.2 Execution Example

Figure 7.3 shows two consecutive steps of the branch node evaluation mechanism. We consider as initial leaf $\omega_{s,M}$, where s is the node processing the incoming requests for both unicast and multicast traffic, and the AnyTraffic group $M = d1, d2$. Fig.7.3a represents the evaluation of the set A_w of adjacent branch nodes for destination $d1$. The set $A = 1, 2, 3$ corresponds to the adjacent nodes of s with a node degree equal or higher than three. The path through node 1 is the shortest path and its cost $x_{s,d1}$ gives the value of the maximum deficit factor $\Delta M_{s,d1}$. For each of the remaining adjacent nodes, an alternative path is computed. The alternative path through node 2 and the one through node 3 may introduce higher cost with respect to the shortest path; these costs define the cumulative path deficit factor $\Delta P^1_{s,d1}$ and $\Delta P^2_{s,d1}$ respectively. At this step, an adjacent node is accepted as candidate branch node if the pruning condition is verified; for example node 2 is accepted if the cost of its alternative path $\Delta P^1_{s,d1}$ is lower than $\Delta M_{s,d1}$. The same evaluation is then repeated for each remaining destination node. Fig.7.3a also depicts evaluation for destination node $d2$. Once all adjacent nodes have been evaluated through the pruning condition, branch node selection is performed. For each accepted candidate branch node, the candidate deficit factor $\Delta C_n, \omega$ is computed. The node with the lowest value is selected as branch node. Assuming that the candidate branch node 2 is selected which corresponds to this situation depicted in Fig.7.3b, nodes s and 2 are added to $T_{s,M}$. The leaf seed is now $\omega_{2,M}$, the set of adjacent nodes of node 2 is $A = 1, 3, 4, 6$, and three alternative paths are considered (the path through node 4 was already accepted from the previous step). The same process described in the previous iteration is then repeated.

7.2.3 Dynamic Anytraffic Heuristic Algorithm

Let $\phi_{s,d}$ and $\varphi_{s,d}$ denote respectively, a join and a leave request between source s and destination d. Graph $T_{s,M} = (N_T, L_T)$ maintenance by the AnyTraffic algorithm under dynamic traffic requests conditions consists in appending node $d \in N_T$ to the graph $T_{s,M}$ when a join request $\phi_{s,d}$ arrives, and releasing node $d \in M$ from $T_{s,M}$ when a leave request $\varphi_{s,d}$ arrives. Graph $T_{s,M}$ is built up by iterative selection of branch nodes $n \in N_T$.

1) Join request

As regards unicast traffic, the join requests are forwarded over either i) a shortest path if the receiver is not a member of any AnyTraffic group or ii) an existing AnyTraffic tree.

As regards multicast traffic, two other situations can occur i) initiate a new AnyTraffic tree; ii) join an existing AnyTraffic tree at one of its branch node.

In any case, if an existing P2P path, rooted at the same source node, is up for such receiver, it is aggregated to the established AnyTraffic tree (saving on state consumption).

A possible approach for a receiver to join an existing tree would be to re-compute the entire tree as if a new group request would arrive (using the static algorithm). Such computation is optimal for a given receivers group and can thus be considered as an upper bound on the algorithm performance. The disadvantage is the need to re-establish the entire tree each time a join request is received.

Algorithm 5 P2MP Join Request Subroutine

```
Require: $G = (N, L)$,$s$,$M$,$d$
Ensure: $p_{n_b,d}$ of $T_{s,M}$
 1: Set $C \leftarrow BreadthFirstSearch(d, N_T)$
 2: if $C \neq \{\emptyset\}$ then
 3:    for $n_b \in C$ do
 4:       $p_{n_b,d} \leftarrow computeSP(n_b, d)$
 5:       $p_{s,n_b,d} = p_{s,n_b} + p_{n_b,d}$
 6:       Compute $\Delta^{(s,d)}_{path}$
 7:       if $b(l) > rate, p_{s,n_b,d}$ then
 8:          if $\Delta^{(s,d)}_{path} < \Delta^{(s,d)}_{max}$ then
 9:             save $min(p_{s,n_b,d})$
10:          end if
11:       end if
12:    end for
13:    if mindeficitPath $\neq \{A\}$ then
14:       update $T_{s,M}$
15:    else
16:       $C \leftarrow BreadthFirstSearch(d, N_T, C)$
17:    end if
18: else
19:    $\phi_{s,d}$ rejected
20: end if
```

To overcome this situation, we propose an extended mechanism to update the tree without re-computing the entire tree. Using this approach, deviation from the best case (as given by the static algorithm) is controlled by the mechanism given on Section 7.2.3. Let's assume a new receiver node $d \in N_T$ attempts to join the AnyTraffic group M supported by the tree $T_{s,M}$. Updating the tree "on-the-fly" lies in joining the closest node of the tree under the maximum deficit constraint. The algorithm performs the following steps:

Step 1) A Breadth-First Search algorithm is executed to find a set of candidate branch nodes $A_\omega \in N_T$ with the shortest hop count to node d;

Step 2) For each node $n \in A_\Lambda$, find the shortest path $p_{n,d}$ to the receiver. For each path $p_{s,n,d}$ obtained by splicing path $p_{s,n}$, which is determined by the tree $T_{s,M}$, and $p_{n,d}$, calculate its deficit $\Delta P_{s,d}$. Then, among all these paths, select the path with the smallest deficit $\Delta P_{s,d}$, such that it satisfies the constraint $\Delta P_{s,d} <= \Delta M_{s,d}$;

Step 3) If such path $p_{s,d}$ is not found, step 1 is repeated by excluding the already processed nodes from the set of candidate nodes A_ω;

Step 4) Once these steps are completed, as the receiver may still have unicast connectivity rooted at the source node of the AnyTraffic tree, the corresponding forwarding table entries are removed and traffic is forwarded over the tree.

In Procedure 5, we present a pseudo-code of the P2MP data path join request subroutine of the algorithm. An alternative to Breadth First Search would be to restrict the set C to those tree's nodes already acting as branch nodes. However, this alternative is outside the scope of this paper.

2) Leave/Prune request

When a multicast traffic receiver wants to leave an AnyTraffic tree, the simplest operation consists of pruning the leaves of the tree which are not used by any other remaining receivers. This leads to two cases: the leaf node could be either a destination node or an intermediate node of the tree. Let's assume a receiver b Ts,M, attempts to leave the AnyTraffic group M. The following operations must be performed:

1) if node b is a leaf node, then the path from branch node n to node b must be pruned;

2) if node b is an intermediate node, the entry for this node must be removed from forwarding table. The forwarding state is not removed because some receivers are still active along the path.

In both cases, a check is performed to verify if any P2P path is up for the leaving receiver. In case the receiver is a member of other AnyTraffic group and the releasing branch node crosses one of the corresponding AnyTraffic tree, unicast traffic may be redirected over one of the existing trees. Concerning unicast release requests, if the receiver is a member of a multicast session, then the request does not result into any state update if the corresponding path shares the same forwarding state with an AnyTraffic tree.

3) Deterioration Control

In a dynamic environment, after a certain period, join and leave requests deteriorate the entire AnyTraffic trees, due to the unpredictability of events. Hence, from time to time, the entire tree should be re-computed to re-establish the equilibrium.

The process of locally re-adapting the tree pursues the detection of deterioration, i.e., deviation of the re-adapted tree from the best case. The deviation is computed by the formula:

$$w * D_s + (1 - w) * D_b \tag{7.8}$$

where D_b and D_s accounts respectively for the bandwidth and state consumption differences. To penalize higher state consumption, D_b and D_s are weighted asymmetrically by the weight factor w. Higher w values imply re-computation when excessive states are used compared to the best case; lower values mean that deviation in terms of consumed bandwidth leads to re-computation.

The pre-determined deterioration threshold value is used to decide to either continue with the on-the-fly adapted tree (up to a certain deviation from the best case) or instead shift to a full tree re-computation. A high threshold value means less re-computation; on the contrary, a low value means stricter control, avoiding bandwidth and state consumption at the expense of more computation.

4) Complexity Analysis

The time complexity is $O(A^{H'} \cdot |N_T|)$, where A is the maximum node degree, H' is the hop distance between the node to be attached to the tree and the most distant node of the tree, and $|NT|$ is the number of nodes of the tree. Indeed, the number of iterations the algorithm performs depends mainly on the candidate branch node search, implemented by the Breadth-First Search algorithm. At each hop, the algorithm explores adjacent nodes looking for candidate branch nodes. Then, for each candidate node, the constraint compliance procedure is applied. In the worst case, all nodes have to be checked. The time complexity can be approximated by $O(|N_T|)$ since any node of the graph G is visited at most only once.

7.3 Simulation Environment

Experiments are executed on an ad-hoc simulator specifically developed to accommodate the proposed algorithms (to the dynamic scenario, it was extended to operate on an event-basis). In order to determine the topological dependencies and assess the robustness of the proposed routing schemes, different network topologies are considered: 37-nodes Cost266 [10]/Rand37, 50-nodes Germany50 [Inkret03]/Rand50 and 100-nodes Rand100 networks. Rand topologies are generated from a random sequence of node degrees [Kim08]. The differentiating properties of these topologies consist in different node degrees and clustering coefficient (c.c.) ranging in the interval [0,1], besides the number of nodes and links. The 37-nodes network topologies, the c.c. is 0.0 to the Cost266 and 0.2489 to the Rand37. The 50-nodes networks topologies, has a c.c. of 0.19 to the German50 and 0.2752 to the Rand50.

Each network node is an ingress-egress node generating, in a bound and discrete process, 150 (200) traffic requests for the static (dynamic) scenario. Both unicast and multicast traffic are generated within a range of two discrete traffic classes, namely class 1 - MPEG-4 standard definition (SD) - of 2 Mbps and class 2 - MPEG-4 high definition (HD) - of 8 Mbps. Different unicast and multicast traffic percentages are considered, namely 50%-50%, 75%-25% and 95%-5% respectively.

For each multicast session, the size of the destination node set |M| ranges between $log_2(N)$ and $[log_2(N)]^2$, where N represents the number of nodes. After performing a number of experiments, we set $\alpha = 0.7$ and $\beta = 0.3$ in the function $F(x)$ (Eq.7.1), and selected the values $\gamma = 0.5$ and $\sigma = 2$ for the candidate deficit $\Delta C_{n,\omega}$ (Eq.7.7). State consumption measures the number of forwarding states required to accommodate each traffic ratio.

We also define the consumption gain as the percentage of performance gain in terms of either bandwidth or state consumption, attained when AnyTraffic routing is compared either to AP1 or AP2. A negative gain means a loss for the AnyTraffic algorithm. The Consumption Gain is defined as follows, where the index $x = 1$ refers to the approach AP1 and index $x = 2$ to the AP2.

$$ConsumptionGain[\%] = \frac{APx - AP3}{APx} * 100 \tag{7.9}$$

In order to settle the bounds of the algorithm, we simulate a best case (in static scenario) where all multicast requests are processed first, creating the network entities (e.g. P2MP trees) for the AnyTraffic data groups and then process the unicast requests looking for the minimum cost path among all those trees. We also processed the requests in a non best case order where the multicast and unicast requests are interchanged and such unicast/P2P traffic requests are not always served by a single tree (in dynamic scenario).

Link Cost Function Below we present the method we use to compute the cost $c(l) \in Z_+$ of each link $l \in L$. Firstly, we find a uniform segmentation of the maximum distance link $L_{\max}$ into $Q_{\max}$ intervals, where $Q_{\max}$ is the maximal node degree in the network. For instance, for the maximum link distance of $100km$ and the maximal node degree of 5 we have the following intervals: $]0, 20]$, $]20, 40]$, etc. Then, for each network link l we find the corresponding interval number i such that $\frac{(i-1)L_{\max}}{Q_{\max}} < d(l) \leq \frac{iL_{\max}}{Q_{\max}}$. Therefore, the link cost is calculated as:

$$c(l) = \left\lceil \frac{Q_{\max} + (i-1)}{Q(l)} \right\rceil, \qquad (7.10)$$

where $Q(l)$ is the degree of the origin node of link l. Such assigned costs give preference to shorter links and their calculation involves a smoothing factor inversely proportional to the node degree. Note that this link cost calculation is possible only if the link distance metric, $d(l)$, is known.

7.4 Results and Analysis

Simulations are performed to estimate the performance of the AnyTraffic algorithm in terms of bandwidth and state consumption, under the following scenarios: i) non-blocking static traffic; ii) dynamic traffic with limited capacity per link.

Two reference approaches are used for comparison purpose: approach 1 (AP1) that forwards both unicast and multicast traffic along "as short as possible" paths (shortest path routing); and approach (AP2) that makes use of shortest path routing for unicast traffic and replication of multicast traffic at branch points of a tree as computed by the minimum-cost path algorithm, a Steiner Tree Heuristic (STH) [Hwang92].

For the dynamic scenario, the latter has been extended with a Greedy tree-based algorithm [Fei02] to process dynamic requests. The AP1 is not considered for comparison in the dynamic case due to its poor performance observed in the static case.

7.4.1 Static Traffic Scenario

The simulation steps consist in i) creating the network entities (trees) for the AnyTraffic groups, and then ii) processing the unicast requests looking for the minimum cost path among the created trees (best case).

Analysis

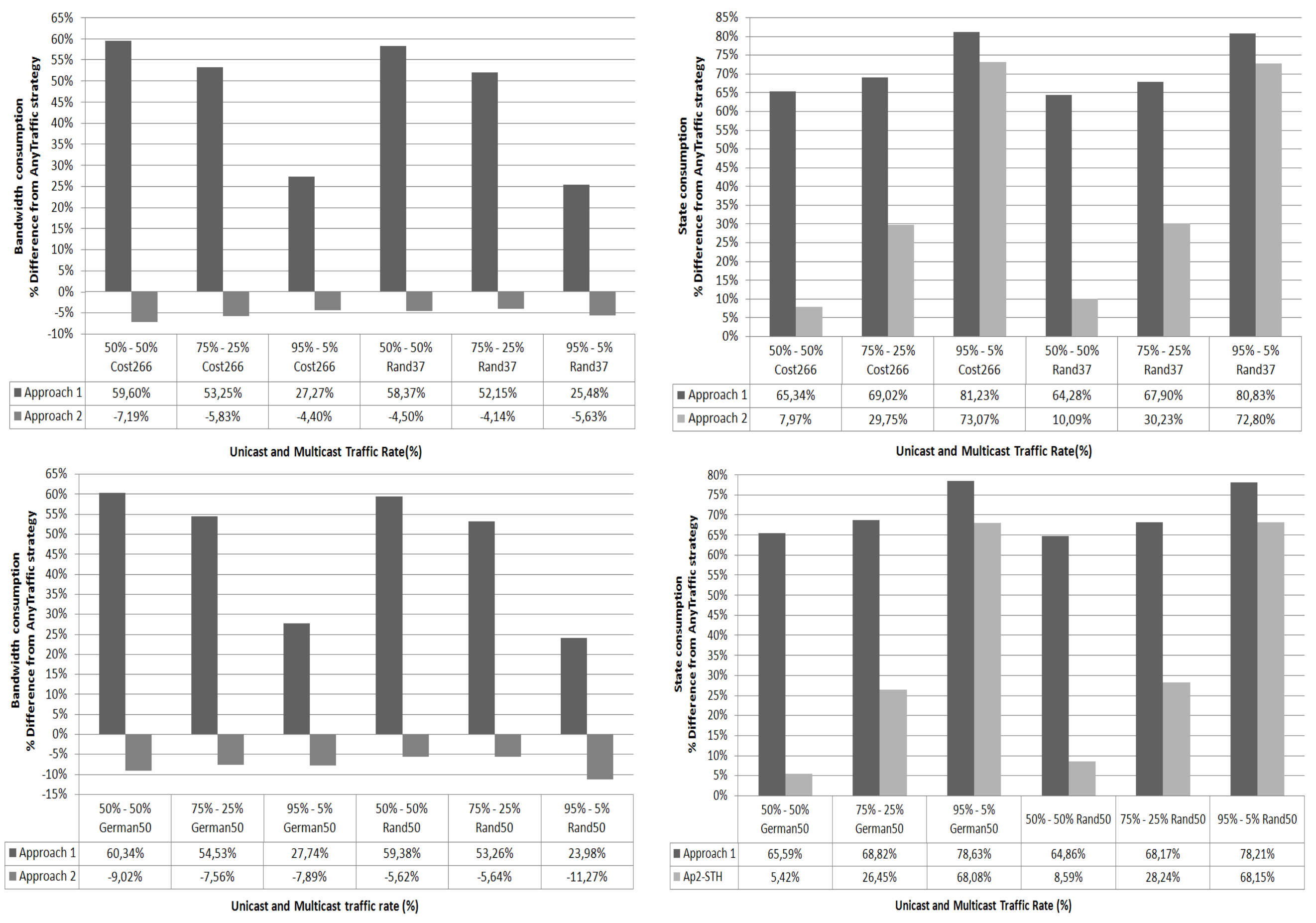
Figure 7.4 (left-column) shows the results obtained for the networks of 37 nodes, namely Cost266 and Rand37, in terms of relative gain in state and bandwidth consumption, respectively, with respect to the generated percentage of unicast and multicast traffic ratios.

As it can be seen from those figures, the proposed approach (AP3) has an outstanding performance in terms of state consumption compared with both AP1 and AP2 to the range of 50%-95% of unicast traffic rate. As unicast traffic rate increases, the gain is higher. From the figures above, for the interval of 75%-25% to 95%-5% of traffic rate in both pair of networks (37 and 50 nodes), we have state consumption gains ranging from around 30% to 70%.

Even though the network entities created by the multicast traffic requests are fewer, there is more unicast traffic that goes over them (i.e. network entities), reducing the number of states consumed with respect to AP1 and AP2.

In terms of bandwidth consumption, the AP3 has worst performance due to the longer paths that unicast traffic has to follow. The tendency of bandwidth consumption is inversely to the state consumption. The additional bandwidth decreases because with less AnyTraffic entities created by multicast requests, the unicast traffic goes more often by P2P (shortest) data paths, becoming closer to the AP1 and AP2 values.

However, this does not invalidate that a considerable amount of unicast traffic is still carried by means of AnyTraffic trees as said before. Nevertheless, these values can be improved at the expense of decreasing a fraction of the state consumption gain by tuning the algorithm (e.g. changing the values of γ and σ).



	50% - 50% Cost266	75% - 25% Cost266	95% - 5% Cost266	50% - 50% Rand37	75% - 25% Rand37	95% - 5% Rand37
Approach 1	59,60%	53,25%	27,27%	58,37%	52,15%	25,48%
Approach 2	-7,19%	-5,83%	-4,40%	-4,50%	-4,14%	-5,63%

	50% - 50% Cost266	75% - 25% Cost266	95% - 5% Cost266	50% - 50% Rand37	75% - 25% Rand37	95% - 5% Rand37
Approach 1	65,34%	69,02%	81,23%	64,28%	67,90%	80,83%
Approach 2	7,97%	29,75%	73,07%	10,09%	30,23%	72,80%

	50% - 50% German50	75% - 25% German50	95% - 5% German50	50% - 50% Rand50	75% - 25% Rand50	95% - 5% Rand50
Approach 1	60,34%	54,53%	27,74%	59,38%	53,26%	23,98%
Approach 2	-9,02%	-7,56%	-7,89%	-5,62%	-5,64%	-11,27%

	50% - 50% German50	75% - 25% German50	95% - 5% German50	50% - 50% Rand50	75% - 25% Rand50	95% - 5% Rand50
Approach 1	65,59%	68,82%	78,63%	64,86%	68,17%	78,21%
Ap2-STH	5,42%	26,45%	68,08%	8,59%	28,24%	68,15%

Figure 7.4: Bandwidth (left-column) and State (right-column) consumption gains: 37-nodes (Cost266 and Rand37) and 50-nodes (German50 and Rand50) networks (top and bottom respectively).

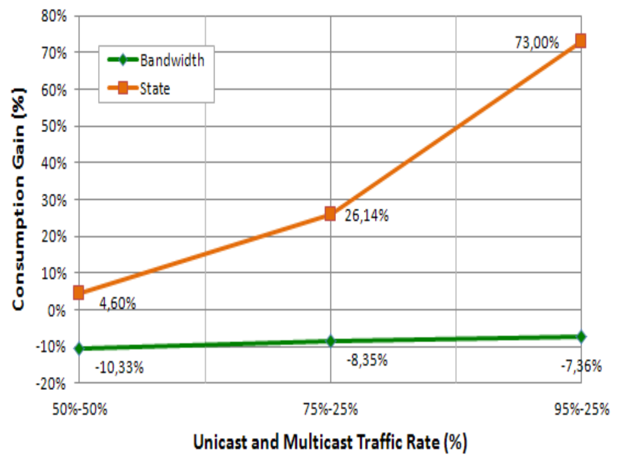


Figure 7.5: 100-nodes network: State and bandwidth consumption gains.

The same behavior is observed for the networks of 50 nodes, namely German50 and Rand50, shown in Figure 7.4 (right-column). However, results are a bit less favorable compared to the results obtained with Cost266 and Rand37. In terms of state consumption, the gain decreases from 32% to 6% for the range 50% to 95% of unicast traffic when compared to the Cost266 network. This observation reflects that more nodes with higher node degree influence the performance of the AnyTraffic algorithm. The bandwidth consumption here is a little higher although it can be reduced by tuning the algorithm to lower the aggregation of data paths. For instance, for the German50 network with 75%-25% of traffic rate, the AP3 has around 27% of state consumption gain when compared with AP2 versus an additional bandwidth consumption of around 8%.

We also observe, for networks of identical size but lower clustering coefficient, that the algorithm performs better because it favors path aggregation at a lower deficit $\Delta P_{s,d}$.

The overall algorithm's behavior for larger topologies is close to the small topologies experimented before. Figure 7.5

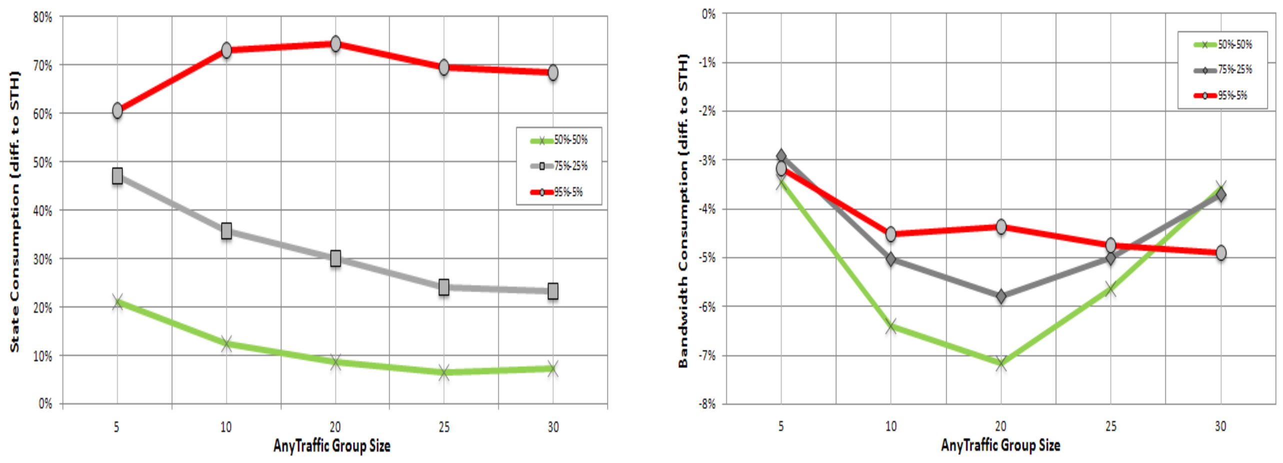


Figure 7.6: State (left) and bandwidth (right) consumptions in function of the AnyTraffic Group Size for the 37-nodes Cost266 network.

shows the performance of the AnyTraffic algorithm for a 100-node network topology. To reduce computational time, only 25% of nodes as ingress-egress nodes are considered. As regards state consumption, the algorithm demonstrates good performance compared with AP2 in the range 50%-95% of unicast traffic. The gain curve follows an exponential growth from 4% up to 70%. In terms of bandwidth consumption, AnyTraffic routing shows worst performance (up to -10%) for unicast traffic due to the longer paths followed by this traffic. Although the bandwidth consumption gain remains negative, it shows a positive trend from -10% to -7%. Indeed, as less AnyTraffic trees are created by multicast requests, forwarding unicast traffic requires more P2P shortest paths (consuming less bandwidth). Therefore, the value of -10% can be considered as an upper bound in terms of bandwidth consumption.

AnyTraffic routing dependency: Group size

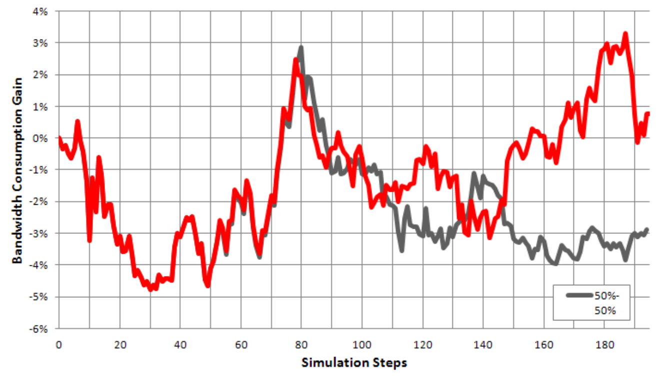
The dependency of AnyTraffic routing with respect to the size of the group has also been studied. Here, we fix the group size $|M| = 5, 10, 20, 25, 30$. Figure 7.6 shows the system resource consumption gains to the 37-nodes Cost266 network.

In terms of state consumption (graphic on the left), although the gain is always positive for the whole set of unicast-multicast traffic pairs, a decreasing trend is observed. As the group size increases, the created AnyTraffic trees are pushed up to their limits (i.e. stretched) by becoming longer and thus requiring more states. The worst case scenario occurs when the group size is maximum. Nonetheless, the gain is always positive: 6% to the $50/50$ (ucast/mcast)traffic pair, 20% to the $75/25$ and 55% to the $95/5$. The increase on the consumption gain for the 95/5 traffic pair, as the group size increases, is due to the impact of the 5% of multicast sessions. Although the percentage of multicast sessions/trees is low, as bigger the group size $|M|$ is, higher is the probability that a receiver node of a P2P data path also be a multicast member.

As regards bandwidth consumption (graphic on the right), the worst gain is around -7%. The concave shape observed for the $50\% - 50\%$ and $75\% - 25\%$ traffic pairs is steeper as the percentage of unicast traffic increases (as longer tree branches increase the bandwidth consumption). Such shape may be explained by the following: the inflection point occurring at $|M| = 20$ perhaps is due to the fact that such group size on the one hand increases the probability that a destination node receives simultaneously unicast and multicast traffic, forcing the aggregation of such P2P data paths, and on the other hand due to such path aggregation, the branch paths are longer which increases the bandwidth consumption. However, considering the former state consumption gains and what represents the state maintenance problem, a worse case performance of -7% in bandwidth consumption has limited effect in view of available link capacity of today's networks.

7.4.2 Dynamic Traffic Scenario

Simulations consist in processing several dynamic requests in which receivers join and leave an AnyTraffic group during its lifetime. Such scenario is modeled as a sequence of join and release requests where the bandwidth resources are



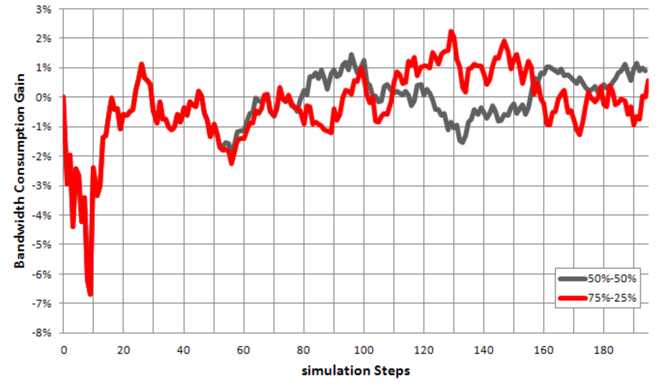
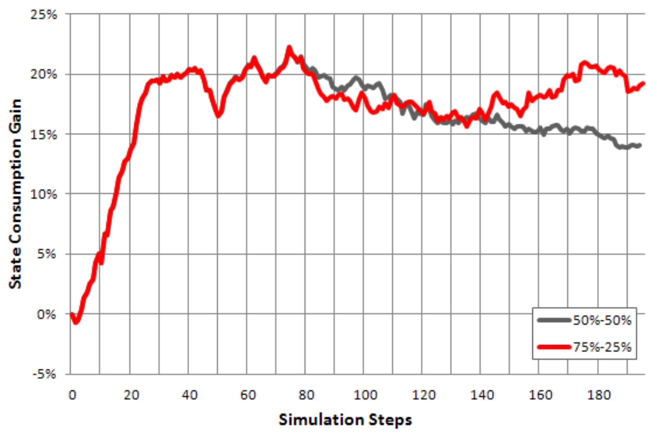


Figure 7.7: Bandwidth consumption: Cost266 (top) and German50 (bottom) network.



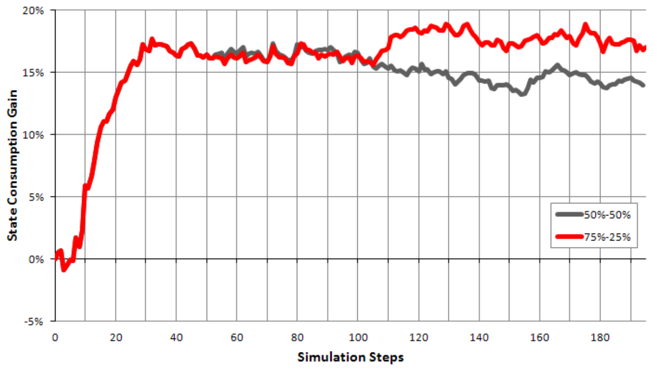


Figure 7.8: State consumption: Cost266 (top) and German50 (bottom) network.

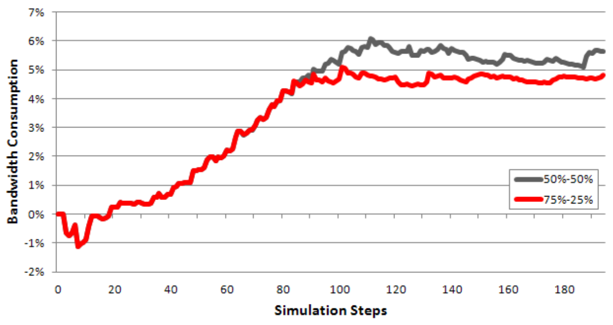
limited to a maximum link capacity set to 10 Gbps. Each simulation step represents one request processing for every node in the network. A probability that follows a non-stationary distribution is associated to each join/release request. This distribution starts with a 100%-0% join/leave probability up to a 50%-50% balanced stage, after several simulation steps.

In order to perform the comparison with the AnyTraffic algorithm, we have extended the STH algorithm to process dynamic requests. A Greedy tree based algorithm [Fei02] is implemented which consists of finding the closest attachment point via the shortest path. Nevertheless, both algorithms must met similar conditions, starting from the set of nodes N i) the node to join must be any node not belonging to the tree it wants to join, and ii) the node to release must be a node of the tree, except the source. A request is rejected when a tree computation fails because there is insufficient bandwidth remaining to process the request or there is not a feasible path (i.e. fulfilling the specific constrains of the algorithm) to join a receiver.

Analysis

The results are presented for two different network topologies, namely the Cost266 and German50 network, and considering only the 50%-50% and 75%-25% unicast-multicast traffic pairs. Figure 7.8 shows the states consumption in both networks. Similar behavior is observed. The consumption gain has an initial linear growth that corresponds to the generation of 100% join data path requests to after stabilize due to the arrival of data path release requests. Both types of traffic pairs have similar growth in both networks. Note that the consumption gain for the 75%-25% pair is always higher than the 50%-50%, except in occasional occasions when small oscillations occur.

For what regards the bandwidth consumption, Fig.7.7 shows higher oscillations in the Cost266 than in the German50 network. In the Cost266 network, the gain oscillates between negative and positive for both traffic pairs up to a point (around the 130 request processing) where they diverge. For the German50 network the gain oscillates less and moves around zero for both 50%-50% and 75%-25% traffic pairs.



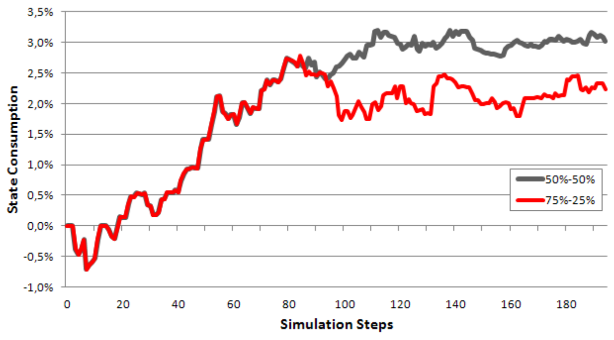


Figure 7.9: Cost266 network: Bandwidth (left) and State (right) consumption gain for the AnyTraffic algorithm when performing with and without deviation/deterioration control mechanism. The threshold for tree re-computation is set to 20%.

Although the high consumption gains in terms of states, mainly due to the forwarding of both unicast and multicast traffic over a single network entity, the bandwidth consumption gain is negative most of the time (maximum peak of -5%). This is due to the longer paths can be often taken on the AnyTraffic tree to forwarding unicast traffic.

Figure 7.9 shows the bandwidth and state consumption gains of the AnyTraffic algorithm when performing with control of deterioration (point 3 of subsection 7.2.3). The deterioration threshold to decide either to continue with on-the-fly tree setup or to perform the entire tree re-computation is set to 20%. After several simulation iterations, we set $w = 0.6$ in Eq.7.8. From Fig.7.9 (right), it can be observed that re-computation gain in terms of state consumption gradually grows to stabilize around 3% for the 50%-50% traffic pair and around 2% for the 75%-25% pair. The difference in the percentage of multicast requests explains this 1% gain variation. Fewer trees decrease the number of common forwarding entries for both unicast and multicast traffic. The low gain values obtained when re-computing the entire tree means that deviations from the optimum are not significant. Note that similar behavior is observed for the bandwidth consumption (not shown). In order to avoid waste of computational resource, a periodic deviation control can be performed when computing the tree on-the-fly.

7.5 Summary

In this study, we addressed the problem of state maintenance overhead in presence of multicast data traffic. The concept of AnyTraffic data group was introduced by the first time, which allows a group of destination nodes to receive both unicast and multicast traffic over the same source-initiated minimum-cost tree (or AnyTraffic tree). A specific heuristic algorithm was devised to accommodate this new routing approach: keep the system resource consumption (for state maintenance) overhead as low as possible while avoiding bandwidth waste by i) relying on replication of multicast traffic at branch points only (like in approach 2) and ii) keeping unicast traffic transmission over "as short as possible" data paths (like in approach 1).

The initial results obtained with the AnyTraffic routing algorithm, when applied to labeled-based forwarding (labels encode topological information) are promising. By stretching the shortest path for unicast traffic forwarding, common forwarding entries can be shared for both unicast and multicast traffic forwarding along the AnyTraffic tree toward the labeled topological locations, and thus the number of forwarding states significantly reduced. The AnyTraffic routing algorithm was compared against traditional schemes routing (standard Shortest Path and classical Steiner tree) using extensive simulation experiments to demonstrate the effectiveness of the proposal and settling the algorithm bounds: it gives a minimum improvement bound on state consumption and higher bound on bandwidth utilization increase.

After a first static scenario attempt, we extended the AnyTraffic tree algorithm to efficiently operate in a dynamic environment with bandwidth constraints, where a edge node receiving traffic can be joined and released of a multicast group (dynamic multicast sessions). In particular, we defined: 1) a grafting and pruning mechanism to allow receivers to join and leave the tree over the multicast session lifetime, respectively, and 2) a deterioration detection mechanism which acts as a trade-off between unnecessary waste of resource due non-optimal configuration of the multicast trees and the re-computation of the entire tree at each join/leave change. The performance analysis demonstrates the effectiveness of the novel AnyTraffic routing algorithm. On one hand, it obtains good improvements against traditional routing schemes in presence of bandwidth constraints. On the other hand, in presence of dynamic changes of leaves, the mechanism devised here to avoid full re-computation every time one of this type of requests arise, has a good performance indeed with a closest performance of full re-computation.

Future work: it is expected the addressing of the following issues: (1) adaptation of the routing algorithm to optimize those cases of p2mp-tree (partial) overlapping where the same source-initiated multicast groups share almost the same nodes. This will improve the state consumption results; (2) refining the proposed algorithm in order to achieve load-balancing and dynamical use of available bandwidth resources; (3) investigation of other maximum deficit function; (4) execution of the algorithm on Internet-like topologies (power law random graphs) with increasing number of nodes up to O(10k) to further determine the dependencies of the algorithm on the node degree, and the clustering coefficient; and finally (5) elaborate on distributed processing (in particular, under dynamic conditions).

Chapter 8

Name-Independent Compact Multicast Routing

Compact routing schemes address the fundamental tradeoff between the memory-space required to store the RT entries and the length of the routing paths that these schemes produce. In other words, a routing scheme is compact if it is memory efficient. Its goodness is measured by its stretch. And the main goal is to minimize the RT size at each node.

Such routing schemes have been extensively studied following the model developed in the late 1980's by Peleg and Upfall [Peleg89]. Since then, in accordance to the distinction operated by Awerbuch [Awerbuch89], between name-dependent, also called labeled, (nodes names are named by polylogarithmic size labels encoding topological information) and name-independent (node names are topologically independent) schemes, various compact routing schemes have been designed, notably in [Thorup01] and [Abraham08], respectively. These schemes are universal (they are designed so as to operate on any graph) however they are limited to p2p traffic (from a given source to a given destination).

As recently introduced by Abraham et al. in [Abraham09], dynamic compact multicast routing algorithms enable the construction of p2mp routing paths from any source to any set of destination nodes (or leaf nodes). The tree determined by a p2mp routing path is commonly referred to as a Multicast Distribution Tree (MDT) as it enables the distribution of multicast traffic from any source to any set of leaf nodes. By means of such dynamic routing scheme, MDTs can dynamically evolve according to the arrival of leaf-initiated join/leave requests. The routing algorithm creates and maintains the set of local routing states at each node part of the MDT. From this state, each nodes part of the MDT can derive the required entries to forward the multicast traffic received from a given source to its leaves.

This Chapter introduces the first known name-independent compact multicast routing algorithm enabling the leaf-initiated, distributed and dynamic construction of the MDT. Hereafter, the proposed routing algorithm is referred to as PPC (Pedroso, Papadimitriou and Careglio). The novelty of the proposed algorithm relies on the locally obtained information (proportional to the node degree) instead of requiring knowledge of the global topology information (proportional to the network size). The main challenge consists thus in mitigate the communication cost, i.e., the number of messages exchanged to build the entire MDT, while keeping an optimal stretch - memory space tradeoff.

To validate the design of our algorithm we first determine the theoretical performance bounds in terms of i) the stretch of the p2mp routing paths it produces, ii) the memory space required to store the resulting RT entries, and iii) the total communication or messaging cost. Next, we evaluate by simulation the performance of the proposed algorithm. For this purpose, we simulate the execution of the proposed algorithm on synthetic power law graphs comprising 10k, 16k and 32k nodes that are representative of the Internet topology. We further compare the obtained performance results with the corresponding Internet CAIDA map. To contrast the actual gain obtained with the proposed algorithm, our performance analysis comprises on the one hand, the comparison against those results produced by the Abraham compact multicast routing scheme [Abraham09]; and on the other hand, against those from two reference schemes, namely the Shortest Path Tree (SPT) and the Steiner Tree (ST) algorithm, when running over the same topologies.

8.1 Concept and Prior Work

As said before, the novelty of the PPC scheme relies on the locally obtained information (proportional to the node degree) instead of requiring knowledge of the global topology information (proportional to the network size). Therefore, during the MDT construction, the routing information needed to reach a given MDT is acquired by means of an incremental two-stage search process. This process, triggered whenever a node decides to join a given multicast source, starts with a local search covering the leaf node's neighborhood. If unsuccessful, the search is performed over the remaining unexplored topology (without requiring global knowledge of the current MDT). The returned information provides the upstream neighbor node along the least cost branching path to the MDT rooted at the selected multicast source node.

Prior work on compact multicast routing is as far as our knowledge goes mainly concentrated around the schemes developed in the seminal paper authored by Abraham in 2009 [Abraham09]. One of the reasons we can advocate is that despite the amount of research work dedicated to compact unicast routing, current schemes are not yet able to efficiently cope with the dynamics of large scale networks. Therefore, running compact multicast routing independently of the underlying unicast routing system would be beneficial. This independence is even the fundamental concept underlying multicast routing schemes such as Protocol Independent Multicast [6]. Nevertheless, we also observe that the scaling problems already faced when multicast routing received main attention from the research community, remain largely unaddressed since so far. Indeed, multicast currently operates as an addressable IP overlay (Class D group addresses) on top of unicast routing topology, leaving up to an order of 100millions of multicast routing table entries. Hence, the need to enable p2mp routing paths (for bandwidth saving purposes) while keeping multicast addressing at the edges of the network and build shared but selective trees inside the network. Indeed, in our approach, multicast forwarding relies on local port information only. Thus memory capacity savings comes from i) keeping 1:N relationship between network edge node and the number of multicast groups and ii) local port-based addressing for the local processing of multicast traffic. Further, we argue that compact multicast routing, by providing the best memory-space vs stretch tradeoff, can possibly address these scaling challenges without requiring deployment of a compact unicast routing scheme.

Preliminaries

Consider a network topology modeled by an undirected weighted graph $G = (V, E, c)$, with $|V| = \nu$, where V represents the finite set of nodes or vertices (all with multicast capabilities), $|E| = \mu$, where E represents the finite set of links or edges, and c a non-negative link cost function $c : E \rightarrow Z^+$ that associates a cost $c(e(u, v))$ to each link $e(u, v) \in E$. For $u, v \in V$, let $c(u, v)$ denote the cost of the path $p(u, v)$ from u to v in G, where the cost of a path is defined as the sum of the costs along its edges. Let S, $S \subset V$, be the finite set of source nodes, and $s \in S$ denote a source node. Let D, $D \subseteq V \backslash \{S\}$, $|S| << |D| << |V|$, be the finite set of all possible destination nodes that can join a multicast source s, and $d \in D$ denote a destination (or leaf) node. A multicast distribution tree $T_{s,M} = (V_T, E_T)$ is defined as an acyclic connected sub-graph of G, i.e., a tree rooted at source $s \in S$ with leaf node set M, $M \subseteq D$.

Comparison

We outline the dynamic compact multicast routing scheme for join-only events scheme as designed by Abraham et al. (part of their seminal work [Abraham09]). In Section 8.6, we provide a detailed comparison between the performance results obtained with our routing scheme and their approach. In Appendix A.4, a set of some self-explanatory examples of such algorithm behavior is also provided.

The Abraham scheme relies on the off-line construction of a bundle $_k$ of sparse covers $TC_{k,2^i}$, defined as $_k = \{TC_{k,2^i}(G) | i \in I\}$ with $k = log(n)$. Sparse covers are grown from a set of center nodes $c(T_i(v))$ located at distance at most $k2^i$ from node v, where $T_i(v)$ denotes the tree in the collection of rooted trees $TC_{k,2^i}(G)$ that contains the ball $B(v, 2^i)$. For each $i \in I$ and $T \in TC_{k,2^i}(G)$, the center node $c(T(v))$ of each node $v \in T$ stores the labels of all

[1] For simplicity, we present here the label-dependent variant of the scheme. In the name-independence version, center nodes store label mappings from names to nodes.

nodes [1] contained in the ball $B(v, 2^i)$, the ball centered on node v of radius 2^i.

Further, the $SPlabel(v)$ stores the label $\lambda(T, c(T))$ for each $T \in (v)$, defined as set of all covers T in the bundle Bk such that $v \in T$. In addition, each node $v \in V$ stores the tree routing information $\mu(T, v)$ for all the trees in its own label $SPlabel(v)$. When a leaf node u desires to join an MDT, it first determines whether or not one of the MDT nodes is already included in its local tree routing information table. If this is the case, it sends the join request to the center node $c(T_i(v))$ with minimum degree cover $i \in I$ that is associated to that MDT node v. The center node $c(T_i(v))$ then passes the label $\mu(T_i(v), u)$ so that the selected MDT node v can forward the multicast traffic to the newly joining leaf node u (without further propagating this label to the source node s). Otherwise (some the leaf node covers define an empty intersection with the MDT), the leaf node u with $SPLabel(u)$ queries the source node s to obtain the set of MDT nodes it currently includes.

Among all index $i \in I$, it then selects the tree $T_{i^*}(v)$ whose intersection with its bundle $B(u)$ is minimum. Once the node, say v, part of this intersection is selected $(T_{i^*}(v) \in B(u))$, leaf node u directs the join request to the associated center node $c(T_{i^*}(v))$. The latter passes a label $(T_{i^*}(v), u)$ so that the selected MDT node v can forward the incoming multicast traffic to the newly joining leaf node u. In order for the source node s to reach node u, node v has to propagate the tuple $[v, c(T_{i^*}(v)), (T_{i^*}(v), u)]$ to source s. The leaf node u updates all nodes covered by its balls $B(u, 2^i)$ to allow them joining the MDT at node u.

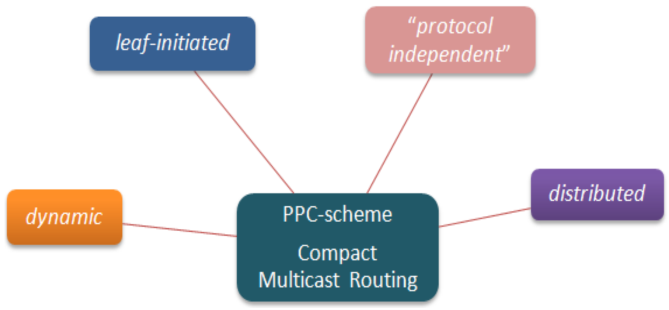


Figure 8.1: PPC routing scheme.

Compared to the Abraham scheme [Abraham09], our name-independent PPC compact multicast routing algorithm is also:

"leaf-initiated" since join/leave requests are initiated by the leaf nodes; however, contrary to the Abraham scheme it operates without requiring prior local dissemination of the node set already part of the MDT or keeping specialized nodes informed about nodes that have joined the MDT.

"distributed" since transit nodes process homogeneously the incoming requests to derive the least cost branching path to the MDT without requiring any centralized processing by the root of the MDT or any specialized processing by means of pre-determined center nodes as the Abraham scheme.

"dynamic" since the incoming requests are timely processed on-line as they arrive without re-computing and/or rebuilding the MDT from scratch.

"independent" refers to its independence from any underlying unicast routing topology required by leaf-initiated multicast routing schemes such as PIM [RFC4601] or any underlying sparse cover construction grown from a set of center nodes (which induce node specialization driving the routing functionality): the local knowledge of the cost to direct neighbor nodes is sufficient for the proposed routing scheme to properly operate. As such, it is actually a true "protocol independent" multicast routing scheme. It is important to emphasize that the sparse cover underlying the Abraham scheme is constructed off-line and requires global knowledge of the network topology to properly operate.

The main (known) limitation of our proposed routing scheme is that is not totally oblivious, in the sense that leave events may result into MDT re-organizations. Means by which such triggered-adaptation can be performed are still under investigation. One possible way to address this limitation consists in notifying downstream nodes from the leave event(s) at the expense of increasing communication cost.

8.2 Algorithm Performance Metrics

In this Section, we detail on the metrics used to analysis the performance of the proposed routing algorithm: the stretch, the storage (i.e., memory-bit space) and the communication cost metrics.

8.2.1 Stretch

The stretch (of a routing scheme) is defined as the ratio over all source-destination pairs between the routing scheme path cost/length and the minimum path cost/length for the same source-destination pair. Intuitively, the stretch of a routing scheme provides a quality measure of the path cost/length increase it produces compared to SP (SP routing schemes, either AS-path length based (path vector routing) or cost-metric based (link-state routing), are stretch-1).

This metric is important to measure the goodness of the routing algorithm as compact routing schemes (producing reduced RTs) are not always able to choose the minimum cost/length path for a given destination. Nevertheless, the routing scheme should favor computation and/or selection of routes whose stretch remains closer to 1. In a multicast context, the stretch is given by the total weight of edges, w_e used by the algorithm (alg) to deliver the multicast packet from source s to all leaf nodes $D \subseteq V$, where V is the total number of nodes, divided by the weight of the minimum optimal tree (opt) sourced at $s \in S \subseteq V$. The optimal tree is given by Steiner Tree (ST) algorithm.

$$stretch = \frac{cost_{alg}}{cost_{opt}} = \frac{\sum w_e^{alg}}{\sum w_e^{opt}} \tag{8.1}$$

8.2.2 Storage (i.e. RT Size)

The memory complexity of a multicast routing scheme is expressed in terms of the maximum number of memory-bits required to locally store the RT entries (the next-hop, destination information associated to any routing path) produced by the routing algorithm, i.e., storage. Thus, the RT size is computed using the bit-size of a single entry and the number of entries it comprises. The storage required by the algorithm is directly related to routing system scalability because the less memory a router needs to store its entries, the more scalable the routing system would be. SP routing schemes are incompressible, i.e., for all nodes in for all graphs, their lower bound equal their upper bound, i.e., $O(nlogn)$ bits are required to store their RT entries [Gavoille96], [Krioukov07]. Therefore, one must take into account the fundamental tradeoff that exists between the stretch of a routing scheme and the size of the RT it produces when designing a routing scheme.

The terminology used here to model the multicast routing information bases is borrowed from PIM [RFC4601]. The following Routing Information Bases are defined:

TIB (Tree Information Base): is the multicast RT. It essentially stores the state of all multicast distribution trees necessary to forward multicast packets at a router. MFIB (multicast forwarding information base) is derived from the TIB. In turn, from the TIB entries (multicast RT entries), forwarding entries are derived that are used to forward the multicast traffic.

URIB (Unicast Routing Information Base): is the unicast control message routing. It is used to determine the next-shortest hop and/or the path (e.g. like BGP) towards a node of the network.

MRIB (Multicast Routing Information Base): is the multicast control message routing. The MRIB is used to determine the next-hop neighbor to which any Join/Prune message is sent (towards tree root or source). This is the multicast topology table (usually derived from underlying "unicast" routing table (URIB), e.g., PIM runs as an overlay routing on top of SP producing shortest-path trees. MRIB entries are constructed to be used for MDT build up and maintenance.

For each one of the algorithms considered in this work, we have the following number and format of RT entries:

A. Shortest-Path Tree (SPT) Algorithm

Three types of routing entries are involved in this algorithm, namely URIB, MRIB and TIB (see Table 8.1). The URIB entries are maintained by every node in the network, consisting in one entry indicating the path towards the multicast source as a sequence of Autonomous System (AS) interfaces, plus M routing entries enabling communication with direct neighbors. This stems because when a route is received from one interface it does not necessarily propagate to all other interfaces; the MRIB entries are maintained by each node of the tree and are derived from the URIB entries; finally the TIB entries also maintained by each node of the tree to forward data packets towards the destination nodes of the multicast group.

B. Steiner Tree (ST) Algorithm

With the absence of unicast traffic, there is no need to URIB entries to be created. In such a way, the MRIB is constructed

Table 8.1: RT entries description: SPT Algorithm.

Type	$\|Entries\|$	Description	Format
URIB	1	AS-Path towards the multicast source, s, defined as a sequence of ASes.	$< p_{s-d} >$, as $AS_s + AS_{k_1} + ... + AS_d$
URIB	M	It enables the communication with the direct neighbors.	<direct neighbor,cost/distance>
MRIB	1 entry per multicast group G $\backslash\{s\}$	It indicates the upstream neighbor which to send the join requests (towards the source s along the tree)	< (S,G), upstream neighbor address>
TIB	1 entry per multicast group G	One entry (state) per multicast distribution tree i.e. each state corresponding to the local entry of the tree for source s and group G.	(S, G) state

Table 8.2: RT entries description: ST Algorithm.

Type	$\|Entries\|$	Description	Format
MRIB	1 per multicast group G $\backslash\{s\}$	MDT topology description, i.e., it indicates the upstream neighbor to which to send the join requests along the MDT.	< (S,G), upstream neighbor address>
MRIB	1 per node of the network	It indicates the best next hop neighbor to which to send the join requests towards the MDT.	<(S,G), best next hop node address>
TIB	1 per multicast group G	One entry (state) per MDT i.e. each state corresponding to the local entry of the tree for source s and group G	(S,G) state

Table 8.3: RT entries description: PPC Algorithm.

Type	$\|Entries\|$	Description	Format
MRIB	1 per multicast group G $\backslash\{s\}$	MDT topology description, i.e., it indicates the upstream neighbor to which to send the join requests along the MDT.	< (S,G), upstream neighbor adress>
TIB	1 per multicast group G	One entry (state) per MDT i.e. each state corresponding to the local entry of the MDT for source s and group G.	(S,G) state necessary to forward the multicast packets.

Table 8.4: Data Structure of the RT entries.

Type	Format	Size (in bits)
URIB	$< \sum^{S} AS\_address >$	$\|S\| * 16$
URIB	$<< interface\_address >, cost >$	$32 + 16 = 46$
MRIB	$<< source\_address, group\_address >, interface\_address >$	$32 * 3 = 96$
TIB	$<< RPF\_neighbor\_addr, source\_addr, group\_addr >, \sum interface\_addr >$	$32 * 3 + \sum^{n} address$

from the algorithm itself and not from the URIB, as in the SPT case (see Table 8.2). In this case, every node of the network should maintain an MRIB entry indicating the least cost node of the MDT to join if it decides to. The dissemination of such current MDT information to remote nodes of the network is done by flooding and not by directed propagation of routing information like in the SPT case. Thus, the previous M URIB entries (from SPT) are discarded. We could indeed assume directed forwarding in the ST case too but this would make the processing at intermediate nodes more complex; hence the flooding procedure is kept for the ST case. The TIB entries are also maintained as in the previous algorithm.

C. PPC Algorithm

The proposed algorithm needs to keep only MRIB and TIB type of entries (see Table 8.3). No URIB entries are created or maintained by this algorithm at any rate (there is NO global view of the topology). As regards the MRIB, it is also constructed from the algorithm itself. Upon the join event of a leaf node, the search procedure of the algorithm will flood its request messages throughout the node's interfaces. There is a clear distinction between "port" and RT entry to a non-adjacent neighbor, otherwise we would have a comparison problem with respect to the overall classification of entries. The gain from PPC scheme comes from its independence from any unicast routing with respect to the SPT and the absence of dissemination information (locally stored) with respect to the ST.

The structures are maintained per interface during the construction (in the MRIB) but released afterwards. A non selected node is not stateful for subsequent searches, which is re-initiated – in fact the tradeoff is to keep full structures per multicast source and then the entries become part of the MRIB (e.g., ST) or release/rediscover for each request but then the communication cost is rather high (e.g. PPC).

Data Structure of the Routing Entries

Each single routing entry must be encoded using a proper data structure scheme, helping to derive its size in number of bits. For instance, let us consider an interface encoded over 32 bits, an address over 32 bits, an AS over 16 bits (as an AS's path being defined as a sequence of AS's) and cost/distance metric over 16 bits. This values are extracted from [RFC4601] and illustrated in Table 8.4.

A compact TIB entry is assumed where it contains several outgoing interface addresses (to the multicast case only). The multicast forwarding mode is performed according to the Reverse Path Forwarding (RPF) mechanism, in which a data packet is accepted for forwarding only if it is received on an interface used to reach the source in unicast.

8.2.3 Communication Cost

The dynamic nature of the routing protocol such as those currently deployed over the Internet allows each router to be kept up to date with respect to non-local topological changes (resulting from topological failures, addition/withdraw of routes and ASes). The latter information is exchanged between routers by means of routing information updates (each router timely distributes to its own peers following specific selection criteria the routing information received from other peers).

Communication cost is defined as the number of routing updated messages that needs to be exchanged between routers to converge after a topology change. Recently, [Korman06] showed that the communication cost lower bound for scale-free graphs is at best linear up to logarithmic factors. The number of routing updates may change according to the advertisement technique (time or event-driven).

8.3 PPC Routing Algorithm

Objective

The objective of the proposed algorithm is to minimize the RT sizes of each node $n \in V$ at the expense of i) routing the packets on p2mp paths with relative small deviation compared to the optimal stretch obtained by the ST algorithm as well as ii) higher communication cost compared to the SPT algorithm.

For this purpose, the proposed algorithm reduces the local storage of routing information by keeping only direct neighbor-related entries rather than tree structures (as in ST) or network graph entries (as in both SPT and ST). In other terms, the novelty of the proposed algorithm is on requiring maintenance of only local topology information while providing the least cost next hop during the MDT construction. That is, our algorithm does not rely on the knowledge of the global topology information or involve the construction of global network structures such as sparse covers. The information needed to reach a given multicast source s is acquired by means of a two-stage search process that returns the upstream node along the least cost branching path to the MDT sourced at s. Such mechanism is triggered whenever a node decides to join a given multicast source s, root of the MDT. After a node becomes member of an MDT, a multicast routing entry is dynamically created and stored in the local TIB. From these RT entries, multicast forwarding entries are locally instantiated.

As stated before, the reduction in memory space consumed by the RT results however in higher communication cost compared to the reference algorithms, namely the SPT and the ST. Higher cost may hinder PPC-scheme applicability to large-scale topologies such as the Internet. Hence, to keep the communication cost as low as possible, the algorithm's search process is segmented in two different stages. The rationale is to put tighter limits on the node space by searching locally in the neighborhood (or vicinity) of the joining leaf node before searching globally. Indeed, the likelihood of finding a node of the MDT within a few hops distance from the joining leaf is high in large topologies (whose diameter is logarithmically proportional to its number of nodes) and this likelihood increases with the size of the MDT. Hence, we segment the search process by executing first a local search covering the leaf node's vicinity ball, and, if unsuccessful, by performing a global search over the remaining topology. By limiting the size (or order) of the vicinity ball taking into account the degree of the node it comprises, one ensures an optimal communication cost. For this purpose, a variable path budget p_{budget} is used to limit the distance travelled by leaf initiated requests to prevent costly (in terms of communication) local search or global search. Additionally, as the most costly searches are resulting from the initial set of leaf nodes joining the multicast traffic source, each source constructs a domain (referred to as source ball). When a request reaches the boundary of that domain it is directly routed to the source.

In the following sections, we describe the design of the proposed compact multicast routing algorithm. We first provide an overall description of the proposed algorithm. Then, we detail the local and global search phases used to discover the least cost branching path from the joining leaf node to the MDT. We also specify the design of the on-line and distributed construction algorithm underlying the incremental least cost branching path discovery process.

8.3.1 Description

The MDT $T_{s,D}$ is constructed iteratively. At each step ω ($\omega = 1, 2, .., |D|$) of the leaf-initiated construction, a randomly selected node u joins $T_{s,M}$, where $M \subseteq D$ corresponds to the current set of nodes part of the MDT at a given construction step.If node u is already part of $T_{s,M}$ ($u \in V_T$) then it is either a transit or branching node of the MDT. Otherwise, node u is not part of $T_{s,M}$ ($u \in D \backslash \{V_T\}$) and it must search for the least cost branching path from node u to node $v \in T_{s,M}$. Among the set $P_{u,v}$ of possible paths $p(u, v)$ from node $u \notin T_{s,M}$ to node $v \in T_{s,M}$, the least cost branching path $p(u, v)*$ is denoted as follows:

$$p(u, v)* = min\{c(u, v)|p(u, v) \in P_{u,v}\} \tag{8.2}$$

In this equation, the cost $c(u, v)$ of the path $p(u, v)$ is defined as the sum of the (tangent) cost $c(u, w)$ of the edge (u, w), where $w = succ(u)$ refers to the upstream neighbor node of u, and the (radial) cost $c(w, v)$ of the path $p(w, v)$, i.e., $c(u, v) = c(u, w) + c(w, v)$. When each node along the least-cost branching path $p(u, v)*$ from leaf node u to $v \in Ts, M$ determines its upstream neighbor node along that path, leaf node u can send a request message to join $T_{s,M}$ to its selected upstream neighbor node. The join message is relayed along the selected least-cost branching path $p(u, v)*$ until it reaches node $v \in T_{s,M}$. Once node u has joined $T_{s,M}, u \in V_T$, and the set M comprises node u, we proceed to the next step by randomly selecting a node $w \in D \backslash V_T$. The Least Cost Branching Path Discovery procedure is detailed in the following subsection.

The iterative construction process ends when all candidate leaf nodes have joined the MDT, i.e., $M = D$ and $D\ V_T = \emptyset, T_{s,M} = T_{s,D}$. The RT of each node $v \in T_{s,M}$ ($v \in V_T$) includes i) one RT entry (stored in the MRIB) that indicates the upstream neighbor node to which the join message is sent for each source node s; this locally stored information enables performing Reverse Path Forwarding check so as to ensure loop-free forwarding of the incoming multicast packets, and ii) one multicast traffic routing entry (stored in the TIB) to enable forwarding of incoming multicast traffic (generated from that source s) from its incoming port to a set of outgoing ports.

Two types of messages are involved at each step of the least cost branching path discovery process, namely the request (type-R) messages flowing in the upstream direction towards the multicast source s, and the response (type-A) messages sent in the downstream direction towards the joining leaf node u. Their content is described below:

Type-R message it is basically responsible to find the MDT. Each message comprises the following information i) a sequence number $\{u_{id}, r_{id}\}$ to prevent duplication of messages, where u_{id} identifies the leaf node u and r_{id} identifies its request to join the multicast source s, ii) the leaf node u's timer value $\tau(u)$ that sets the waiting time at intermediates nodes before answering back to the downstream neighbor node$pred(u)$, and iii) a path budget π, starting at leaf node u from $\pi(u) = \pi_{max}$, set at leaf node u. The π_{max} value is bound by the graph diameter (the length of the longest shortest path) for which approximation algorithms exist, as well as method for computing a lower and upper bounds [RFC4601]. Starting from leaf node u, the path budget $\pi(u)$ is decremented at each node v according to the travelled distance: each traversed edge accounts for a distance decrease of 1. When the type-R message reaches node $v, if \pi(v) = 0$, the latter does not further propagate the type-R message in order to keep the communication cost as low as possible; otherwise, the value $\pi(v)$ is decremented and passed to the neighboring nodes.

Type-A message is sent in response to type-R messages, each type-A message comprises i) the cost $c(w, v)*$ of the locally selected least cost path $p(w, v)*$ from the local node w to v such that $v \in T_{s,M}$; a node $v \notin T_{s,M}$ generating a type-A message to its downstream neighbor nodes sets this cost to infinite, and ii) when $v \notin T_{s,M}$, the identifier of node v.

The PPC algorithm and its sub-routines pseudo-codes can be found in the Appendix.

8.3.2 Least Cost Branching Path Discovery

Forward Direction: The process starts with the leaf node u sending type-R messages to its direct neighboring nodes w ($w = succ(u)$). If $w = v \in T_{s,M}$, it replies back immediately a type-A message to its downstream node(s), indicating, in this case, $c(w = v, v) = 0$. Otherwise, node w decrements the message budget value π by 1 and set its local timer for such type-R message, $\tau(w)$.

Node w then forwards the type-R message throughout its neighboring nodes (equal to $degree(w) - 1$), except to the node from which the incoming type-R message was received (split horizon). We say that node w sends a type-R message to its upstream neighbor nodes. To decrease the communication cost, node w checks the resulting message budget π before sending the type-R message, discarding those messages with $\pi = 0$. Moreover, node w keeps the received type-R message sequence number in order to prevent duplication of messages toward its upstream neighboring nodes and consequently to avoid loops. Thus, when node w receives another type-R message requesting to join the same multicast source s as part of the multicast group, this message is not further propagated to its upstream neighbor nodes.

Note that node w simply records the incoming edge (to subsequently send a type-A message when appropriated), and the number of incoming type-R message can be at most equal to $deg(w)$. This process continues until the type-R message reaches a node $v \in T_{s,M}$ (for the first leaf, v corresponds to the multicast source s).

Backward Direction: Before answering back to its downstream node(s), node $w \neq u$,$v \in T_{s,M}$ must verify one of the following conditions: i) having received the entire set of type-A messages from its upstream neighboring nodes before its timer $\tau(i)$ expires, or ii) having waited until expiration of its timer $\tau(i)$ - note that $\tau(i)$ is initiated after the reception of the first type-R message received.

Once one of these two conditions is met, node w computes all the candidate branching path costs $c(w, v)$ using the information received from its upstream neighboring nodes. It then selects the intermediate least cost one $p(w, v)*$ and sends the corresponding cost value, $c(w, v)*$ to its downstream node(s). If the number of type-A messages received is null by the timer elapsing, the cost value $c(w, v)$ is set to infinite, indicating that the multicast source s is unreachable. Node further downstream in the leaf node u direction may ignore these messages if they receive type-A message(s) from other upstream nodes with finite cost value $c(w, v)*$.

The algorithm terminates when the leaf node u receives all type-A message (in response to the type-R messages it initiated) and determines the upstream neighboring node along the least-cost branching path $p(u,v)*$ towards the MDT. Again, if the number of type-A messages received is null by the the timer expiration or the cost value $c(u,v)$ in all received type-A messages is set to infinite, leaf node u declares the multicast source s unreachable. Leaf node u further proceed by sending to this upstream neighboring node a Join request message ϕ_u requesting connectivity to the $T_{S,M}$.

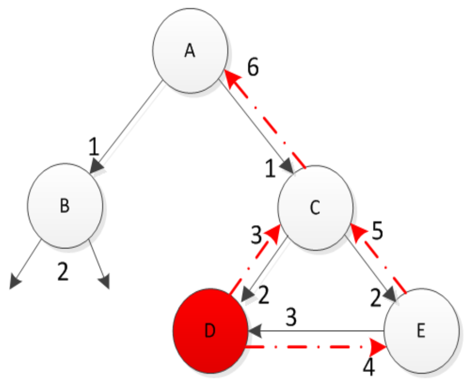


Figure 8.2: Join Process: normal arrows represent the Type-R messages and dot arrows the Type-A messages. Node A can be either a leaf node (e.g. node u) or a transit node not part of MDT, while node D is part of MDT.

Example Figure 8.2 illustrates an example of the general search procedure. Normal and dot arrows represent type-R and type-A messages, respectively. Let us assume that node $A \in V \notin T_{s,M}$ sends type-R messages to its upstream neighboring nodes B and C ($t = t_1$). Because nodes $A, B \notin T_{s,M}$, each one of them repeats the same process that node A and send type-R messages to their neighboring nodes (except, by application of split horizon, to the node A from which they received the type-R message).

Let us now focus in the neighboring node C. It sends type-R messages to node D and node E ($t = t_2$). As node $D \in T_{s,M}$, it answers back to node C with a type-A message ($t = t_3$). On the other hand, node $E \notin T_{s,M}$ forwards itself the type-R message to its neighboring nodes (again, except to node C due to split horizon - $t = t_3$). Note that node C does not reply back to node A while it does not receive a type-A message from node E (or the timer τ of node C expires). Assuming node E has only 2 interfaces, it receives the type-A message from node D at $t = t_4$. In turn, node C receives it at $t = t_5$. When the number of type-A messages received by node C matches the number of type-R messages sent, it computes the least cost branching path and replies it (either $p(c,d)$ or $p(c,e,d)$) to node A, at $t = t_6$.

8.3.3 Segmentation of the Search Space

The communication cost can be a serious constraint in large topologies, $O(> 1k)$ nodes. Although the disruptive attitude in the design of the proposed algorithm, which leads to considerable gains in the RT size, it also brings a high number of exchanged messages that may be unacceptable: searching the entire topology every time a leaf node u decides to join a MDT is too costly from a communication perspective (strong impact on e.g. algorithm time convergence and computational cost). In order to mitigate it, the algorithm's search process is segmented in two different stages. The rationale is to put tighter limits and search locally before search globally. As said before, the likelihood of finding a node of the MDT within a few hops distance from the joining leaf is high in large topologies (whose diameter is logarithmically proportional to its number of nodes) and it increases with the size of the MDT.

The segmentation of the algorithm's search space consists in execute first a local search covering the leaf's neighborhood, and if unsuccessful, executing a global search over the remaining topology. Note that a dedicated tag, called $flag\_e$, which enables distinguishing between messages exchanged during the search phases, is added to both types of messages (type-R and type-A). Both messages are tagged as internal when setting $flag\_e = 0$ (if belonging to the local search procedure), and as external when $flag\_e = 1$, otherwise.

Local Search This first stage consists in a limited search within a certain perimeter of the topology around the joining leaf node u. As illustrated in Fig.8.3, the contiguous set of nodes covered during this first stage is called vicinity ball, $B \subseteq V$. Each node $b \in B$ is therefore referred to as vicinity node. The vicinity ball B of node u, $B(u)$, is delimited by vicinity edge nodes, $b_v(u)$, i.e., nodes $v \in V$ at a given hop-count distance from node u.

The way this vicinity ball is determined as a huge impact on the communication cost as we observed. In Section 8.6, we show the differences on the communication cost by using the initial methodology (the number of vicinity nodes proportional to $n^{0.5}/log(n)$, where π is decremented at each hop

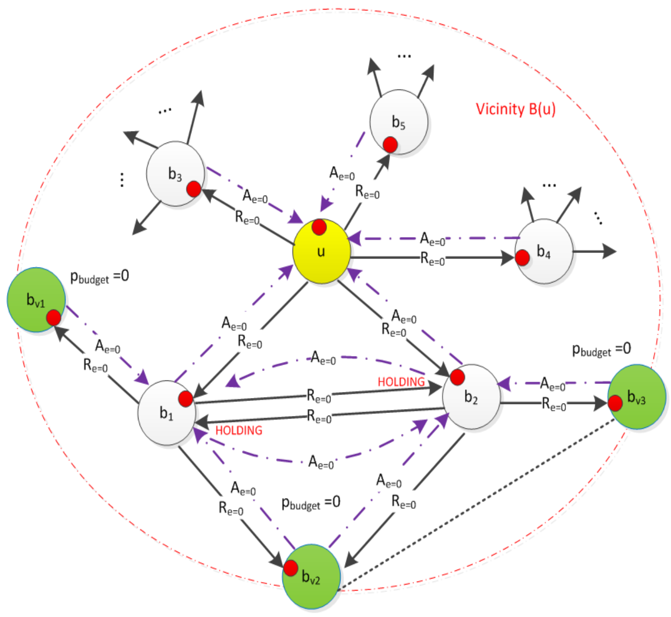


Figure 8.3: Local Search stage: search the node of the MDT within a limited perimeter called vicinity, B(u).

with the vicinity node's out-degree. Nodes setting $\pi < 0$ are identified as vicinity edge nodes of $B(u)$) or instead the methodology described below:

At leaf node u, the path budget π carried in the type-R message is initialized by setting its value $\pi(u) = \pi_{max}$. If the degree of node u, $degree(u) \leq \alpha$, then $\pi(u) = 3$, if $\alpha \leq degree(u) < \beta$, $\alpha < \beta$, then $\pi(u) = 2$; otherwise $\pi(u) = 1$. Values α and β are small integer values determined a priori from the node degree distribution characterizing power law graphs. Starting from node u, where $\pi(u) = \pi_{max}$, the path budget π is decremented by 1 at each node v if $\pi(v) < \beta$; otherwise it is set to 1. This condition prevents propagation of the type-R message beyond adjacent nodes of high degree nodes. At node v, if the decremented $\pi(v)$ value reaches 0, node v does not further propagate the type-R message in order to keep the communication cost as low as possible while increasing the likelihood of finding a node $v \in T_{s,M}$. This procedure determines the maximum distance that type-R messages with $flag\_e = 0$ can traverse and determines the edge nodes of node's u vicinity ball $B(u)$. Indeed, vicinity edge nodes $b_v(u)$ are the nodes for which the path budget π reaches 0.

For instance, let us consider the case in Fig.8.3. The leaf node u sets $\pi(u) = \pi_{max} = 2$, assuming that $\alpha < degree(u) = 5 < \beta$. At the neighboring node b_1 of node u, the budget gets reduced to $\pi(b_1) = \pi(u) - 1 = 2 - 1 = 1$ as $\alpha < degree(b_1) = 4 < \beta$ (the same for b_2). As expected, in the next neighboring nodes, i.e., b_{v1}, b_{v2} and b_{v3}, $\pi = 0$ - setting the vicinity edge nodes.

When a given leaf node u decides to join the multicast source s, it sends a type-R message ($flag_e = 0$) to all the direct upstream neighbor nodes of node u (referred to as $succ(u)$) to find the least cost branching path $p(u, v)*$ to a node $v \in T_{s,M}$ ($v \in V_T$). Again using Fig.8.3 as an example, leaf node u sends type-R message to nodes $b_1, \ldots, b_5$. At condition that $succ(u)$ has not yet processed a type-R message with the same sequence number, $succ(u)$ successively propagate the message following a split horizon until it reaches either a node $v \in T_{s,M}$ or a node $v \notin T_{s,M}$ and $\pi(v) = 0$. In the latter case, a vicinity edge node v is reached (node $v = b_v$) but no node belonging to $T_{s,M}$ can be found. The role of vicinity edge nodes is described next in the Global Search procedure.

At this point, node v replies to its downstream neighbor node(s) from which it has received the type-R message(s) with a type-A message. The type-A messages sent by node v in response to its downstream neighbor nodes $w = pred(v) \notin T_{s,M}$ are processed as follows. If node $v = b_v \notin T_{s,M}$, then the type-A message (issued by node v to its downstream neighbors) sets the branching path cost to infinite. If not, then the type-A message (issued by node v to its downstream neighbors) sets this cost to value 0 which indicates that node $v \in T_{s,M}$.

Subsequently, each node $w \neq b_v, v = succ(w)$, computes the branching path costs $c(w, v)$ from itself to each node v by using Eq.8.2, where either $v \in T_{s,M}$ or $v = b_v \neq T_{s,M}$. Node w then selects the least cost branching path $p(w, v)*$ and sends the corresponding cost value $c(w, v)*$ to its own downstream node(s) x such that $w = succ(x)$. Observe that each node w maintains no additional routing information besides the degree(w) entries required at each step of the execution. At waiting timer $\tau(u)$ expiration, if the set of type-A messages received by node u is empty or if the cost c(succ(u),v) is set to infinite in all received type-A message, node u declares the multicast source s unreachable (launching the global search). Otherwise, leaf node u determines among its neighbor nodes $succ(u)$ from which it received type-A messages, the upstream node $succ(u*)$ along the least-cost branching path $p(u, v)*(= min\{c(u, succ(u*)) + c(succ(u*), v)|p(u, v) \in P_{u,v}\})$ from node u to $v \in T_{s,M}$. Leaf node u then further proceeds by sending a message to $succ(u)$ to join $T_{s,M}$.

Global Search: This stage represents the search of the MDT's branching node outside the vicinity of the joining leaf node. This process is triggered by the leaf node u when the local search phase declares the multicast source s as unreachable in its vicinity ball $B(u)$. However, it "only" starts at each vicinity edge node $b_v(u)$ - see Fig.8.4. During this search phase, type-R and type-A messages are tagged as external (i.e., $flag\_e = 1$). In order to start a global search phase without restarting from the local neighborhood of the triggering node, the following procedures are considered:

- ✓ The first procedure enables external type-R messages reaching the vicinity edge nodes without traveling again the complete set of nodes inside its vicinity ball $B(u)$. For this purpose, the leaf node u sends the external type-R messages directly to each of its vicinity edge nodes. Targeted forwarding of these messages from the leaf node u to each vicinity edge node b_v is possible because i) during the local search phase, the internal type-A messages (i.e., $flag\_e = 0$) received by the leaf node u include the identifier of the node b_v that initiates them, and ii) each vicinity nodes $b \in B(u)$ keeps per vicinity edge node $b_v u$ a single active interface from which type-A message with infinite cost has been received (indicating that the neighbor node sits along the path from leaf node u to a given edge node b_v).
- ✓ The second prevents that a given node $b \in B(u)$ receives back external type-R messages during the global search phase. For this purpose, vicinity edge node b_v filter incoming external type-R messages (i.e., $flag\_e = 1$). Remember that during the local search, internal type-A messages sent in response to the reception of type-R message ($flag\_e = 0$) are tagged with the $flag\_e = 0$. Interfaces sending such type-A message are removed from the list of interfaces for relaying type-R message ($flag\_e = 1$). The exception is for interfaces having received a type-R message ($flag\_e = 1$) with leaf node u as sender to enable edge vicinity nodes to send back the answer to node u once the global search completes for that node $b_v(u)$.

During the global search phase - see Fig.8.4 -, the $\pi(u)$ budget value is set at node u to a threshold equal to the graph diameter (length of the longest shortest path) and the waiting time $\tau(u)$ to a value that prevent waiting indefinitely. Moreover, upon reception of the type-R message ($flag\_e = 1$) from node u, each edge vicinity node $b_v(u)$ sets the maximum waiting timer $\tau(b_v(u)) = \tau(u) - 1$. The subsequent search process proceeds as follows: assume that node $b_v(u)$ sends an external type-R message to each of its upstream neighbor nodes except to its downstream node (part of the vicinity ball of the node from which the message has been received). At waiting timer $\tau(b_v(u))$ expiration, if the set of type-A messages received by node $b_v(u)$ is empty or if the cost $c(succ(b_v(u)), v)$ is set to infinite in all received type-A message, node $b_v(u)$ declares the multicast source s as unreachable. Otherwise, node $b_v(u)$ determines among its neighbor nodes $succ(b_v(u))$ from which it received a type-A message, the upstream node $succ(vb(u))*$ along the least-cost branching path $p(b_v(u), v)*$ to $T_{s,M}$, defined as:

$$p(b_v(u), v)* = min\{c(b_v(u), succ(b_v(u))*) + \\ + c(succ(b_v(u))*, v)|p(b_v(u), v) \in P_{b_v(u),v}\} \quad (8.3)$$

Node $b_v(u)$ is ready to answer back to node u once either of the following condition is met: i) it receives the entire set of type-A messages from its upstream neighbor nodes before its waiting timer $\tau(b_v(u))$ expires or ii) the waiting timer $\tau(b_v(u))$ initiated after reception of the first type-R message ($flag\_e = 1$) from leaf node u expires. When one of these two conditions is met, node $b_v(u)$ selects the

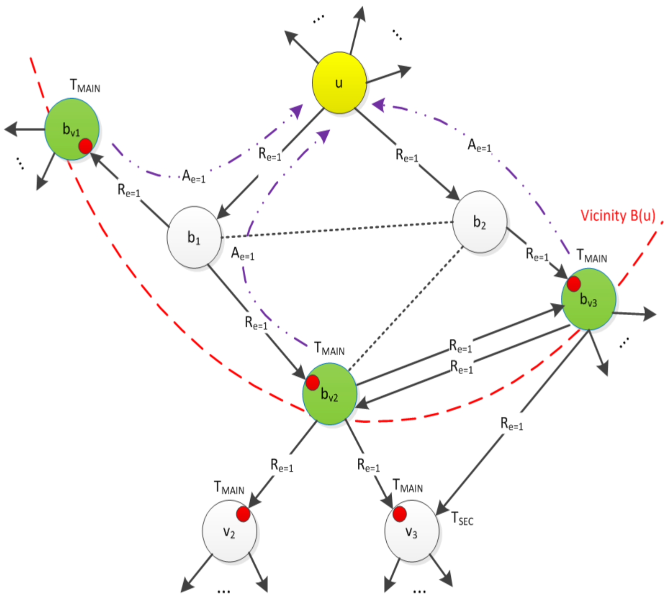


Figure 8.4: Global Search stage: If local search fails to find a node of the MDT, a search outside the vicinity must be performed.

least cost branching path $p(b_v(u), v)*$ and sends the corresponding cost value, $c(b_v(u), v)*$ directly to the joining node u. At waiting timer $\tau(b_v(u))$ expiration, if the set of type-A message received by node $b_v(u)$ is empty, the cost value $c(b_v(u), v)*$ is set to infinite indicating that the multicast source s is unreachable. Hence, as soon as this search phase completes, each node $b_v(u)$ returns a unique type-A message ($flag\_e = 1$) directly to the leaf node u from which it initially received an external type-R message. Thus, contrary to the local search stage, no computation or selection is performed by nodes $b \in B(u)$ along the path taken by the type-A messages ($flag\_e = 1$) sent towards the leaf node u.

This relay path is determined by the incoming interface maintained by each node $b \in B(u)$ upon reception of type-R message ($flag\_e = 1$) from leaf node u. Figure 8.4 shows the node b_1 receiving two type-A messages from b_{v1} and b_{v2}. In opposition to the previous stage, here b_1 does not perform any computation or routing decision. It just forwards the incoming type-A messages received from nodes b_{v1} and b_{v2} towards the leaf node u. The leaf node u receives as many type-A messages as the number of vicinity edge nodes defined. A global timer is set at the joining leaf node u to prevent a too long waiting time.

Note that the temporary records locally created during the local search phase are subsequently deleted by the node sending a type-A message ($flag\_e = 0$) that does not include an infinite cost to a vicinity edge node $b_v(u)$. The remaining records, locally created during the global search phase, are deleted by the node sending a type-A message ($flag\_e = 1$).

8.3.4 Source Node Vicinity Ball

As the most costly searches are resulting from the initial set of leaf nodes joining the MDT, each source constructs a source ball such that when a type-R message reaches the boundary of that domain it is directly routed to the source. This prevents searching at the neighborhood of the multicast traffic source. For this purpose, the multicast source node initiates a procedure that builds a ball around the size shall be at least as big as the average leaf node size. To be effective, this procedure shall construct a ball with i) size at least as large as the average size of leaf node's vicinity ball, and ii) radius computed from its outgoing ports shall be inversely proportional to the neighbor's node degree.

The procedure performs as follows: the source collects its neighbor's node degree to reach at a minimum size $x = x_{min}$ but up to a certain maximum diameter, d_{max}. At each collection step the following conditions are checked:

- ✓ If the value x reaches the optimal value of the ball size x_{opt} and diameter $d = d_{opt} \leq d_{max}$, then we get an optimal solution. The source ball construction stops.

✓ If $x > x_{opt}$ and the diameter d being reached is such that $d << d_{opt}$, then the source node s selects additional neighbor nodes so as to increase the diameter (up to value d_{max}) while limiting the increase above the value x_{opt}. For this purpose, we define an epsilon ε of increase of x such that $x - \varepsilon \to x_{opt}$.

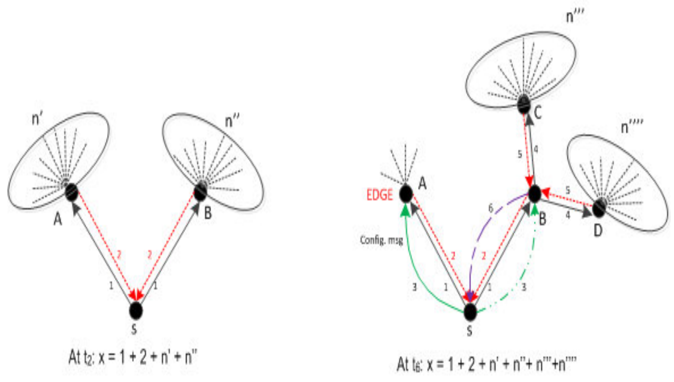

n'
n''
A
B
S
At t₂: x = 1 + 2 + n' + n''
n'''
n''''
EDGE
A
C
B
D
Config. msg
S
At t₆: x = 1 + 2 + n' + n''+ n'''+n''''

Figure 8.5: At t_2, $x > x_{opt}$. In this case, $y = max\{n', n''\}$. If $x - y > x_{opt}$, then both neighbor nodes A and B are set as edges. Otherwise, only node A is set as edge of the ball. (on the right) The source sends a config. message to node A and a "query" message through (neighbor) node B in order to reach in turn the adjacent (neighbor) nodes of B. These nodes reply towards the source with their node degree and the decision process is repeated at t_6 (iff $x <= x_{opt}$).

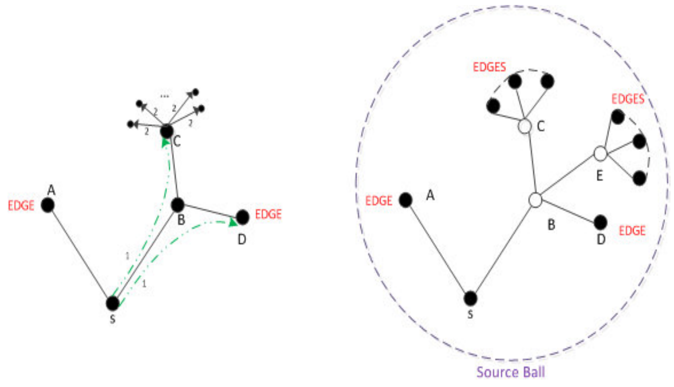


Figure 8.6: (on the left) A third iteration is shown, where the source sends both query and config. messages further then its direct neighbors. (on the right) The final aspect of the source ball is illustrated, where some branches are longer than others.

Epsilon ε is the number of adjacencies added from a given source node's neighbor t at distance $d(s,t)$ to a set of adjacent neighbors at distance $d(s,t)+1$ from s so as to reach a decent diameter while limiting the number of nodes in excess above x_{opt}. In practice, if $x > x_{opt}$, source node s selects neighbors located at a distance $d(s,t)$ the nodes with smallest degree. Note that each node $b_v(s) \in B(s)$ of the vicinity ball of the source node s, must now maintain an additional MRIB entry to relay type-R and type-A messages towards the source node s. Thus, in total $2 * (|B(s)| - 1)$ additional RT entries are to be considered. Note however that if a leaf node $u = b_v(s) \in B(s)$, then that node u does not need any more to trigger any search procedure.

8.3.5 Computational Complexity

The computational cost is defined with respect to time and resource complexity. While the time complexity is limited by the maximum time a leaf node u waits when joining an MDT (i.e., algorithm convergence time), the resource complexity consists in the trade-off between memory consumption (to keep the RT) vs. CPU (to process the messages). In our case, the RT "gains" provided by the proposed algorithm comes at the expense of higher communication and processing complexity.

Nevertheless, one of the main advantages of the proposed algorithm is its independence from any underlying unicast routing information and thus from any topological changes that do not affect the MDT. The algorithm terminates when the leaf node u has the upstream entry towards the MDT, which is computed upon the reception of all the type-A messages from its direct neighboring nodes. In a worst case scenario, the former happens within a maximum time $\mathcal{T}(u)$, which should cover the time (t_{max}) taken by type-R (forward direction) and type-A (backward direction) messages to ensure the leaf node u joins the MDT through the least cost branching path, $p(u,v)^* = min\{c(u,v) : p(u,v) \in P_{u,v}\}$. t_{max} is determined by the diameter of the graph G (the greatest distance between any pair of nodes), given by: $d(G) = max\{e(v) : v \in V(G)\}$. $\mathcal{T}(u)$ is defined as follows:

$$\mathcal{T}(u) = t_{max} = 2 * d(G) * (t_p) \ , \tag{8.4}$$

where t_p represents the propagation time. Note that the processing time of the messages at each node can be easily neglected, as well as the transmission time: the routing decisions time are constant and the information volume (i.e., message's size) to be processed is relatively small.

8.3.6 Theoretical Performance Bounds

In this subsection, we provide the theoretical performance bounds of the proposed algorithm in terms of i) the stretch of the p2mp routing paths it produces, ii) the memory space required to store the resulting routing table entries, and iii) the total communication or messaging cost.

Stretch

Minimizing the tree-cost sequentially, namely, the total cost of the edges used during the algorithm while building the multicast tree of the various stages and minimizing the tree-cost globally leads to different stretch bounds. As we consider a dynamic join scenario, the former is considered. The stretch bound analysis involves three different cases:

- ✓ Consider a joining node u and $s \in B(u)$. Then the local search initiated by node u will find the least cost branching path if the path budget $\pi(u)$ in the type-R message initiated is sufficient to reach the source node s. It is obvious to see that if this condition is met, the resulting stretch increase is minimal.

- ✓ Consider a joining node u and $s \notin B(u)$. If $v \in T_s, M$ and $v \in B(u)$, then the local search process will find the actual least cost branching path if and only if there no other node $w \in T_{s,M}$ from the joining node u that can be found at shorter distance, i.e., $d(u, w) \geq d(u, v)$. Indeed, the distance limit set by the joining node on the local search process by means of the path budget $\pi(u)$, allows to reach a node $v \in B(u)$ before triggering a global search. Due to the degree bound set when decrementing the path budget $\pi(u)$, a node $v \in B(u)$ may be reached during the local search phase before reaching node $w \in T_{s,M}$ and $w \notin B(u)$ such that $d(u, w) < d(u, v)$. Henceforth, the stretch increase is bound by the fraction of such nodes conditioned by the joining node selection and the current number of nodes $T_{s,M}$. The stretch bound can thus be derived from the following formula:

$$\frac{1}{N}\sum_{i=1}^{N}\frac{d(u, v_i)}{min\{d(u, w_i)\}} \tag{8.5}$$

  where, $\forall i \in M, |M| = N, \exists v_i, w_i \in V : min\{d(u, w_i)\} < d(u, v_i)$ with a probability $P(v_i \in T_{s,M} | v_i \in B(u)).P(w_i \in T_{s,M} | w_i \notin B(u)) > 0$.

- ✓ Consider a joining node u and $s \notin B(u)$. If $v \in T_{s,M}$ and $v \notin B(u)$, then the global search process will find the least cost branching path if the path budget $\pi(u)$ in the type-R message initiated is sufficient to reach the source node s. It is obvious to see that if this condition is met, the resulting stretch increase is minimal.

Storage

Each node $v \in T_{s,M}$ stores in its local RT at most one MRIB entry and at most one TIB entry whose size is proportional to the local tree out-degree k (as this entry indicates the outgoing ports for the incoming multicast traffic). Assuming a minimum port encoding proportional to $log(k)$, the storage size per node $v \in T_{s,M}$ is $O(klog(k))$ bits. Nodes $b_v(s) \in B(s)$ require an additionally storage capacity for the MRIB enabling to relay type-R and type-A messages.

Communication

Each join event as initiated by a leaf node u_i results in a communication cost $C(u_i)$, i.e., the number of messages exchanged for leaf node u_i to join a node $v \in T_{s,M}$, that is given by the following formula:

$$C(u_i) = 2m'X + 2m'Y + 2(m - m')Z \tag{8.6}$$

In this equation, m and m' are respectively the number of edges in the vicinity ball $B(u)$ of leaf node u and the total number of edges $|E|$. X is the probability that at least one node $v \in T_{s,M}$ is comprised in the ball $B(u)$ of the selected node $u \in V$, Y is the probability that none of the tree nodes $v \in Ts, M$ are comprised in the ball $B(u)$ of the selected node $u \in V$, and Z is the probability that all nodes $v \in V$ part of the tree $T_{s,M}$ are not in the ball of the selected node u (all tree nodes lie outside of the ball $B(u)$ of the selected node u).

The total communication cost, i.e., the cost to build the entire MDT is then determined by the sum of the individual communication costs $C(u_i)$ of the $i = 1, , |N|$ leaves composing the tree. In [Pedroso11b], we demonstrate that the derivative of Eq.8.6 with respect to the number of nodes in the vicinity of node u, $|B(u)|$, provides the value of the ball size that minimizes the communication cost while corresponding to the order of the largest connected sub-graph of diameter d_{max} that can be constructed.

8.3.7 Timer Setting Mechanism

Each upstream node, $v \notin D \notin T_{S,M}$, may receive as many type-R messages as its incoming interfaces (one per interface though), and sometimes lagged in time. And, as said before, it only replies back once it has all the type-A messages to its own sent type-R messages. In order to prevent infinite waiting time and reduce the algorithm convergence time, an overall timer mechanism is implemented.

The idea is to enable one timer per each interface receiving type-R messages (instead of a global timer per node set by the first request that arrives). By doing this, we guarantee that the sender nodes of those messages will always receive a type-A message whenever information is available (even if it is not the best at the moment). This mechanism consists in the following steps:

* The first type-R message to be received enables a timer defined as "main", T_{main}, and is propagated according to the former algorithm description.

* Next, each possible forthcoming type-R messages that may arrive, enable a proper timer defined as "secondary", T_{sec}, where $T_{main} > \forall T_{sec}$. These messages are not propagated and remain on a *holding state* until at least one type-A message is received or T_{sec} expires.

In a worse case scenario, a node v must wait T_{main} before it replies back to the downstream node from which it received the first type-R message (i.e., until it has all the type-A messages from its neighboring nodes) and T_{sec} to reply to those other downstream nodes from which it received those other type-R messages arriving later (whether with empty or incomplete information - in order to avoid deadlock situations).

For instance, consider the example depicted in Fig.8.7. Imagine that two type-A messages (one from node C and another from node E) arrives at node B at a time $t_{reply} \ll T_1 \ll T_2$. Yet, it cannot reply back to node A because it should wait also for a type-A message from node D. In a worst case, node D would reply closer to T_2. Whenever $T_2 = T'_2$, by the time node B receives the type-A message from node C, C's timer to node A's type-R message would have elapsed and the path solution (e.g. E-B-C) be lost. Note that all this becomes particulary problematic on a non-equal edge cost topology scenario and assuming equal decrement (x) of the timer values (approx. same propagation and transmission time in every edge), $x = t_{prop} + t_{tx}$.

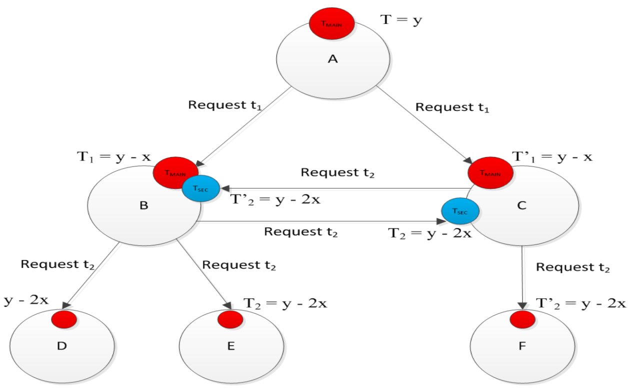


Figure 8.7: Main (red) and Secondary (blue) labeling of the nodes' timer (case 1 and 2).

Identified problems

There are particular cases due to topology characteristics that lead to some shortcomings in the whole timer setting process. This limits the performance of the algorithm, chiefly its convergence time. Those shortcomings are here identified (cases from 1 to 4), and some proposals to solve them suggested.

Case 1: refers to Fig.8.7. Upon the reception of type-R messages (from node B), nodes D and E set their (main) timers $T_2(D) = T_2(E) = y-2x$ (red). However, also node B sets a (secondary) timer to the type-R message coming from node C $T_{sec}(C) = T'_2 = y - 2x$ (blue), where $T_2 = T'_2$. In this case, if either node D or E replies closer to T2, node B will reply to node C with either an empty or incomplete type-A message.

Case 2: refers also to Fig.8.7. Node B must wait $T_{sec}(C) = T'_2 = y - 2x < T_1$ before it replies back to node A even if both nodes D and E have already replied much before their T_2 elapses. Note that a type-A message from node C must be also waited. In turn, node C will not reply to node B until itself receives a type-A message from node B too, which will never come before the timer $T_{sec}(C)$ of node B elapses (the same will happen with $T_{sec}(B)$ of node C.

Case 3: refers to Fig.8.8. The time that a leaf node $u = A$ waits to join a $T_{s,M}$ (i.e., receive he least cost path $p(u, v)*$) can be defined as the algorithm time convergence, Θ. This

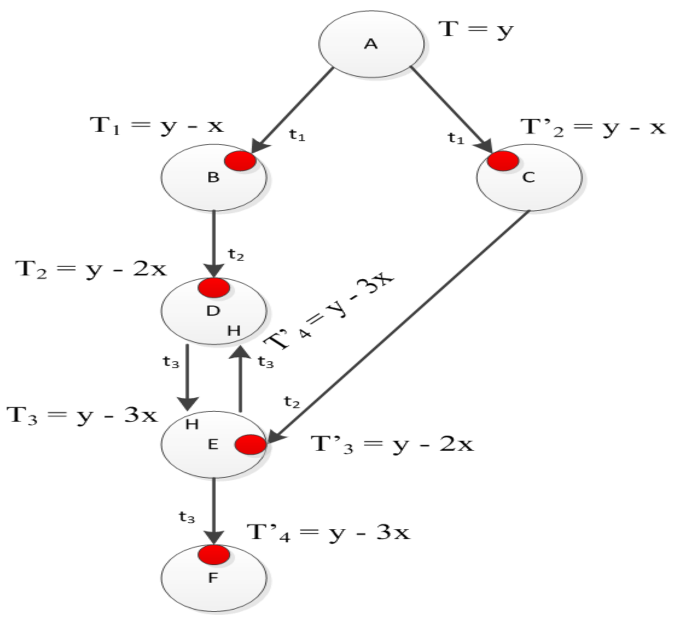


Figure 8.8: Case 3: The algorithm time convergence Θ may be maximum.

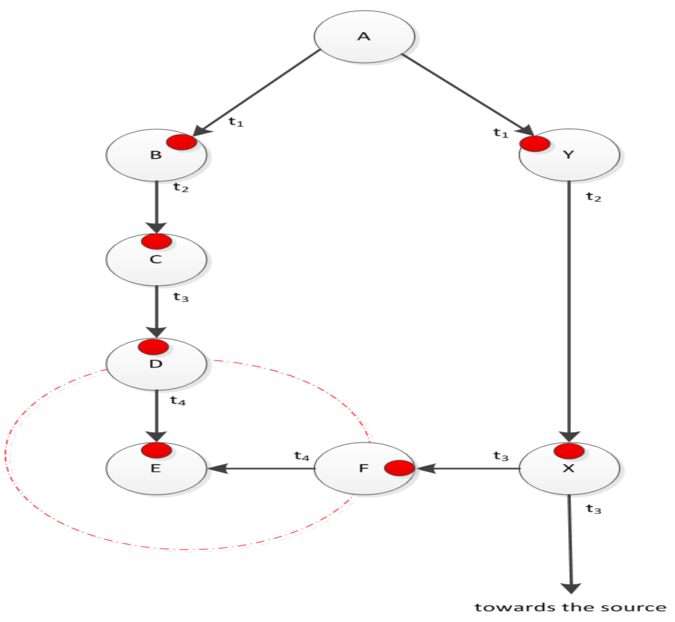


Figure 8.9: Case 4: deadlock situation either between nodes E-F or nodes D-E.

value is maximum ($\Theta = \Theta_{max}$) iff the type-R message expires its budget value π to find a node $v = F \in T_{s,M}$ that is at a distance d equal to the network diameter, i.e., of $d(p_{u,v}) = diameter(G)$. In this situation, the backward process is leveraged by a chain event of elapsing nodes' timers. As a consequence, the leaf node may compute some non-optimal branching path solution. According to the figure, when node E receives a type-A message from node F, its timer with respect to node D, $T_{sec}(D) = T_3 = y - 3x$ has already elapsed. In a non-equal edge cost scenario, it may that the path (A-B-D-E) would be the least cost compared to the path (A-C-E).

Case 4: refers to Fig.8.9. A deadlock occurs either between nodes E and F or nodes D and E (it depends on which type-R messages node E processes first). This requires node X waits $T_{main}(Y)$ to reply back to node Y, which receives an empty type-A message from its neighboring node F. In such a way, the branching path (A-B-C-D-E-F-X) is excluded from the evaluation at A.

Speeding up the type-A message reply

The algorithm convergence time can be considerably improved by adding some optimized procedures to handle the aforementioned cases. Nevertheless, a transversal improvement consists in trading the optimal branching path solution per the algorithm convergence time. Is to say, every time a node receives a type-A message, it can boost the waiting time count down in order to speed up the process up to the leaf node. Consequently, sometimes the leaf node may not receive the best branching path to join the tree but a lower convergence time would be achieved.

Case 1 (solution): node B should be able to detect such situation and reset T'_2. The new T'_2 value must be $T_1 < T'_2 > T_2$.

Case 2 (solution): node B sends an incomplete type-A message to node C whenever the following two conditions are verified (8.7) and (8.8) - note that $type - R = m_R$, $type - A = m_A$ and I_{out} is the number of outgoing interfaces. If a type-A message is received from node C before T_1, it unblocks the situation. The second condition (8.8) helps to guarantee that the type-A message is not empty (see case 4 solution).

$$\sum |m_R| = \sum |m_A| + \sum |m_R(holding)|, \tag{8.7}$$

and

$$|I_{out}| = |m_R| \neq 1, \tag{8.8}$$

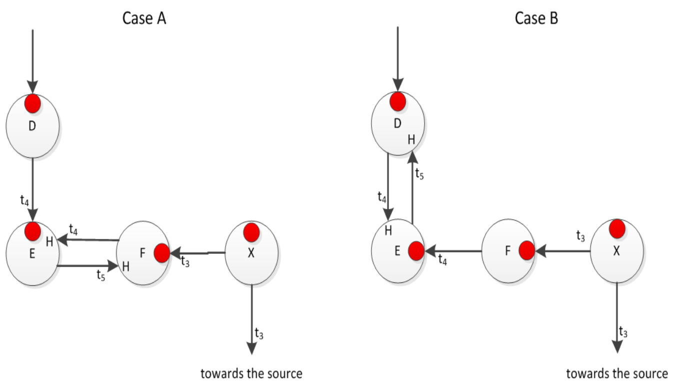


Figure 8.10: Case 4: zoom-in of the deadlock situation depicted in Fig.8.9.

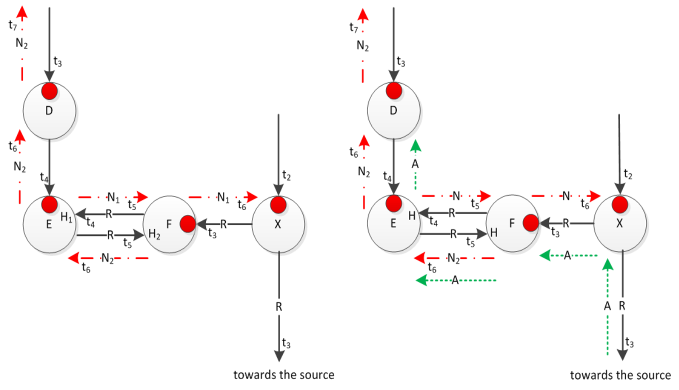


Figure 8.11: Case 4 (solution): an auxiliary type-N (i.e. negative) message is used to inform branching node X that it must send its type-A message also throughout such interface.

Case 3 (solution): if node F replies at T_4', there is no solution. Otherwise, a non-empty type-A message is forwarded.

Case 4 (solution): an auxiliary type of message, type-N, to be sent on the backward direction of the holding type-R message in order to inform the next branch node (with node degree ≥ 3) that it should forward a type-A message as soon as it receives one, using the same interface of such auxiliary type-N message. Figure 8.10 shows the two variations of the problem and Fig.8.11 illustrates the proposed solution (case A of Fig.8.10 is assumed). In the left hand-side scheme of the figure, node E sends a type-N message to node F, being consequently forwarded to node X. It stops at node X because it is a branch node (degree ≥ 3), which means it is waiting for replies on another interfaces too: node X is the node to be informed. In such a way, node X is now aware that it should forward a received type-A message (coming from other interface) through this marked interface (i.e. X-F), even if it is not suppose to as a type-R message has been previously sent - see right hand-side scheme of the figure.

Owing to this mechanism, node X can speed up its answer to node Y, at the same time that node E has a non-empty answer to send to node D (the sub-path: E-F-X-{sub-path}). Note that node F does the same when receives a type-R message that goes on hold. However, we know that we will not receive any useful information through this interface (in such example it would stop at node A). In order to avoid unnecessary communication cost, a budget value can be introduced to these type-N messages as for the type-R messages.

8.4 AnyTraffic-PPC Algorithm

Introduced in the previous Chapter 7, the AnyTraffic concept contributes with forwarding state reduction by sharing a single forwarding entry to handle both unicast and multicast traffic. The branching node selection is done according to a given pruning condition in order to guarantee a low increase of bandwidth consumption as well as of the length of unicast path (i.e., stretch).

Our objective is then to apply such concept to Compact Routing schemes, and therefore to introduce a completely novel approach, contributing to an even bigger reduction of the RT size. Take into account the PPC algorithm, this would result in a name-independent, leaf-initiated, distributed Compact AnyTraffic Routing algorithm, referred to as A-PPC scheme. This approach would not only save in URIB (as it eliminates them even in the presence of unicast traffic) and MRIB entries, but in TIB entries as well.

In order to achieve the proposed A-PPC scheme, some changes must take place at the standard PPC scheme. Thus, a new criterion should be added to the branching path selection process performed by the joining leaf node u. As in the AnyTraffic scheme, the resulting e2e path cost $c(u, s)$ must be lower than the maximum cost increment allowed for the stretch-1 SP between the same pair of nodes, which is given by the maximum deficit factor $\Delta_{\max}^{s,d}$ (Exp. 7.3 of Chapter 7). Note that the $c(u, s) = c(u, v) + c(v, s)$, where $v \in T_{s,M}$ (a candidate branching node).

In such a way, the leaf node u must be provided with additional information in order to perform the comparison (i.e., run the pruning condition). First, the type-A message is extended to carry the path cost of the MDT path between the candidate branching node $v \in T_{s,M}$ (i.e. the origin of the type-A message) and the MDT source node s, $c(v,s)$. In such a way, each node $v \in V_T$ needs to keep such information (i.e., $c(v,s)$) as one additional entry, resulting in a slight increase on the number of entries at the RT ($|MDT_{nodes}|$ new entries per MDT). Note, however, that the discovery mechanism remains unaltered where each intermediate node processing type-A messages maintains the routing computation on the branching path cost and not on the full path cost.

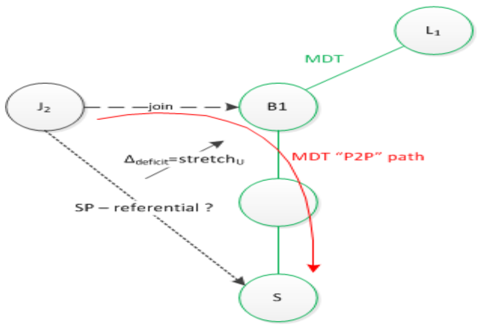


Figure 8.12: A-PPC scheme: how to find the $x_{s,u}$ of the p2p path in a distributed environment without incurring in too high communication cost?

The main challenge relaying under such distributed scenario is to find the proper way to provide the leaf node u with the reference cost ($x_{s,u}$) of the stretch-1 p2p path (($p_{s,u}$) without incurring in an extremely high communication cost and dependent of topology changes. The problem is illustrated in Fig.8.12. A more extensive study is required to deploy such an approach, which is left for future work due to time limitations. Here, only a preliminary study is conducted in order to assess how "stretched" are the individual p2p paths (from source to leaf node) of the MDT constructed by the PCC scheme if they had to carry unicast traffic. To this end, we compare the cost of those p2p paths (maximum, minimum and average) of the MDT constructed by the PPC scheme against the cost of the p2p paths of the MDT constructed by the SPT scheme (same source and multicast group is assumed).

This will give an idea about how many times the maximum deficit function is not accomplished ($cost_{p2p}^{PPC} - cost_{p2p}^{SPT} > \Delta_{\max}$) and to find the proper reasons that justify the deployment of the A-PPC scheme. The result of this preliminary study is presented in Section 8.6.3. The following ratio ($S_U(s,M)$) is defined to represent the aforementioned stretch.

$$S_U(s,M) = \frac{cost_{p2p}^{PPC}}{cost_{p2p}^{SPT}} \tag{8.9}$$

8.5 Simulation Environment

The performances of the PPC algorithm are analyzed by means of exhaustive simulations on large scale topologies representative of the Internet Autonomous System (AS) topology (addressed in the next subsection). The simulations are executed on the ad-hoc, event-driven Java simulator, which was previously developed for the AnyTraffic performance study, and here re-defined to implement the PPC algorithm (some compression techniques had to be applied to handle such large-scale topologies). The execution scenario considers the construction of p2mp routing paths (i.e. MDT) for leaf node sets of increasing size from a minimum of 500 up to 4000 nodes (selected randomly) with increment of 500 nodes. Almost every simulation is executed 10 times by considering 10 different multicast sources. The Shortest-Path Tree (SPT) and the Steiner Tree (ST) algorithms are executed over the same topologies in order to retrieve a performance comparison with state-of-the-art approaches.

- ✓ The SPT algorithm provides the reference for the communication cost. It is constructed from a loop-avoidance path-vector routing algorithm carrying the identifier of the multicast source s and the routing path to reach that source. Each node keeps thus a RT entry per neighbor node (to exchange messages) and a RT entry per path to the multicast source s.

- ✓ The ST algorithm provides the reference in terms of stretch. In order to obtain the near optimal solution for the ST, we consider a ST-Integer Linear Programming formulation. For this purpose, we have adapted the formulation provided in [SAGE] for bi-directional graphs (see Appendix A). The communication cost for the ST measures at each step of its construction the number of messages initiated by nodes part of the MDT. These messages contain the minimal information for remote nodes not (yet) belonging to the MDT to join it. Using this information, each node knows how to reach the closest node of the MDT. Thus, although the ST is computed centrally, the communication cost accounts for the total number of messages exchanged during the MDT building process as a dynamic scenario would perform.

Recall that stretch, storage (in terms of memory-bit space consumption and number of RT entries) and communication cost are the metrics used to perform the comparison. In particular, the RT comprises the MRIB, the TIB entries as well as the URIB entries for the SPT scheme that relies on the underlying unicast routing topology. Each RT entry must be encoded using a proper data structure, helping to derive its size (number of bits). For instance, let us consider an interface encoded over 32 bits, an address over 32 bits, an AS over 16 bits (as an AS's path being defined as a sequence of AS's) and cost/distance metric over 16 bits [RFC4601].

8.5.1 AS-Internet Representative Topologies

The PPC algorithm is envisioned to inter-domain routing of Future Internet, and thus we will run it over AS-Internet representative topologies. ASes are an important abstraction because they are the "unit of routing policy" in the routing system of the global Internet (under a single administrative control). ASes peer with each other to exchange traffic, and these peering relationships define the high-level global Internet topology. For the purposes of analysis, these peering relationships are represented with an AS graph, where nodes represent ASes and links represent peering relationships.

Both synthetic power-law topologies (10k, 16k and 32k nodes), generated according to the Generalized Linear Preference (GLP) model [Bu02], and corresponding Internet CAIDA topology map (16k and 32k-nodes) [CAIDA] are used in the assessment. The properties of these topologies are summarized in Table 8.5.

Table 8.5: Internet Representative Topology Properties

	Nodes - Links	**Avg. - Max. Node Degree**
GLP	10.000 - 35.432	7,09 - 675
	16.000 - 56.993	7,12 - 769
	32.000 - 120.436	7,53 - 1165
CAIDA	16.301 - 32.955	4.04 - 2331
	32.618 - 146.816	4,50 - 2520

The GLP evolutive topology generation model, which relies on generalized linear preferential attachment, produces power-law graphs representative of the Internet AS topology (one node models an AS). Furthermore, the Internet has scale-free properties that can be defined according to two main parameters, namely node degree and clustering coefficient:

Node degree (k) distribution: approximated by long tail power law distribution $P(k) \sim k^{-\gamma}$, $\gamma = 2.254$ (scaling index or power law exponent, typically 2 <= γ <= 3)

Clustering coefficient: characterizes the extent to which vertices adjacent to any vertex v are adjacent to each other) = 0.4. Strong clustering means large number of triangular sub-graphs (>< regular tree structure).

8.5.2 Communication Cost Accounting

The communication cost accounts for the total number of messages exchanged in response to any non-local topological change. Those messages can carry out either topological and routing information, as in the SPT and ST cases, or information related with the MDT building process itself, as in the PPC case. Note that the accounting of the communication cost (C_c) is done per MDT and is defined as follows:

$$C_c = m_u + m_j + m_l + m_r \qquad (8.10)$$

where m_u represents the update routing messages (e.g. OSPF-TE Link State Update type), m_r those routing messages generated exclusively by the routing scheme (e.g. type-R and type-A messages), and $m_j; m_l$ those representing request messages either to join or leave an MDT, respectively. The join and release messages are shared by all the algorithms considered in this work.

Communication Cost of the Off-line computed ST

The C_c should account for the total number of exchanged messages during the dynamic tree building process, however the ST-MDT considered in this study is computed centrally in a static way (i.e. source-routing). In this case, a dynamic computation of the ST should be emulated and generating a sequence of join events. At each step of the MDT construction (i.e. at every join leaf node event), the novel nodes now part of the MDT will trigger the flooding of routing messages to announce their membership (a more elegant approach must be found to simulate the routing dissemination instead of

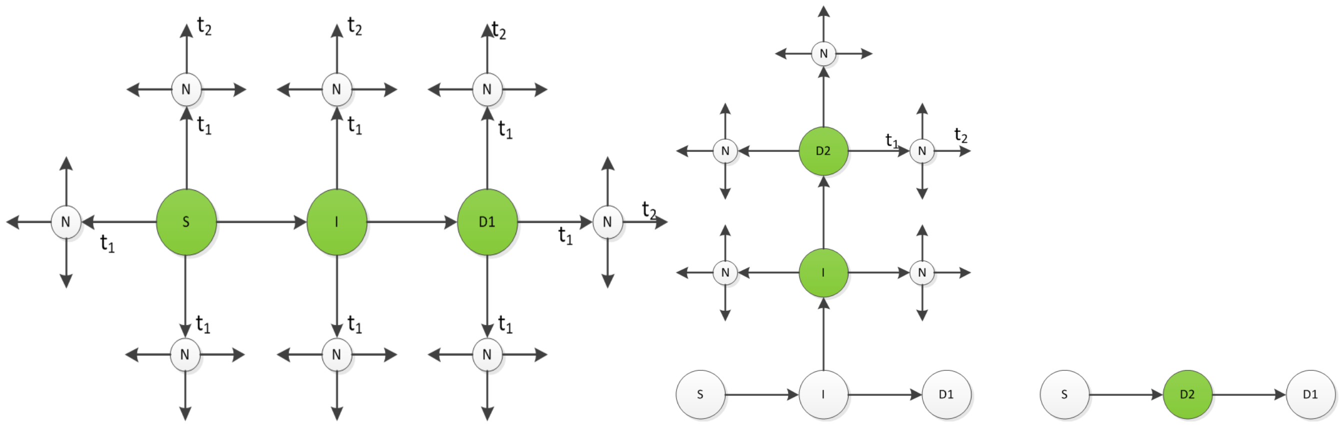


Figure 8.13: Examples of the communication measurement for the off-line ST case.

flooding). These messages contain the minimal information for remote nodes not (yet) belonging to the MDT to join it. The communication scheme is computed as follows:

1) Definition of an hypothetical join leaf node sequence for the computed ST from the multicast leaf set D.

2) For each leaf node $u \in D$, get the branching path along the MDT. All the nodes belonging to such path must inform the rest of the network that they are potential branching nodes to them, defining the set Q.

3) If a node $n \notin Q$, it generates such routing message and floods it.

Figure 8.13 depicts the communication process on three different occasions. First, a leaf node $D1$ joins the MDT through its multicast source s, $M = \{D_1\}$. In this case, each node of the new branching path $p(s, I, D_1)$ should spread out their membership to all the other nodes of the network (except to their neighboring nodes also part of the MDT). The same is done when leaf node D_2 decides instead leave the MDT (middle scheme). Two procedures are possible: i) D_2 announces is leaving and every remaining node of the MDT floods new information across the network or ii) every remaining node of the tree announces directly that D_2 is leaving at the same it announces its position. On the other hand, if the leaf node D_2 decides to join the MDT but it is also a node part of it, no messages are flooded.

8.5.3 Distance/Path-Vector Daemon

The flooding of routing messages with topology information (e.g. link-state OSPF protocol) is rather unrealistic on a flat network of more than 10k-nodes. Hence, we use a distance/path-vector scheme in the SPT routing case. In particular, a simple Distance Vector (DV) is implemented, augmented with the path to the multicast source (a Path vector(PV)), i.e., while a pure DV approach gives only the distance, an augmented approach (i.e. PV) also gives the path. The behavior is a simplified BGP routing update propagation with the AS-Path length as metric and selection criteria.

A split horizon dissemination of the distance/path is considered where the multicast source node starts the entire process. The multicast source node sends messages (announcing itself) throughout all its outgoing interfaces. Then, each one of neighboring nodes forwards the message (with the updated path) also throughout all its outgoing interfaces except the one from where it has received (the source) and so on.

In practice, each node maintains in addition reachability/path to each other (thus N-1 entry routing entry) (just imagine unicast traffic) but for multicast purposes only this is not the case. So, in a first step we will not count the $N-1$ entries to do the comparison, since we don't not account for "unicast" entries in our counting of RT entries. The total number of messages is equal to the

$$C_c = degree(source) + \sum_{i=0}^{N} (degree(node_i) - 1) \quad (8.11)$$

, or one could simply assume that the average node degree is equal to $2 * \frac{L}{N}$, which means your number of messages is proportional to L (the number of links). Here, there is accounting of differential delays (shorter paths having higher delays than longer paths due to the propagation and processing of messages, e.g., path $p(a, b, c)$ has shorter delay than path $p(a, c)$, which results into node c re-issuing another message telling that a shorter path is available. Future work should take this effect into account.

8.6 Results and Analysis

Large-scale Internet representative topologies, with node sizes = $10k$, $16k$ and $32k$, are used to assess the PPC scheme performance in a non-blocking dynamic scenario. Note, however, that we only depict and analyze here those most representative results, namely those referring to the 10k-nodes (with row search segmentation) and 32k-nodes (with optimized search segmentation) topologies. The rest of the simulation campaign can be found in Appendix.

First, a join-events only scenario is considered. The results obtained with the different techniques on the segmentation of the search space are also pointed out. Second, the initial results on the interleaved sequence of join/leave events scenario are presented and discussed. Finally, we present the study about the p2p paths of the PPC MDT and how stretch they are compared to the SP-Tree (SPT), if unicast would to be carried on it (following the AnyTraffic Labeled concept). This data set will be useful to set the basis of a future A-PPC scheme.

8.6.1 Join-only events Scenario

The performance analysis of the PPC algorithm starts by considering join-events only over time. In other words, once a leaf node joins a multicast session it will not leave until the multicast is complete.

The communication cost is a critical metric to determine the applicability of the proposed compact routing algorithm to large scale topologies (from 10k up to 32k nodes). The high communication cost obtained with PPC compared to the SPT, even if much lower than the communication cost implied by the ST, can be explained by the presence of high degree nodes in power law graphs (nodes that have a degree of the order of 100 or even higher). The mitigation techniques presented before (i.e., the optimized leaf and source vicinity balls) will help to keep it at acceptable values.

Note, however, that such computed communication cost does not take into account for the evolution of the routing topology. This evolution will impact severely in multicast routing algorithms such as the SPT that are strongly dependent on non-local unicast routing information compared to the proposed PPC algorithm. It is worth mentioning that the memory and the capacity required to process communication messages are relatively limited.

A. 10k-nodes GLP topology with (row) Search Segmentation

The search segmentation procedure is here applied resulting in a considerable reduction of the communication cost. Initially, a row leaf node ball dimensioning was implemented (following the f(n) function depicted in Fig.8.20). Latter on, we devised an optimized version, as described in subsection 8.3.3, which improved even more the communication cost, as we show in Point C. The maximum multicast leaf set is set to 2500 (with increments of 500 nodes). Each execution is performed 10 times by considering 10 different multicast sources.

Stretch Figure 8.14 illustrates the stretch ratio of the multicast routes (i.e. MDT) set up by the PPC and the SPT algorithms compared to the ST reference algorithm. The multiplicative stretch for the PPC is slightly higher than 1. Its trend curve decreases as the multicast group size increases (from 1.08 up to 1.04 for multicast group size ranging from 500 to 2500). In addition, it remains constant from group size of 2000 to 2500. Compared to the SPT, the PPC maintains a constant average gain of 6.5% along the different group sizes. Also, comparing it with the stretch values obtained before (point A) and shown in Fig.9.2 (top-left), a slight improvement is observed.

Another interesting observation is obtained by measuring the cumulative percentage of multicast routes in function of the stretch evolution. In Fig.8.15, at least 50% of the multicast routes created by the PPC have a stretch lower than the minimum stretch (1.04) reached for all the multicast group sizes. Except for the group size of 500 which has a maximum stretch of 1.08, the other group sizes lead to a maxi-

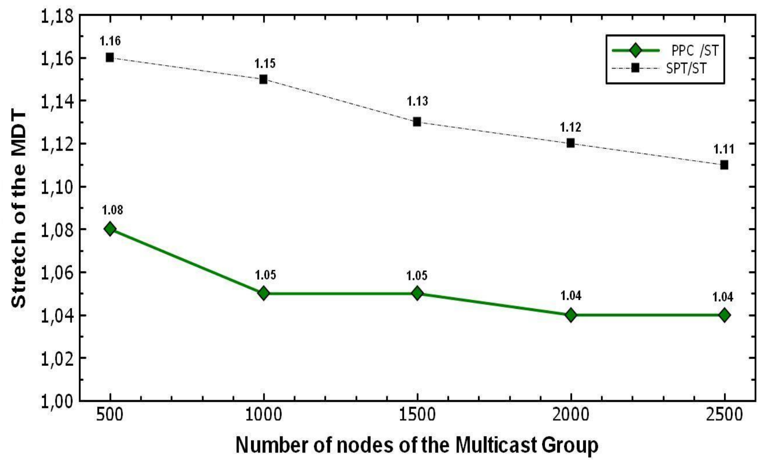


Figure 8.14: Stretch of the MDT as function of the number of nodes of the multicast group size.

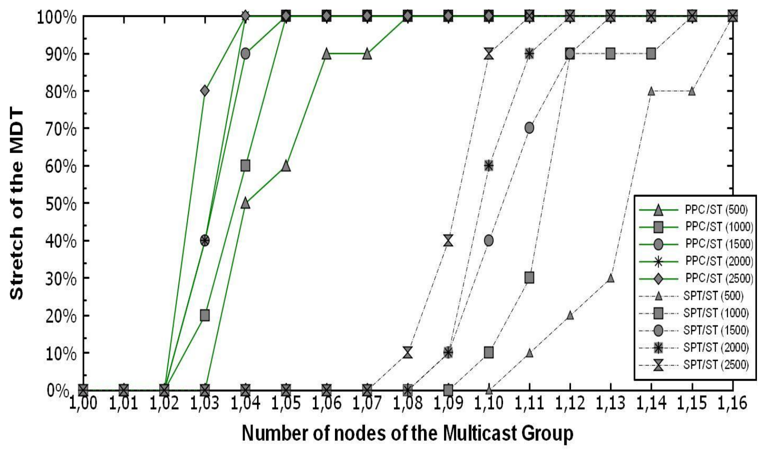


Figure 8.15: Cumulative percentage of multicast routes as a function of the stretch evolution.

mum stretch less than 1.05. As the multicast group size increases, the percentage of routing paths of lower stretch also increases. Compared to the SPT (right-hand side of Fig.8.15), for group sizes of 500 nodes, only 10% of the routing paths have a stretch equal or less than 1.11. For group sizes of 2500, only 10% of the multicast routes have a stretch equal or less than 1.08. All p2mp routing paths produced by the PPC lead to a maximum stretch of 1.08 independently of the multicast group size.

Storage The PPC algorithm shows outstanding performance in terms of the total number of RT entries it produces as shown in Table 8.6. The highest number of RT entries obtained for a multicast group size of 2500 ($5,805$ entries) is 2.8 times smaller than the number of RT entries produced by the ST algorithm ($15,642$ entries) and 15 times smaller than the number of the RT entries for the SPT algorithm ($87,036$ entries).

Figure 8.16 illustrates the relative gain in terms of the total number of RT entries produced by the PPC against the ST and SPT algorithms. An increasing gain as the multicast group size decreases can be observed. Moreover, as the size of the multicast group increases, both PPC and ST algorithms show a similar growing trend compared to the SPT algorithm.

Table 8.6: Number of RT entries for SPT, ST, and PPC with respect to the multicast group size.

Scheme	**Multicast Group Size**				
	500	1000	1500	2000	2500
SPT	82,393	83,656	84,837	85,955	87,036
ST	11,354	12,504	13,587	14,626	15,642
PPC	1,416	2,596	3,707	4,770	5,805

Figure 8.17 depicts the relative gain in terms of the memory-bit space consumed by the total number of RT entries produced by the PPC against the total number of RT entries pro-

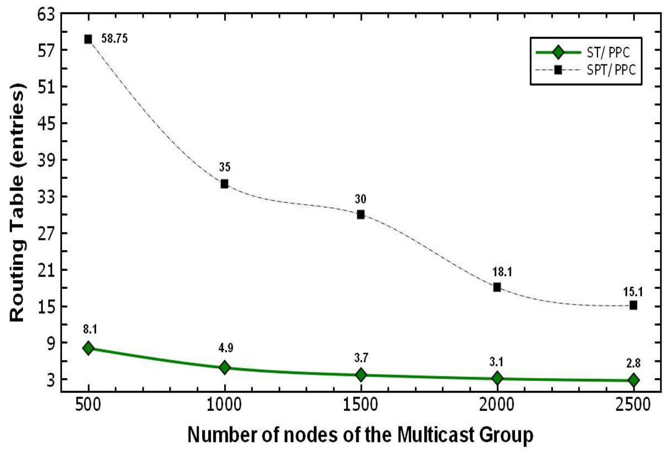


Figure 8.16: RT size ratio (in terms of number of RT entries) as function of the multicast group size.

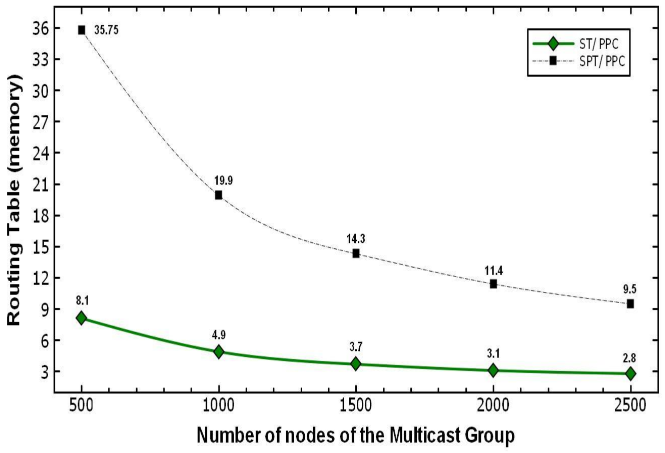


Figure 8.17: RT size ratio (in terms of memory-bits) as function of the multicast group size.

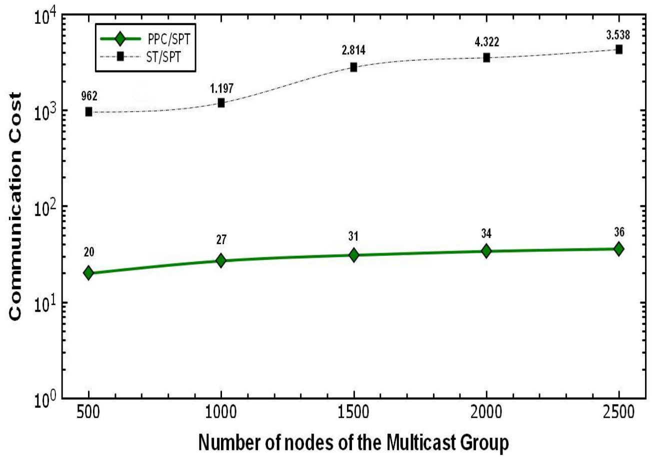


Figure 8.18: The communication cost ratio as function of the number of multicast group size

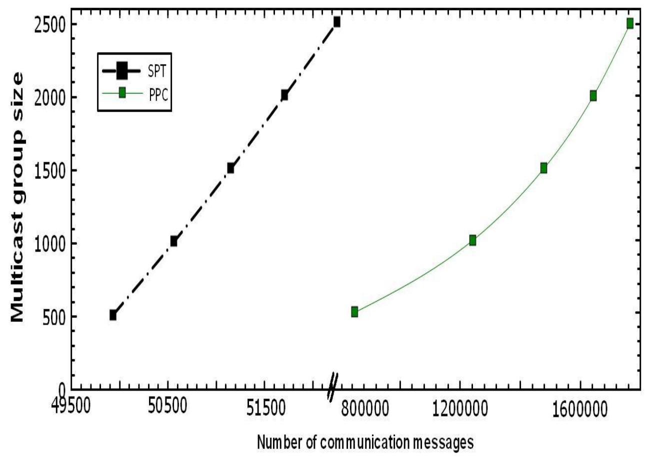


Figure 8.19: The multicast group size with respect to the number of communication messages

duced by the ST and SPT algorithms. As it can be observed, the relative memory gain of the PPC compared to the ST algorithm is never lower than 2.8 (for a multicast group size of 2500) and reaches a maximum of 8.1 as the multicast group size decreases to 500. The same trend is observed when comparing the PPC to the SPT algorithm, the relative memory gain ranges from 9.5 (for a group size of 2500) up to 35.75 (for a group size of 500). Despite of its better communication cost performance (as detailed in the next paragraph), the memory-space consumed and the number of RT entries produced by the SPT algorithm grows exponentially with the size of the multicast group. For the PPC, the curve grows sub-linearly: as the size of the multicast group increases the increment in number of RT entries becomes smaller.

Communication Cost The communication cost is a crucial metric to determine the applicability of the proposed algorithm to power-law topologies comprising of the order of 10k nodes. The two-stage search procedure presented in subsection 8.3.3 plays an important role in mitigating such cost. As depicted in Fig. 8.18, the communication cost ratio of the PPC is relatively high compared to the SPT even if much lower than the communication cost implied by the ST. As said before, this can be explained by the presence of high degree nodes (nodes that have a degree of the order to 100 or even higher) in power law graphs. However, recall that these communication cost values do not take into account for the evolution of the routing topology. This evolution will impact multicast routing algorithms such as the SPT that are strongly dependent on non-local unicast routing information compared to the PPC.

Between the PPC and the SPT algorithm, the difference of scale in terms of the number of messages exchanged can be observed from the curves of Fig.8.19. Despite their noticeable difference (maximum of 52,252 SPT-messages vs. 1,765,403 PPC-messages), these curves show that the communication cost for the SPT algorithm grows linearly with the multicast group size whereas the PPC has a concave curve, meaning a sub-linear dependence on the group size. Moreover, as depicted in Fig.8, the communication cost curve for the PPC decreases as the number of nodes composing the multicast group increases. This trend leads us to expect that a saturation level can be reached around a cost ratio not higher than 40 as the multicast group size continues to grow. It is worth mentioning that the memory-space and the processing capacity consumption by communication messages are relatively small.

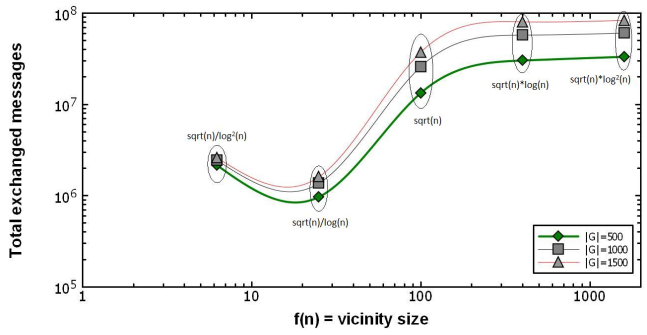


Figure 8.20: Number of exchanged messages according to the vicinity size (defined by local search stage), n=10k.

Note that a row leaf ball dimensioning is initially considered. As mentioned before, the vicinity size proportionally to $\frac{n^{0.5}}{log(n)}$ was thought to lead to the minimum number of exchanged messages and thus to the minimum communication cost, as shown in Fig.8.20.

B. 32k-nodes GLP topology and CAIDA Internet map with (optimized) Search Segmentation

Finally, the PPC is also simulated over $32k$-nodes topologies, getting closer to the real number of AS in the Internet, currently set in $37k$. Note that the search segmentation here used is enhanced with a new vicinity ball deployed around the source node, as explained in Subsection 8.3.4 and with a proper dimensioning of the leaf node vicinity ball too. This will help to achieve a greater reduction of the communication cost for such huge topologies, as we show in next Point C. The previous observed trends of a stable and low stretch and considerable RT size reductions, produced by the PPC, are maintained also here.

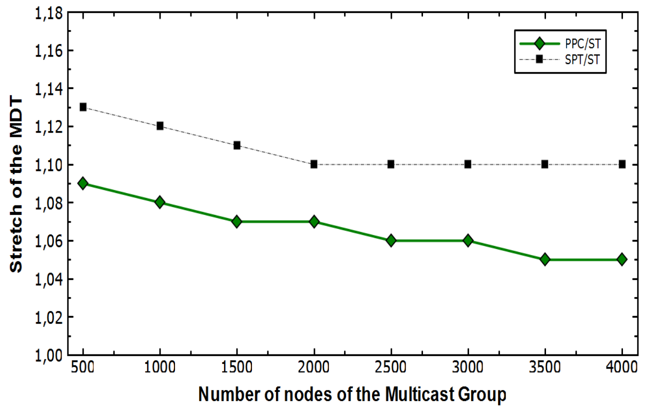


Figure 8.21: (GLP) Stretch as a function of Leaf Node Set Size.

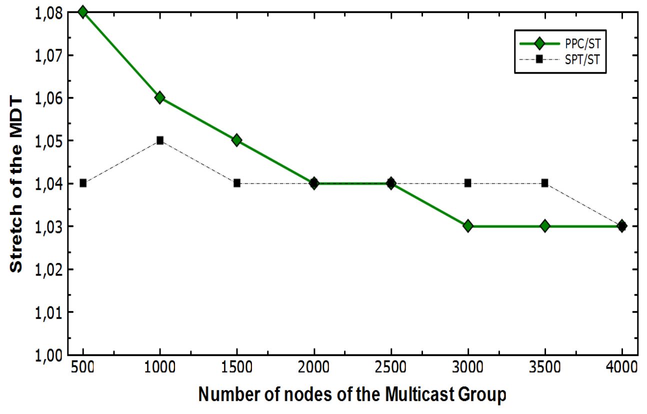


Figure 8.22: (CAIDA) Stretch as a function of Leaf Node Set Size.

Stretch Regarding the GLP Topology, the multiplicative stretch for the PPC algorithm is slightly higher than 1, as shown in Fig.8.21. As the leaf node set increases from 500 to 4000, its trend curve decreases from 1.09 (maximum value reached for 500 leaf nodes) to 1.05 (minimum value reached for 4000 leaf nodes). Compared to the SPT stretch, our algorithm maintains an average gain of 4% along the different group sizes. Regarding the CAIDA Map, the multiplicative stretch for the proposed algorithm is slightly higher than 1, as shown in Fig.8.22. As the leaf node set increases from 500 to 4000, its trend curve decreases from 1.08 (maximum value reached for 500 leaf nodes) to 1.03 (minimum value reached for 4000 leaf nodes). Compared to the SPT stretch, our algorithm maintains a maximum deterioration of 4% for sets of

500 leaf nodes; this deterioration becomes negligible as the size of the leaf node sets increases.

Storage This section details the simulation results obtained for the memory capacity required to store the RT entries (underlying the MDT) and the achievable reduction produced by proposed PPC scheme.

Table 8.7: GLP Topology: Number of RT entries for SPT, ST, and PPC with respect to the leaf node set size.

Leaf Node Set Size	**Routing Scheme**		
	SPT	ST	PPC
500	274555	33483	1646
1000	275927	34733	2986
1500	277261	35965	4272
2000	278561	37191	5554
2500	279827	38349	6756
3000	281045	39443	7874
3500	282265	40559	9022
4000	283477	41643	10154

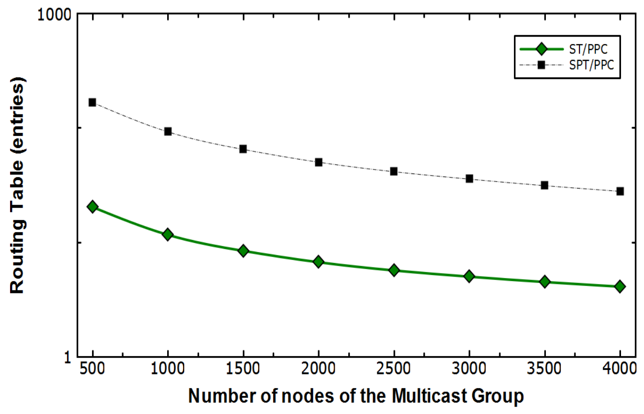


Figure 8.23: (GLP) RT Size Ratio as a function of Leaf Node Set Size.

From Table 8.7, we can observe that the PPC produces significantly less RT entries that the ST and SPT reference algorithms (GLP Topology). The highest number of RT entries is obtained for a set of 4000 leaf nodes: 10154 RT entries. This value is 4,10 times smaller than the number of RT entries produced by the ST algorithm (41643 RT entries) and 27,92 times smaller than the number of the RT entries produced by the SPT algorithm (283477 RT entries). Figure 8.23 illustrates the relative gain expressed in terms of the ratio between the total number of RT entries produced by the ST and the SPT references and our algorithm. An increasing gain can be observed as the size of the leaf node set decreases from 4.10 (leaf set of 4000 nodes) to 20.34 (leaf set of 500 nodes) compared to the ST algorithm and from 27.92 (leaf set of 4000 nodes) to 166.08 (leaf set of 500 nodes) compared to the SPT algorithm.

With respect to the CAIDA Map, the PPC also produces significantly less RT entries that the ST and SPT reference algorithms, as illustrated in Table 8.8. The highest number of RT entries is obtained for set of 4000 leaf nodes: 13169 RT entries. This value is 3.21 times smaller than the number of RT entries produced by the ST algorithm (42,277 RT entries) and 14.38 times smaller than the number of the RT entries produced by the SPT algorithm (189,431 RT entries).

Table 8.8: CAIDA Map: Number of RT entries for SPT, ST, and PPC with respect to the leaf node set size.

Leaf Node Set Size	**Routing Scheme**		
	SPT	ST	PPC
500	180993	34111	4919
1000	182339	35391	6237
1500	183609	36609	7471
2000	184825	37779	8653
2500	186009	38935	9807
3000	187151	40047	10921
3500	188325	41199	12059
4000	189431	42277	13169

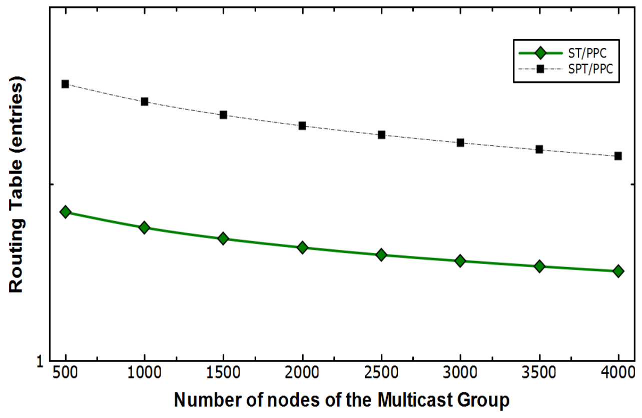


Figure 8.24: (CAIDA) RT Size Ratio as a function of Leaf Node Set Size

Figure 8.24 illustrates the relative gain expressed in terms of the ratio between the total number of RT entries produced by the ST and SPT references and PPC algorithm. An increasing gain can be observed as the size of the leaf node set decreases from 3,21 (leaf set of 4000 nodes) to 6,93 (leaf set of 500 nodes) compared to the ST algorithm and from 14,38 (leaf set of 4000 nodes) to 36,79 (leaf set of 500 nodes) compared to the SPT algorithm. Interestingly, the obtained gain values for the CAIDA map are smaller than those obtained for the GLP topology. This difference can be explained resulting from the difference in tree-depth: 6 (leaf set of 500 nodes) to 9 (leaf set of 4000 nodes) for the CAIDA map vs 8 (leaf set of 500 nodes) to 11 (leaf set of 4000 nodes) for the GLP topology.

Communication Cost The source node vicinity ball construction (defined in Subsection 8.3.4) has showed here to restrain efficiently the communication cost. Note that the communication cost deteriorates in function of the topology size.

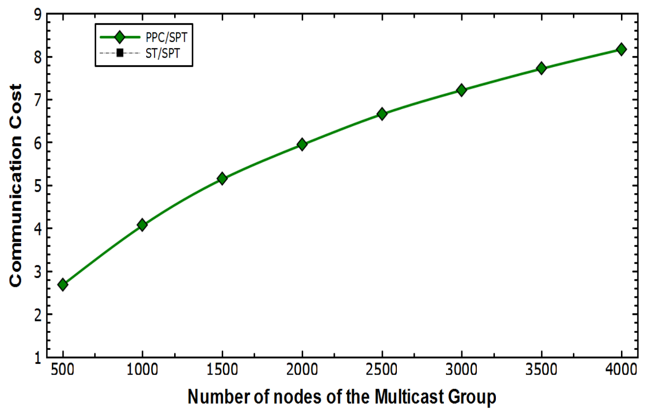


Figure 8.25: (GLP) Communication Cost Ratio as a function of Leaf Node Set Size.

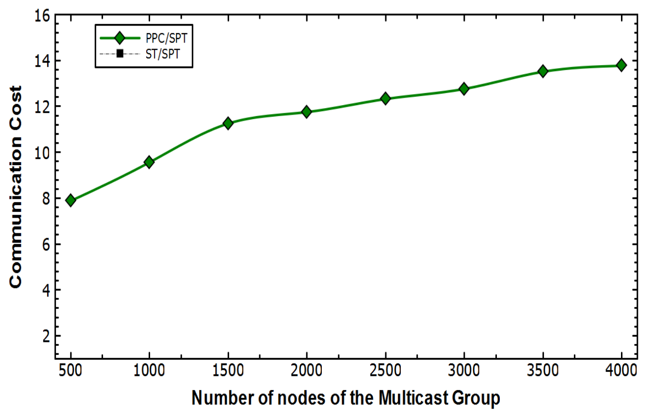


Figure 8.26: (CAIDA) Communication Cost Ratio as a function of Leaf Node Set Size.

As depicted in Fig.8.25, the communication cost ratio in the GLP Topology for the PPC algorithm is relatively high compared to the SPT even if much lower than the communication cost implied by the ST (not represented in this figure). Indeed, the communication cost ratio increases from 2.69 (leaf set of 500 nodes) to 8.17 (leaf set of 4000 nodes). Moreover, as shown in Figure 5, the communication cost of the proposed algorithm compared to the SPT communication cost, decreases as the number of nodes composing the leaf node set increases. This trend leads us to expect that a saturation level can be reached around a communication cost ratio not higher than 10 to 15 as the size of the lead node set continues to grow. The same trend can be observed for the CAIDA Map - Fig.8.26- where the communication cost ratio between PPC and SPT increases from 7.88 (leaf set of 500 nodes) to 13.77 (leaf set of 4000 nodes). The difference observed between the CAIDA map and the GLP topology can be explained from the following observation the tree-depth differs by a unit (3 vs 4). This difference induces a relatively higher cost of the SPT when running over the GLP topology.

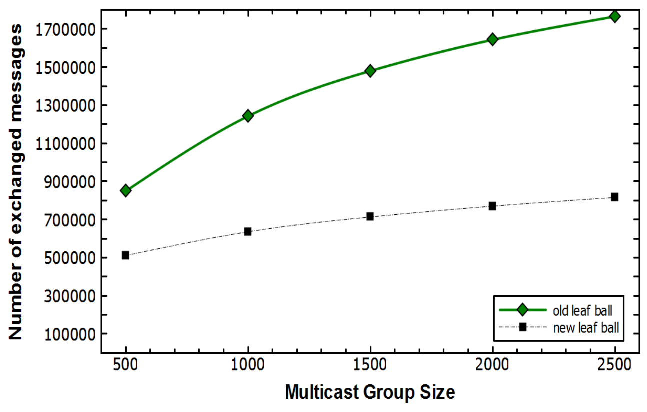


Figure 8.27: The gain obtained by using the new dimensioned Leaf node vicinity ball, $B(u)$.

C. The impact of the new dimensioned Leaf and Source Vicinity Balls

The objective is to highlight the improvement obtained in the communication cost due to i) the new dimensioning of the leaf node vicinity ball, Fig.8.27, and ii) the the source node vicinity ball, Fig.8.28 - (both considered in the 32-k nodes topology results - previous Point B). However, the results of this study refer to a 10k-nodes GLP topology using the same 10 different source nodes than before, as well as the same multicast groups. The gains with the new leaf ball go from 40% up to 54% as the leaf set in the MDT increases. With respect to the source ball, the gains go from 44% up to 61%, although in the inverse way as more leaf nodes in the MDT means a small difference in the number of exchanged messages.

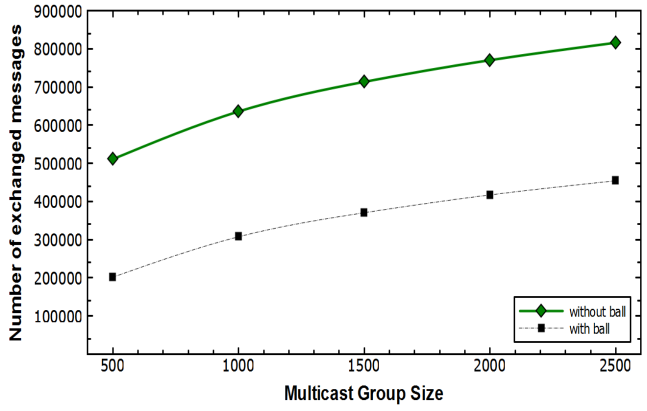


Figure 8.28: The gain obtained by deploying a vicinity ball also at the source node, $B(s)$ (the new dimensioning leaf's vicinity is considered).

For instance, note that the number of exchanged messages in the 32-k topology (Point B.) is lower than those exchanged for the 10k-nodes with row segmentation (Point A), even for the bigger multicast group sets.

D. Comparison with the Abraham Scheme

Here below, the performance in terms of the stretch of the p2mp routing paths produced and the memory space required by the proposed algorithm and by the Abraham routing scheme as specified in [Abraham09] (for dynamic join only events) are compared.

1) Stretch For the scheme allowing only dynamic join events, the MDT cost is given by Lemma7 of [Abraham09]. The authors determine that the proposed dynamic multicast algorithm is $O(min\{logn, log\Delta\}.logn)$ competitive compared to the cost of the optimal algorithm - Steiner Tree. In this formula, the factor Δ is the aspect ratio defined as the ratio between max $d(u,v)$ and min $d(u,v)$, for any $u, v \in V$. Considering an aspect ratio Δ of 6 and a network of $32k$ nodes the stretch is about 3.5. Thus the stretch upper bound of the p2mp routing path produced by the Abraham scheme, even if universal (applicable to any graph), is about 3 times higher than the one produced by our scheme.

2) Storage Following the description of the Abraham scheme provided in Section 8.1, the storage requirement is given by the memory space that includes i) the tree routing information $\mu(T,v)$ stored by each node v, for all trees in its own $SPLabel(v)$ leading to a total storage of $O(log^3n.log\Delta/loglogn)$ bits, ii) for each $i \in I$ and $T \in TC_{k,2^i}(G)$, the center node $c(T(v))$ of each node $v \in T$ that stores the labels of all nodes contained in the ball $B(v, 2^i)$ leading to a total storage over all radii of $O(kn^{1+1/k}log\Delta)$ bits; in addition, each node v stores $O(log\Delta)$ labels of size $(kn^{1/k})$ each leading to a total memory consumption of $(kn^{1+1/k})$ bits. The resulting memory storage requires about $700kbits$ for a tree comprising 4000 leaf nodes. For the same leaf set size, our routing scheme requires about $1250kbits$.

8.6.2 Join/Leave Events Scenario

The fully dynamic scenario consists in an interleaved finite sequence of join/leave events. At each stage j of the MDT construction, a leaf node u either joins or leaves it. Let D be the current MDT set of multicast targets (i.e. destination nodes) and M the set of all nodes that joined the MDT along the time. The performance metrics are now given respect to such stage j.

Interleaved Join/Leave Event Sequence

The interleaved sequence is modelled by a Zipf-law distribution, which is one of the discrete power law probability distributions. It is a good representation of the lifetime of each leaf node in a multicast session, giving us both the join event time and the duration of the leaf node in the MDT (i.e. the leave event time). Zipf's law predicts the frequency/probability of elements of rank k out of a population of N elements. In our case, N is either the period during which join events can occur or the lifetime of a leaf node in a multicast session. It is expressed like:

$$P(k) = \frac{\frac{1}{k^s}}{\sum_{n=1}^{N} \frac{1}{n^s}} \tag{8.12}$$

The ST computation of such sequence is done in blocks as the ST algorithm is computed in a centralized manner. Those blocks are determined by the leave events. For instance, if we have the following sequence: $< J_1, J_2, J_3, J_4, L_2, J_5, L_3 >$, we will have the following blocks to be computed: $< J_1, J_2, J_3, J_4 >$, $< J_1, J_3, J_4 >$, $< J_1, J_3, J_4, J_5 >$, and finally $< J_1, J_4, J_5 >$.

MDT Deterioration

Figures 8.29 and 8.30 show the stretch and the RT size ratio in terms of bit-space, respectively, at each stage of the construction of the MDT. The results are respect to just one multicast session, i.e., one MDT with $M = 100$ in the $1k$-nodes GLP topology.

The objective is to observe the deterioration in the MDT produced by PPC scheme due to the dynamics of the interleaved join/leave event sequence. In Fig.8.29, it is easily observed that a re-arrangement of the MDT (due to session lifetime) is required to keep the stretch closer to 1. The stretch for the SPT is not so high because it is oblivious algorithm (i.e. path routes are independent of each others) and thus the leave events do not have an impact so severe.

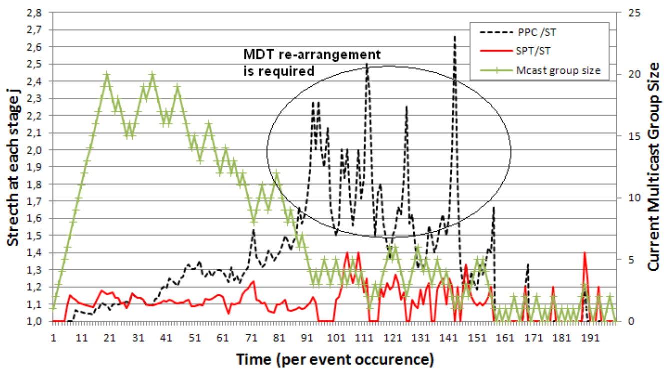


Figure 8.29: Stretch of the MDT at each stage j of the MDT construction.

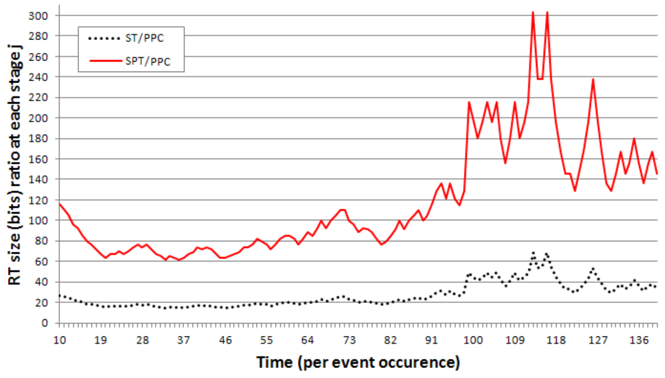


Figure 8.30: RT (bits) size ratio at each stage j of the MDT construction.

Nevertheless, incredibly high gains on the RT size are achieved. Compared to the ST, the PPC scheme is approx. 20 times better in terms of the RT ratio (in number of bits), while compared to the SPT, the gain is above 60 times. Even though the PCC scheme has worse stretch as leaf nodes start to leave the MDT, the RT ratio slightly increases for both the reference algorithms as the PCC scheme has a higher aggregation of paths (and thus of entries), albeit longer (and non-optimized). Note that we represent only the time sample between 10 and 142 in the Fig. 8.30 because the extremely high differences obtained in the edges would add too much graphical noise to the figures.

The communication cost is not illustrated due to the difficulty to measure it for the ST case. But it is expected to be extremely high as to achieve such optimal, minimum cost MDT.

8.6.3 A-PPC scheme: setting the basis

The preliminary evaluation of the name-independent A-PPC scheme consists in observe how stretched are the individual p2p paths inside the MDT of the PPC scheme (i.e. path between the leaf node u and the source node s within the MDT). The illustrated results refer to the maximum (representing the worst case) and the average p2p path cost within the MDT for 10 different sources. This is enough to give us an idea about how often the maximum deficit function (Exp. 7.3) of the AnyTraffic scheme is not accomplished and to help us to properly design the required adaptations. To make the observation easier, we express the maximum deficit function in terms of ratio (S_{Max}):

$$S_{Max} = \frac{(\Delta_{\max}^{s,u} - x_{s,u})}{x_{s,u}} = \frac{F(x)}{x_{s,u}} \qquad (8.13)$$

where $F(X)$ is the Exp.7.1 and $x_{s,d}$ the cost of the p2p path between the source s and the leaf u. First, the S_{Max} ratio is computed and then compared against the ratio S_U given by Exp.8.9. The comparison is given in terms of the difference in percentage between the S_U stretch and the S_{Max} ratio for each one of the 10 different sources considered in the study. Such comparison contributes to assess i) how many times the maximum deficit function is not accomplished (positive values) and ii) how far the p2p stretch of the MDT is from the maximum allowed.

Figures 8.33 and 8.32 show the comparison for the leaf set size of 1000 considering the maximum and average p2p path costs, respectively. In the average case, only in three MDTs the maximum deficit function is accomplished and in four out of other 7 MDTs the stretch is considerable high. In the maximum P2P path cost case (worst case scenario), all the MDTs have a maximum path whose stretch is in average around 50% than the maximum allowed by the AnyTraffic deficit function. Figures 8.33 and 8.34 show the same but for the leaf set size of 2000. In these cases, the scenario is even worse. In the average case, none of the MDTs meet the deficit condition and the values are high. In the maximum case, the same happens. For instance, there is one MDT that has a maximum path cost that almost doubles the maximum value allowed.

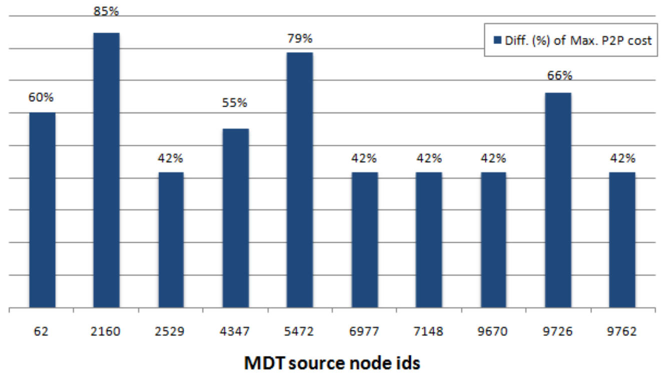


Figure 8.31: Difference in percentage between the S_U stretch and the S_{Max} ratio considering the **Max.** p2p cost of the MDTs (group size = 1000).

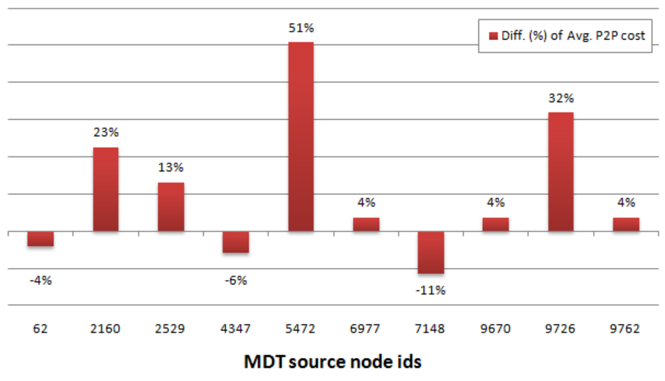


Figure 8.32: Difference in percentage between the S_U stretch and the S_{Max} ratio considering the **Avg.** p2p cost of the MDTs (group size = 1000).

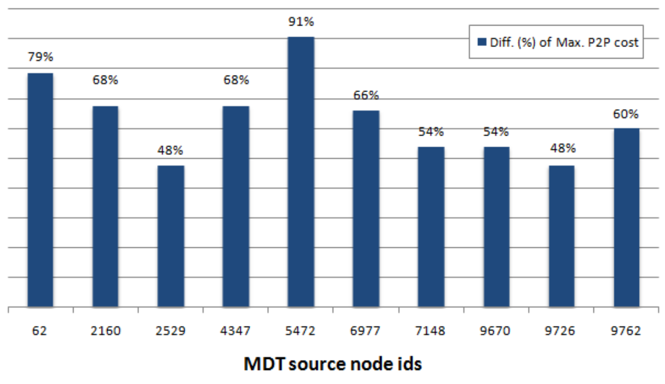


Figure 8.33: Difference in percentage between the S_U stretch and the S_{Max} ratio considering the **Max.** p2p cost of the MDTs (group size = 2000).

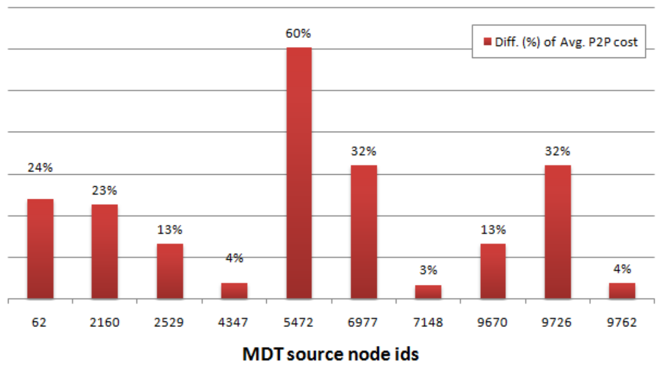


Figure 8.34: Difference in percentage between the S_U stretch and the S_{Max} ratio considering the **Avg.** p2p cost of the MDTs (group size = 2000).

8.7 Summary

In this Section, we introduce the first known name-independent compact multicast routing algorithm (named here PPC), enabling the leaf-initiated, distributed and dynamic construction of an MDT. The performance obtained shows substantial gain compared to the ST in terms of the RT entries and memory space required to store them: minimum factor of 2.8|3.21 for sets of 2500|4000 leaf nodes, i.e., 25%|12.5% of the topology size with respect to the 10k and 32k nodes topologies, respectively. The stretch deterioration compared to the ST ranges from 8% to 4% (3% for the 32k topology), for multicast group size of 500 to 2500 (again 4000 for the 32-k topology), respectively; thus, decreasing with increasing group sizes. It is worth to mention that similar trends were observed along the different topologies used in the simulation campaign (see Appendix), reinforcing the quality and strength of the proposed algorithm.

Regarding the communication cost, the proposed search space segmentation to the PPC algorithm -local search first covering the leaf's node vicinity, and if unsuccessful, a global search over the remaining topology- enables to keep such cost within reasonable bounds compared to the reference SPT scheme and sub-linearly proportional to the multicast group size. Moreover, the optimized leaf's ball dimensioning together with the new vicinity ball of the source node, contribute strongly to improve even more its communication cost (gains ranging from 40% to 54% with the new leaf's ball and from 61% to 44% with the source's ball).

The first assessment of the impact of an interleaved join/leave event sequence in the PPC scheme was also performed, showing that re-arrangement policies and mechanisms to cope with the limited session lifetime of each participant are strictly needed to maintain the PPC scheme stretch as low as possible. However, the huge gains observed respect to the RT size, will give us margin to work it out (i.e., trading RT size gains per lower stretch).

The first approach to a compact AnyTraffic routing, named A-PPC, was equally performed. Although the achieved MDT stretch values of PPC (fully oriented to multicast forwarding) are quite good, it was observed that the majority of the individual p2p path of the PPC derived MDT do not accomplish the AnyTraffic premise (i.e., they are too stretched to cope with unicast traffic requirements), which means AnyTraffic-related functions should be adapted to be then applied to such distributed environment. The challenge however is precisely how to deploy them in such a distributed environment.

Further work will be nevertheless conducted to further design and deploy the mentioned MDT re-arrangement techniques in order that such promising results achieved here can be also verified for dynamic sequences of node join and node leave events and non-stationary topologies. In the same way, efforts will be done to implement the A-PPC scheme and assess the PPC performance on Future Internet topologies with expectable sizes in order of the 100k-nodes.

Summary

Part II introduces two breakthrough paradigms on the design of a routing system scalable with the Future Internet requirements, namely the AnyTraffic Labeled Routing and the Name-Independent Compact Multicast Routing (i.e PPC).

In a way to address the current routing scalability problem and the fundamental limits of current stretch-1 shortest-path routing, we investigate the trade-offs between RT size (to enhance scalability itself), routing scheme stretch (to ensure routing quality) and communication cost (to efficiently and timely react to various failures).

The AnyTraffic Concept proposes a creative model where a single routing entity (i.e., a tree) is used to forward both type of traffic (unicast and multicast) navigating on today's networks. In other words, a unique forwarding entry at each node is shared. The savings are huge at the expense of a small increase in the bandwidth consumption (mainly due to resulting longer routing paths). However, it requires full topology view as it computes in a centralized way.

Focusing on a distributed approach, we come out with the PPC compact multicast routing, which is dynamic, leaf-initiated (distributed) and name-independent (i.e. topology unaware). The challenge remains on performing efficiently under dynamic conditions (topology and policy dynamics) and limit the number of messages exchanged in order to react efficiently and timely to those changes. Very promising results were achieved indeed. Internet representative large-scale topologies are used to assess the algorithm.

A first study of the behavior of the PPC in a fully dynamic scenario (where leaf nodes join and leave the MDT) was also considered. As expected, we concluded that MDT re-arrangement mechanism must be deployed in order to keep the stretch under acceptable values as the PPC algorithm is not oblivious. Also, as a natural step of convergence, we launch the first basis for an AnyTraffic PPC (or A-PPC). Although the PPC MDT is efficient for multicast traffic, it produces sometimes too long paths if unicast is to be forwarded through such entity. A strategy must be defined to endow each leaf node with the SP referential in order to process the AnyTraffic deficit function. Again, we assess our outcome according to the following methodology (SWOT):

Strengths

· Up to 8 times smaller RT sizes compared to state-of-the-art.

· Low stretch and bound communication cost.

· Protection against topological changes (as it avoids the dissemination of such information) - PPC.

· No off-line partition of the network is required.

Weakness

· AnyTraffic is centralized and requires full topology view.

· PPC is not oblivious. Extra MDT re-arrangement techniques are required.

Opportunities

· Shared and multi-source MDTs.

· Huge potential in joining AnyTraffic and PPC.

· Investigate on better alternative deficit functions to AnyTraffic.

· PPC reduced number of exchanged messages due to topology changes compared with state-of-the-art.

Threats

· The number of exchanged messages related with the discovery MDT procedure as well as with the MDT re-arrangement.

Part III

Conclusions and Future Work

Chapter 9

Concluding Remarks and Future Work

The popularization of Internet has turned the telecom world upside down over the last two decades. Network operators, vendors and service provides are being challenged to adapt themselves to Internet requirements in a way to properly serve the huge number of demanding users (residential and business). The Internet (data-oriented network) is supported by an IP packet-switched architecture over a circuit-switched, optical-based architecture (voice-oriented network), which results in a complex and rather costly infrastructure to the transport of IP traffic (the dominant traffic nowadays).

In such a way, a simple and IP-adapted architecture is desired (IP-over-WDM). New requirements in terms of bandwidth utilization and QoS provisioning are in the horizon of both the transport network, which needs to satisfy such upper (IP) packet-oriented infrastructure and services, and the IP network itself, as it needs to cope with such huge number of client applications and its dynamics efficiently.

This being said, this work aims to a Future Optical Internet architecture. In particular, it aims to an IP over optical burst switching (OBS) scenario, addressing both its transport network (Part I) and IP network (Part II) problems, namely the Generalized Multi-Protocol Label Switching (GMPLS) operability as control plane for the OBS network and the inter-domain routing scalability problem in the IP network (Internet).

PART I: GMPLS-OBS Interoperability Design

The IP-over-WDM architecture is desired to the future optical Internet. Two of its main requirements are i) a control plane providing protection and restoration mechanisms as well as automatic resource and connection management as defined by ASON paradigm; and ii) an all-optical switching layer oriented to packet transport, eliminating redundancy in IP transport and reducing operational costs. Both GMPLS and OBS technologies are part of that set of solutions, providing intelligence in the control and management of resources (GMPLS) as well as a good network resource access and usage (OBS). So far, they cannot coexist as GMPLS is not adapted to handle OBS.

Despite the GMPLS principle of unified control, little effort has been put on extending it to incorporate the OBS technology and many open questions still remain. The GMPLS and OBS interoperability design has been dragging on for quite a long time. The first general attempt is dated from the year 2000 and since then no major work has ever come up with significant neither functional or operational proposals, nor performance analysis.

Architecture, Model and Protocols

We have proposed a fully collaborative GMPLS-OBS control architecture and model, preserving both technology's identities. We maintain GMPLS technology-independent as the interoperability with OBS and further extensions do not compromise the overall GMPLS applicability to other switching technologies, while keeping the most distinctive OBS property of statistical multiplexing (sharing of resources by different traffic flows).

From a functional perspective, the proposed architecture answers to almost all the interoperability technology-specific issues. The signalling mismatching, as it may be the major barrier in the whole process, is successfully overcome and the two-way GMPLS and the one-way OBS signalling paradigms can efficiently coexist. The reservation of resources is kept at the OBS signalling time (as in any conventional OBS network) while a virtualization of the resource allocation is done at the GMPLS signalling time aiming at absolute QoS figures. The idea behind is to rely on GMPLS to perform all those complex control tasks (being provided so far at the OBS switching layer as well as adding some missing ones), and maintain OBS as simple as possible to perform fast switching.

A network model is described where this architecture allows not only a vertical interoperability (GMPLS-OBS) but also an horizontal interoperability in a way that a single GMPLS control instance can handle multiple switching domains even if OBS is present (the MRN/MLN requirements [RFC5212] are accomplished).

From an operational perspective, the control model offers (absolute) QoS guarantees overcoming OBS performance issues by making use of the GMPLS traffic-engineering (TE) features. A control task assignment (sharing of control between GMPLS and OBS control layers under an interoperable control plane) is proposed based on time requirements. Those time constrained operations are left to OBS such as BCP processing, reservation of resources for the burst duration, while the others (not time constrained) are left to GMPLS such as b-LSP setup and tear-down and network discovery.

The setup latency of the two-way GMPLS signalling process is shown to be residual compared to the b-LSP average duration. A VT, as a set of several b-LSPs, is modelled to provide absolute QoS figures and independent routing policies and RWA algorithms are proposed to achieve it.

Keys extensions to the GMPLS protocol standards are equally approached and suggested to cope with such architecture. An OBS-specific ISC is proposed to properly integrate OBS in the GMPLS framework, disabling the resource reservation at the GMPLS signalling time. A new LSP region was created due to the novel b-LSP object which is defined to accommodate such burst switching requirements at the GMPLS level. Minor extensions are proposed both to the GMPLS Signalling as its messages should carry new parameters at the T_SPEc object, the definition of a new label meaning (no label swapping is done) and format and to the GMPLS routing as its messages should carry those OBS-specific parameters across the network in order to update the node's databases.

Absolute QoS-provisioning in OBS networks making use of GMPLS TE features

The concept of virtualization and, in particular, the joint problem of routing (route of b-LSPs) and WA (adequate allocation of wavelengths on the links belonging to those b-LSPs) with e2e QoS guarantees, is a novelty and is successfully applied here, providing OBS with absolute QoS figures and improving its overall loss performance. This optimized resource usage releases more space to route BE traffic and consequently improving its loss figures too.

Several numerical simulations, run at different network topologies and (static and dynamic) traffic scenarios, show the feasibility and reliability of our proposals. Besides the accomplishment of any QoS level requested by quality-demanding (HP) traffic, the model also achieves a constant behavior showing independence of the network load. Compared with the Dynamic Wavelength Group technique (DWG), a reference approach in the literature providing absolute QoS in OBS networks, our model surpass their performance as we cope with more loaded network scenarios and achieve better BE traffic losses.

We have also deployed a GMPLS-driven mechanism to reconfigure the b-LSP capacity when unexpected traffic variations occur. Even though the VT is dimensioning to guarantee the QoS level, these short-duration variations may happen and can be handled without the reconfiguration of the VT but only through minor local adjustments.

Future steps

- ✓ Proof-of-concept with the implementation of such control architecture and model in a network test-bed.
- ✓ Study of new OBS loss model in the VT modelling, allowing the sharing of resource among different QoS classes.
- ✓ Study of new OBS loss model in the VT modelling (e.g. Engset instead of Erlang) assuming a different traffic distribution (e.g. Pareto instead of Poisson).
- ✓ Run the model in a scenario with failures (links, OBS controller, GMPLS controller).

✓ Study the impact of having a different topology to GMPLS and a 1:N GMPLS-OBS correlation.

PART II: Future Internet Routing Scalability

The Internet routing system of large backbone operators is facing scalability problems. Millions of devices and service instances are demanding fast network convergence times upon (topological or policies) changes, mobility and multihoming, and high quality content distribution (QoS and QoE) and the current Internet features are not prepared to accommodate such high volumes and dynamics. The growth of ASes is not expected to stop (100k to be reached in the next years), which consequently leads to an unbearable growth of the routing tables too. The inter-domain routing system is already consuming a huge memory-space at the nodes. The foreseen RT size values are not scalable with the size of the future Internet topologies.

In addition, as multimedia content/stream is proliferating, the problem becomes even worse as more entries need to be stored. Multicast routing, which is catching up again as a bandwidth efficient technique, still suffers from older issues as they still lay on underlying unicast routing protocols and full topology view. Overlaying multicast routing on top of unicast suffers from the same scaling limitations as current unicast routing with the addition of the level of indirection added by the multicast routing application.

New routing paradigms

The challenge remains on the definition of new routing paradigms so as to design, develop, and validate distributed and dynamic routing schemes suitable for the future Internet and its evolution. We address the root causes (disruptive attitude) instead of short-term incremental fixes to attenuate symptoms (evolutionary attitude), achieving outstanding results to the triple trade-off: stretch (routing quality) x RT size scale (sub-linear growth) x dynamics (limit the routing information exchanges).

AnyTraffic Concept

One way of saving on routing entries could be by defying the forwarding paradigms and schemes deployed in today's packet-switched networks. In such a way, we propose a routing scheme that computes paths along which combined unicast and multicast traffic can be forwarded altogether, i.e., over the same path, aiming at state (forwarding entries) consumption reduction. In this case, a single forwarding entry is shared to forward both type of traffics. For this purpose, the concept of AnyTraffic group is introduced that defines a set of nodes capable to process both unicast and multicast traffic received from the same (AnyTraffic) tree. The resulting scheme is referred to as AnyTraffic routing.

This research work comprises the definition of an heuristic algorithm to accommodate such AnyTraffic group and to find the proper set of branch nodes of the tree, as well as to control the stretch of the branching paths (bounding the damage implied to P2P data paths of unicast traffic). It settles a minimum bound on the state consumption and a higher bound on bandwidth utilization increase. The algorithm supports dynamic changes of the leaf node set during multicast session lifetime by adapting the corresponding tree upon deterioration threshold detection.

An extensive simulation campaign was performed considering both static and dynamic traffic scenarios to i) determine the dependencies of the algorithm (node degree, clustering coefficient and group size); and ii) evaluate its performance under dynamic conditions. The results obtained show that the AnyTraffic algorithm can successfully handle dynamic requests while achieving considerable reduction of forwarding state consumption with small increase in bandwidth utilization compared to the Steiner Tree algorithm.

Name-independent Compact Multicast Routing

Compact routing aims to overcome the current poor scaling properties of Internet features. It addresses the fundamental tradeoff between the memory-space required to store the routing table entries and the length of the routing paths that these schemes produce. In other words, a routing scheme is COMPACT if it is memory efficient. Its goodness is measured by its STRETCH. The main goal is to minimize the size of the routing table at each node.

So far, we are at an apparent impasse as current static, name-dependent (i.e. topology aware) compact routing algorithm does not cope with the desired level of scalability to future Internet and name-independent compact routing algorithms do not perform better than name-dependent in Internet-like topologies so far. In the context of multicast routing, a first compact multicast routing attempt has been recently proposed [Abraham09], in 2009. However, it is a source-routed, name-dependent compact scheme, postponing the desired routing name-independency.

This research work introduces the first known name-independent compact multicast routing (PPC) algorithm enabling the leaf-initiated, distributed and dynamic construction of p2mp routing paths from any source to any set of destinations (or leaves). A radical approach was taken, as no topology information is assumed to be kept at the nodes besides its local knowledge about its direct neighbors. In such a way, a discovery procedure was entirely define as to allow a leaf node to join a multicast distribution tree. Due to the high communication cost that it implies, we have successfully deployed a segmentation of the search space, mitigating the number of messages exchanged to build the MDT.

The PPC algorithm was simulated on synthetic power law graphs modeling (comprising both 10k and 16k nodes) the Internet topology and the CAIDA map of the Internet topology (comprising 16k nodes). Simulation results confirm that the PPC scheme can provide a suitable algorithmic basis for balanced stretch and memory space consumption. Substantial gains in terms of the RT entries and memory space required to store them are obtained. Compared to the SPT, the gain in memory space consumption results from the elimination of the underlying unicast RT entries whereas, compared to the ST, this gain is mainly due to the elimination of the RT entries required at each step of the routing path construction. The proposed two-phase search process keeps the PPC communication cost within reasonable bounds compared to the reference SPT scheme and sub-linearly proportional to the multicast group size.

Future steps:

This research topic is very promising as it addresses the routing scalability problem in the Internet. Below, we give some points of research to be continued:

- ✓ Define MDT Re-arrangement policies and proceed to its implementation.
- ✓ Complete comparison with the Abraham scheme fully dynamic scheme (numerical and theoretical).
- ✓ Adapt the proposed algorithm to cope with shared MDT and multi-source scenarios.
- ✓ Run the algorithms in non-equal link cost scenarios of large-scale topology and assess the algorithm behavior.
- ✓ Detail on the proposal of name-independent compact AnyTraffic routing.
- ✓ Investigate on other AnyTraffic maximum deficit functions.

Appendix

A.1 Network Topologies

Figure 9.1 illustrates the two different optical network topologies considered in the (G)OBS study, namely the 14-nodes NSF and the 28-nodes EON networks.

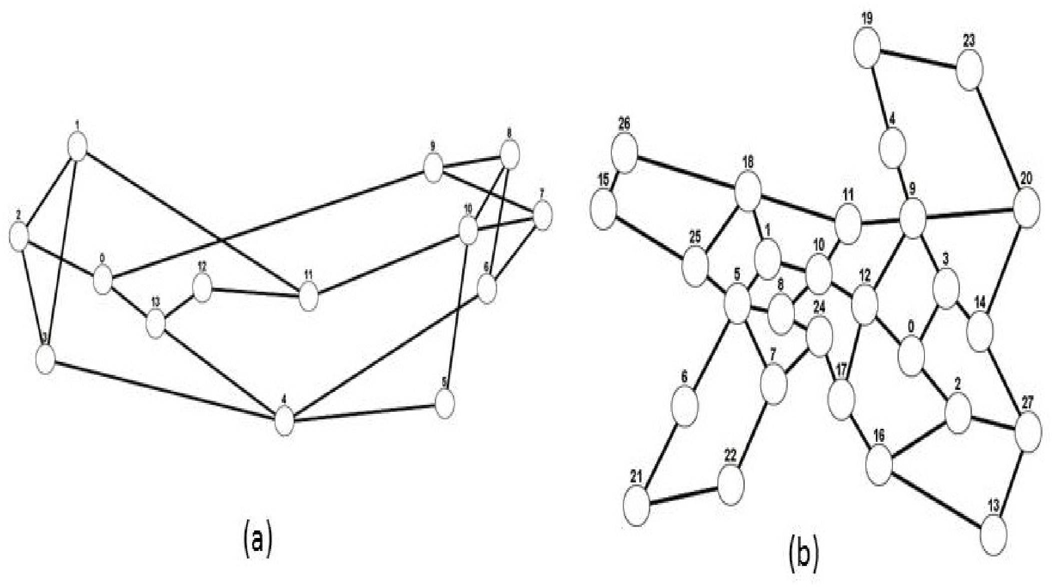


Figure 9.1: Optical Network topologies.

A.2 Benchmark Routing Algorithms

We consider the Shortest-Path Tree (SPT) and the Steiner Tree (ST) algorithms to benchmark the proposed algorithm. SPT is the best way to construct optimal source-based distribution trees: optimal path cost but higher resource consumption. Thus, it provides the communication cost reference. On the other hand, ST is the best way to create optimal shared distribution trees, where we consider an optimize algorithm based on shortest path distances (i.e. ST algorithm). It is the reference in terms of stretch.

In order to obtain the near optimal solution for the ST, we consider a ST-Integer Linear Programming formulation. For this purpose, we have adapted the formulation provided in [SAGE] to be computed on bi-directional graphs.

Shortest-Path Tree Algorithm

The SPT is a connected subgraph without cycles (i.e. a tree) of a given weighted graph so that the distance/cost between a selected source node and any other node of a multicast group, g, is minimal. It is rooted at the multicast source node, s. The Dijkstra's algorithm is used to compute the SPT, from a given vertex.

Whenever a node wants to join the SPT, it sends a $< s, g >$ join message out on the proper interface towards the source node for that group using the proper multicast protocol to inform the upstream node that it wishes to join the distribution tree for that group. The upstream node receiving such message adds the interface on which the message was received and then sends a (S,G) join message out the interface towards the source. The process is repeated in every subsequent node, building the SPT as it goes. The process stops when the join message reaches i) the multicast source node or ii) a node that already has multicast forwarding state for this source-group pair. In either case, the branch is created and each of the nodes has multicast forwarding state for the source-group pair, and packets can flow down the distribution tree from source to receiver.

Steiner Tree Algorithm

The Steiner problem asks for a shortest network which spans a given set of points. Minimum spanning networks have been well-studied when all connections are required to be between the given points. More information about ST heuristics can be found in [Hwang92].

Minimum-Path Heuristic

The computation of a minimum-cost Steiner tree is a NP-complete problem [Garey77]. As a first step to approach our problem, we have implemented the minimum cost path heuristic algorithm (MPH) to compute a minimum-cost Steiner tree for a multicast connection. In MPH, starting from a source node, the tree is gradually grown until it spans all destination nodes belonging to a multicast group. The growth is usually based on the addition of shortest paths between destination nodes already in the tree and destination nodes not yet in the tree. A full description of the heuristic algorithm can be found in [Takahashi80].

Integer Linear Programming

In order to obtain the near optimal solution for ST algorithm, we consider a ST-Integer Linear Programming formulation. We here adapt the formulation given in [SAGE] to be computed on a bi-directional graph. It is defined as follows:

Given a graph G, a cost function $c : E(G) \to \mathbb{R}$ and a set M of vertices, we want to find an acyclic sub-graph of minimum cost, T, linking all them together. This sub-graph T of G has $\mathcal{V} = |V(T)|$ vertices and $\mathcal{E} = |E(T)| = |V(T)| - 1$ edges and contains each vertex from M. Note that E is a set of bi-directional edges. For such reason, we set e_i and e_{out} as any incoming edge and outgoing edge, respectively.

Notation:
$\mathcal{E}$ = edges
$\mathcal{V}$ = vertices
x_e = binary variable indicating if $e \in$ sub-graph T
x_v = binary variable indicating if vertex $v \in T$
c_e = cost of an edge $e \in \mathcal{E}$

minimize $\quad U = \sum_{e \in \mathcal{E}} c_e . x_e$ (ILP)

subject to (9.1a)

$$\sum_{\substack{e_{out} \in \mathcal{E} \\ e \sim v}} x_e \geq 1, \quad \forall v \in \mathcal{M}, \tag{9.1b}$$

$$\sum_{e_{in} \in \mathcal{E}} x_e \leq 0, \quad v = s, \tag{9.1c}$$

$$\sum_{\substack{e_{in} \in \mathcal{E} \\ e_{in} \sim v}} x_e \leq 1, \quad \forall v \in \mathcal{V}, \forall e_{in} \in \mathcal{E}, \tag{9.1d}$$

$$x_e \leq x_v, \quad e \sim v, \forall v \in \mathcal{V}, \forall e \in \mathcal{E}, \tag{9.1e}$$

$$\sum_{\substack{e_{out} \in \mathcal{E} \\ e \sim v}} \leq C. \left(x_v + \sum_{\substack{e_{in} \in \mathcal{E} \\ e \sim v}} \right), \quad \forall v \in \mathcal{V} \tag{9.1f}$$

$$\sum_{\substack{e_{out} \in \mathcal{E} \\ e \sim v}} \geq -C. \left((1-) + \left(1 - \sum_{\substack{e_{in} \in \mathcal{E} \\ e \sim v}} \right) \right) + 1, \quad \forall v \in \mathcal{V} \tag{9.1g}$$

$$\sum_{v \in \mathcal{V}} x_v - \sum_{e \in \mathcal{E}} x_e = 1, \tag{9.1h}$$

$$\boldsymbol{x_e} \in \{0,1\}^{\mathcal{E}}, \quad \forall e \in \mathcal{E}, \tag{9.1i}$$

$$\boldsymbol{x_v} \in \{0,1\}^{\mathcal{V}}, \quad \forall v \in \mathcal{V}, \tag{9.1j}$$

The objective of the optimization problem ILP is to minimize the total number of links used to connect all the vertices in M. (9.1b) gives that each node of the multicast group has to have at least one of its links, either incoming or outgoing, as part of the final sub-graph; (9.1c) the source node can not have any of its incoming links as part of the final sub-graph; (9.1d) only one incoming link of the node can be part of the final sub-graph; (9.1e) means that if a link is part of the sub-graph so it is the node; (9.1f) and (9.1g) guarantee that if a node is not part of the multicast group and it has one incoming link, it must have at least one outgoing link as part of the sub-graph too; (9.1h) says that the total number of nodes is equal to the number of links plus 1.

ILP Time Complexity: $\quad O(|M| + N.V + 1 + V + N) = O(|M| + (1 + V)(N + 1))$

A.3 PPC Scheme - Results on 10k and 16k-nodes topologies

1) 10k-nodes GLP topology without Search Segmentation

The first analyze of the PPC algorithm performance refers to its execution on a large-scale topology of $10k$ nodes. The execution scenario considers the construction of p2mp routing paths for multicast leaf set of increasing size from 500 to 2000 nodes (selected randomly) with increment of 500 nodes. Note that the search segmentation is not applied here.

Stretch Fig.9.2 (top-left) depicts the stretch ratio between the PPC and the ST algorithms, as well as the ratio between the SPT and the ST. Results show that the PPC reaches slightly higher than 1 multiplicative stretch (from 1.09 to 1.06 for multicast group sizes ranging from 500 to 2000 nodes) and a gain of at least 5% compared to the SPT (multicast group size of 500 nodes) that remains approximately constant with increasing group size.

Storage As shown in Fig.9.2 (top-right), the gain in terms of the number of RT obtained for the ST compared to the PPC decreases from 8.8 to 3.4 when the multicast group size increases from 500 to 2000. The same trend is observed when comparing the SPT to the PPC. As depicted in Fig.9.2 (bottom-right), the gain in terms of memory space consumption obtained for the ST compared to the PPC varies from 10.1 to 3.8 when the multicast group size increases from 500 to 2000. This gain decreases from about 8.8 to 3.4 when comparing the SPT to the PPC. In both figures, even if the gain decreases with increasing multicast group size, the results show significant benefit when the multicast group size remains relatively small.

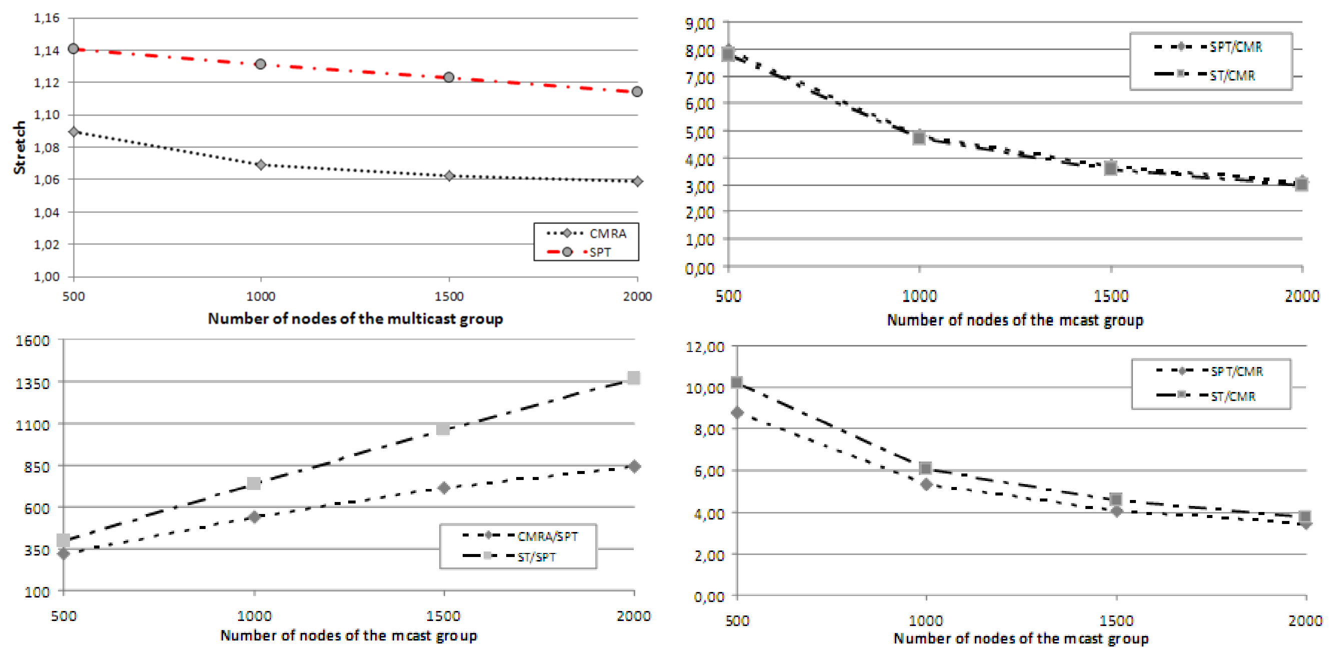


Figure 9.2: Left-column: Stretch (top) and Communication Cost (bottom) ratios. Right-column: RT size ratio in terms of number of entries (top) and RT size ratio in terms of memory-bit space (bottom).

Communication cost As depicted in Fig.9.2 (bottom-left), the communication cost ratio for PPC is relatively high compared to the SPT even if twice lower than the communication cost implied by the ST, when the size of the multicast group reaches 2000. This observation can be explained by the presence of high degree nodes (nodes that have a degree of the order to 100 or even higher) in the power law graph, as mentioned before. This results show (implicitly) that PPC are best suited for setting up shared MDT (where a single MDT can accommodate multiple $< s, g >$ pairs) more than selective MDT (one MDT per $< s, g >$ pair).

Nevertheless, these results suggest that further improvements on the communication process are desirable to make such scheme applicable for power law graphs comprising of the order of 10k nodes. Enforcing type-R message propagation by parts (after a certain fraction of the diameter the type-A message is redirected to the leaf node which selects least-cost candidate among the available sub-paths - search space segmentation) will show a significant improvement of at least one order of magnitude on the communication cost at the expense of slightly deteriorating stretch (as shown next (in B)).

2) 16k-nodes GLP topology and CAIDA Internet map WITH (row) Search Segmentation

Here, the PPC algorithm is simulated on i) synthetic power law graphs (16k nodes and 36k links) and ii) the CAIDA Internet AS-topology map (16k nodes and 48k links). In this case, the simulation scenario builds p2mp routing paths for multicast groups of increasing size from 500 to 4000 nodes (selected randomly) with increment of 500 nodes.

For the GLP topology (Fig.9.3a), the stretch of the PPC is slightly higher than 1 (1.03). The same trend is observed for the CAIDA map (moving from 1.05 to 1.02). In both cases, it remains almost constant when the multicast group size increases. The relative gain (max: 0.1-min: 0.01) compared to the SPT decreases with increasing group size, this trend is deeper for the CAIDA topology (see Fig.9.4a).

For both topologies, the number of RT entries and memory space required to store them, show substantial but decreasing gain for the PPC compared to the ST as the multicast group size increases. For a multicast group size of 500|4000 nodes, the number 1453|9271 of RT entries produced by the PPC is 12.0|2.7 times smaller than the number 17407|24997 of RT entries produced by the ST (see Fig.9.3b and 9.4b).

Similar results are observed for the memory space consumption. Compared to the SPT, the increase in communication cost for the PPC ranges from 14 to 18 times (for the CAIDA map) and from 17 to 50 (for the GLP topology) and even if both ranges are much smaller compared to the ST (see Fig.9.3c and 9.4c). The difference between topologies can be explained

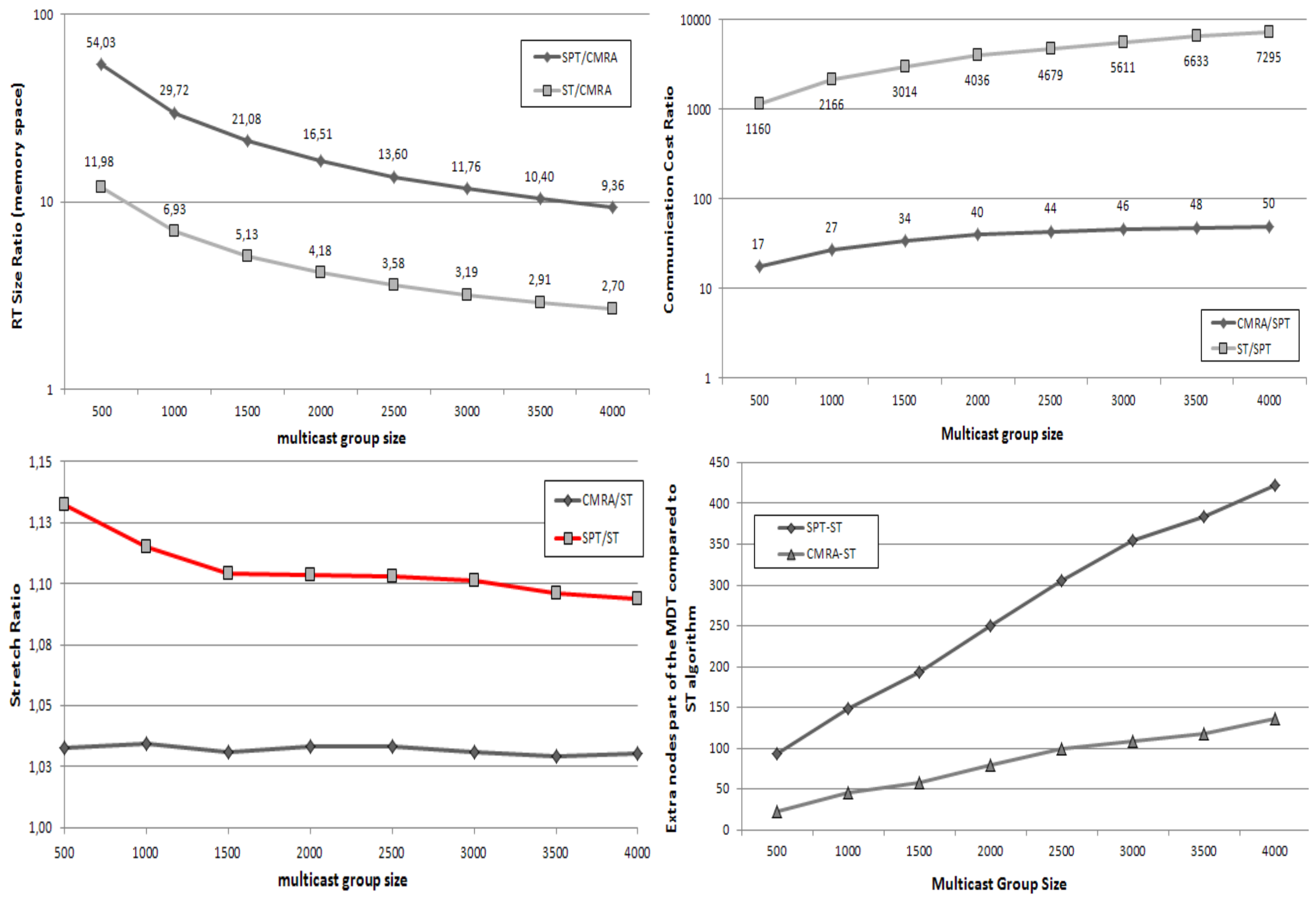


Figure 9.3: (GLP 16k-nodes Topology): (left-bottom) Stretch - (left-top) RT size ratio (memory-bit space) - (right-top) Communication cost ratio - (right-bottom) extra nodes part of the MDT compared to ST MDT.

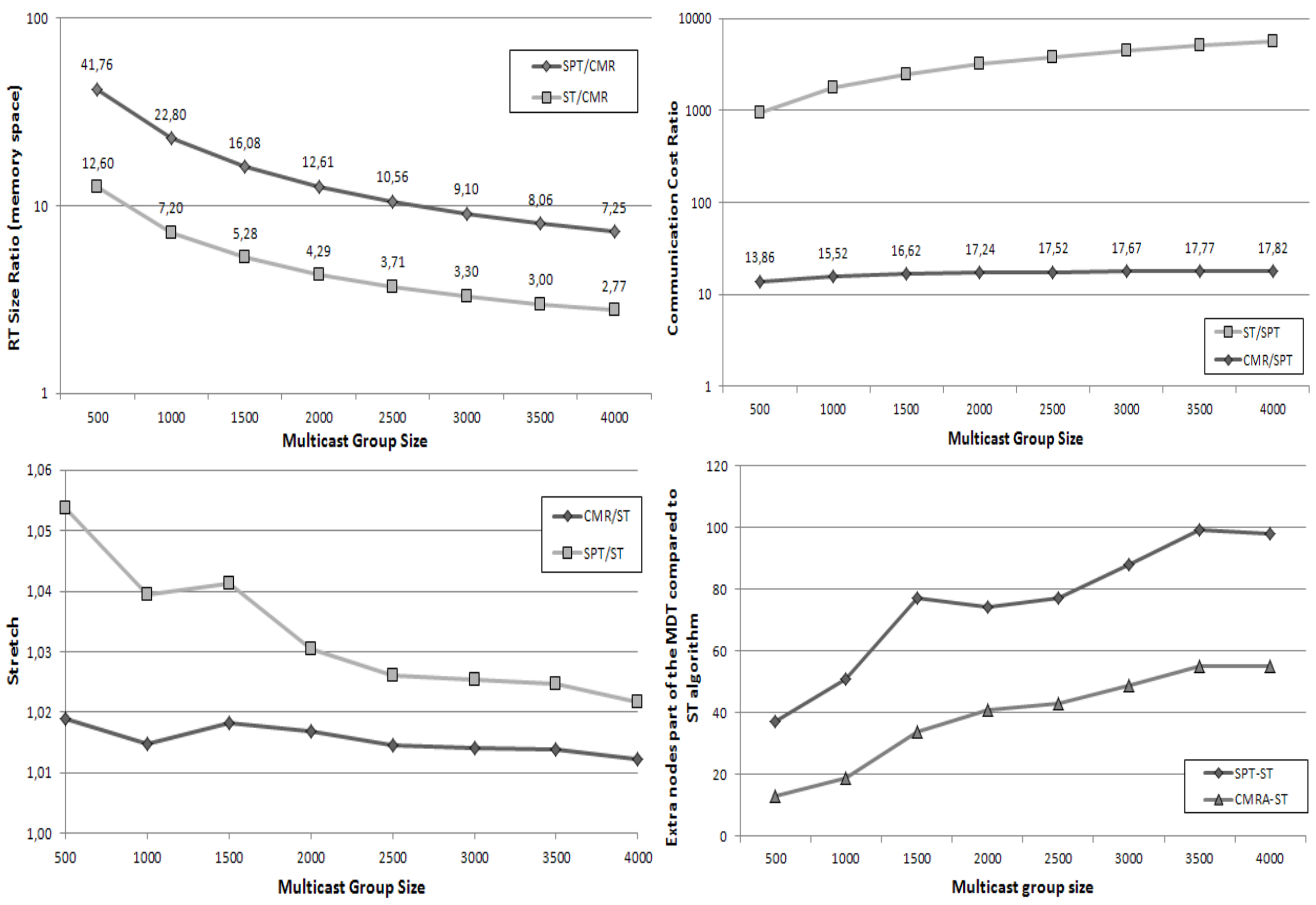


Figure 9.4: (CAIDA 16k-nodes map): (left-bottom) Stretch - (left-top) RT size ratio (memory-bit space) - (right-top) Communication cost ratio - (right-bottom) extra nodes part of the MDT compared to ST MDT.

by the higher number of edges in the GLP topology. Despite this noticeable difference, the SPT communication cost grows linearly with the multicast group size whereas the PPC curve is concave implying sub-linear dependence. Moreover, the PPC curve decelerates as the multicast group size increases leading to saturation ratio around 18 (for the CAIDA map) and 50 (for the GLP topology).

A.4 PPC Scheme Pseudo-Code

Algorithm 6 PPC Algorithm

```
Require: $G, T_{s,M}, \tau(u) = \tau_{max}, \pi = \pi_{max}, u \notin T_{s,M}$
Ensure: $p(u,v)* \neq \emptyset, c(u,v)* > 0$
 1: set $\mathcal{E} = \emptyset$
 2: W = neighbors(u), $w \in W$
 3: Send type-R message to each w node
 4: for $w \in W$ do
 5:     sender = u; receiver = w;
 6:     message m = {Type-R,sender,receiver,$\pi$,$\tau$}
 7:     $\mathcal{E} \leftarrow \mathcal{E} \cup \{m\}$
 8: end for
 9: while $\mathcal{E} \neq \emptyset$ do
10:     $m \leftarrow \mathcal{E}$
11:     if m.type equals to Type-R then
12:         ProcessingType-R(m)
13:     else
14:         ProcessingType-A(m)
15:     end if
16:     Update nodes' timer values
17:     for $n \in N$ do
18:         if $\tau(n)$.enable then
19:             if $\tau(n)$.expired then
20:                 $\Omega\{p(w,v), c(w,v)\}$ = computeSubBranching-
                    Path(n)
21:                 if $n \neq u$ then
22:                     sender = n; receiver = downstream(n)
23:                     m = {Type-A,sender,receiver,$\Omega$}
24:                     $\mathcal{E} \leftarrow \mathcal{E} \cup \{m\}$
25:                 end if
26:             else
27:                 Update $\tau(n)$
28:             end if
29:         end if
30:     end for
31: end while
32: Return $p(u,v)*$ and $c(u,v)*$
```

Algorithm 7 Processing Type-R Message

```
Require: $m$, m.type == Type-R
Ensure: new $m.sender = m.receiver$, $m.receiver \neq m.sender$
 1: i = m.sender and j = m.receiver
 2: if $j \in T_{s,M}$ then
 3:     $path = \emptyset$
 4:     $path \leftarrow path \cup \{j\}$
 5:     $\Omega = \{path, radial = 0\}$
 6:     m = {Type-A,sender=j,receiver=i,$\Omega$}
 7:     $\mathcal{E} \leftarrow \mathcal{E} \cup \{m\}$
 8: else
 9:     if $m.\pi > 0$ then
10:         W = neighbors(j)$\setminus\{i\}$
11:         if W = $\emptyset$ then
12:             m = {Type-A,j,i,$\emptyset$}
13:             $\mathcal{E} \leftarrow \mathcal{E} \cup \{m\}$
14:         else
15:             if j.firstime then
16:                 save the downstream node i at node j
17:                 $\pi^{'}$ = updated $m.\pi$
18:                 $\tau^{'}$ = updated $\tau$
19:                 enable $\tau = \tau^{'}$
20:                 for $w \in W$ do
21:                     m = {Type-R,sender=j,receiver=k,$\pi^{'}$,$\tau^{'}$}
22:                     $\mathcal{E} \leftarrow \mathcal{E} \cup \{m\}$
23:                 end for
24:             else
25:                 enable secondary timer
26:                 store the type-A message to this holding Type-
                    R message
27:             end if
28:         end if
29:     else
30:         m = {Type-A,j,i,$\emptyset$}
31:         $\mathcal{E} \leftarrow \mathcal{E} \cup \{m\}$
32:     end if
33: end if
```

Algorithm 8 Processing Type-A Message

```
Require: Message $m$, m.type == "answer"
Ensure: new message $m$, m.type == "answer"
1: i = m.receiver
2: k = m.sender
3: j = node of $T_{s,M}$
4: if Timer $\tau$ has not expired then
5:     A ← $A \cup \{m\}$ //save Type-A message at node i, set A
6:     W = neighbors(i)
7:     if $|A| = |W - 1|$ then
8:         $\Omega\{p_{i,j}, x_{i,j}\}$ = computeSubBranchingPath(i,A)
9:         sender = i; receiver = downstream(i)
10:        m = {Type-A,sender,receiver,$\Omega$}
11:        $\mathcal{E} \leftarrow \mathcal{E} \cup \{m\}$
12:    else
13:        Keep on holding stage
14:    end if
15: end if
```

Algorithm 9 Compute Sub Branching Path

```
Require: set A, node i
Ensure: $\Omega\{p(i,j)*, c(i,j)*\}$
1: $c(i,j)* = 10000$
2: for $m \in A$ do
3:     k = m.sender
4:     $radial_{k,j} = m.\Omega\{radial\}$ where $j \in T_{s,M}$
5:     $p_{k,j} = m.\Omega\{path\}$
6:     $x_{i,j} = tangent_{i,k} + radial_{k,j}$
7:     if $x_{i,j} < x^*_{i,j}$ then
8:         $x^*_{i,j} = x_{i,j}$
9:         $p^*_{i,j} \leftarrow p_{k,j} \cup \{i\}$
10:    end if
11: end for
12: Return $\Omega\{p(i,j)*, c(i,j)*\}$
```

Algorithm 10 Two-Stage Search Procedure

```
Require: G, N, $T = T_{s,M}$, diameter, u
Ensure: $\Omega_{global} \neq \emptyset$ if $\Omega_{local} = \emptyset$, or $\Omega_{local} \neq \emptyset$
1: $\pi = \pi_{max}$
2: set flag_e = 0
3: $\Omega_{local} \leftarrow$ PPC($T$,$\pi$,u,flag_e)
4: if $\Omega_{local}$ = $\emptyset$ then
5:     set $\pi = diameter$
6:     set flag_e = 1
7:     set $\mathcal{E} = \emptyset$
8:     L = active upstream interfaces to forward type-R
       flag_e=1 messages
9:     for l $\in L(u)$ do
10:        Send request m
11:        m = {type-R,u,l,$\pi^{'}$}
12:        $\mathcal{E} \leftarrow \mathcal{E} \cup \{m\}$
13:    end for
14:    while $\mathcal{E} = \emptyset$ do
15:        if $\pi > 0$ then
16:            if L(m.receiver) $\neq \emptyset$ then
17:                $b_v$ is "found"
18:                $\Omega_{global} \leftarrow$ PPC($T$,$p_{budget}$,u,flag e)
19:            else
20:                Forward type-R message, flag_e=1
21:                $\mathcal{E} \leftarrow \mathcal{E} \cup \{m\}$
22:            end if
23:        end if
24:    end while
25: end if
26: Returns $p(u,v)*$
```

A.5 Abraham Scheme Break-down

These figures are held as an auxiliary to the interpretation of the Abraham scheme proposed in [Abraham09] and break-down in Section 8.1 of Chapter 8. For instance, Figure 9.5 shows the initial stage of the process, where a leaf node a_1 wants to joint node $u = s$ (top scheme). It starts by (1) sending the SPLabel(a_1) to the center node $C(T_{i^*+1}(u = s))$. The route $a_1 - C(T_{i^*+1}(u = s))$ will be a shortest path route. Destination label = $\lambda(T_{i^*+1}, C(T_{i^*+1}(u = s)))$. Then, (2) that center node $C(T_{i^*+1}(u = s))$ does the mapping of $a_1 \rightarrow TZlabel(T_{i^*+1}(u = s), a_1)$, which is then passed to node u towards the multicast source - in this case $u = s$ (bottom scheme). Finally, (3) node $u = s$ knows how to forward to a_1 due to the received $TZlabel(T_{i^*+1}(u = s), a_1)$. Node $u = s$ keeps the tuple: $(u = s, C(T_{i^*+1}(u = s)), TZlabel(T_{i^*+1}(u = s), a_1))$ to perform the forwarding of data packets.

Figure 9.6 shows the arrival of a new leaf a_2 and the U_i set update procedure. Assume the case where a_1 is already part of the tree and the $T_{i*+1}(u)$ selected is from node $u = a_1$. The former process is then repeated. Note that the orange dots represent the neighboring nodes of a_2 within a radius of 2^{i*} to be updated, i.e., all U_i sets from $i = 0$ to $i <= i^*$.

Figure 9.7 and 9.8 show the two cases where a new leaf node b could be inside or outside the ball of a_2, $B(a_2, 2^{i<=i*})$, respectively. In the case leaf node b is inside the ball, some of its U_i sets were previously updated: $U_i = \{s, a_2\}$ from $i = 0$ to $i \in i^*$ and $U_i = \{s\}$ from $i = i^{*+1}$ to $i \in i_{max}$. As $d(b, u = a_2) < 2^{(i*+1)'}$, no (U_i) updates are propagated and no

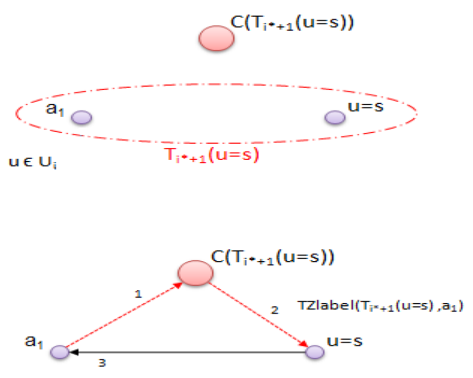


Figure 9.5: Initial stage: (top) a leaf node a_1 wants to joint node $u = s$. (bottom) The center node does the mapping and passes the TZLabel towards source node s.

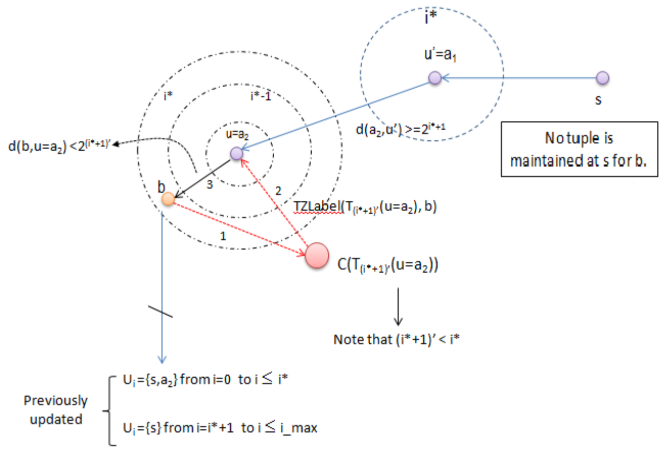
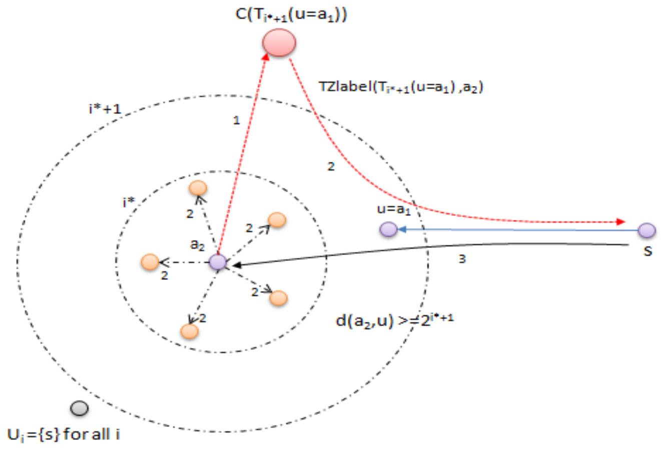

i*
u'=a_1
s
i*
i*-1
u=a_2
d(a_2,u')>=2^{i*+1}
No tuple is maintained at s for b.
d(b,u=a_2)<2^{(i*+1)'}
b
3
2
TZLabel(T_{(i*+1)'}(u=a_2), b)
1
C(T_{(i*+1)'}(u=a_2))
Note that (i*+1)' < i*
Previously updated
U_i={s,a_2} from i=0 to i ≤ i*
U_i={s} from i=i*+1 to i ≤ i_max

Figure 9.7: A new leaf node b is inside the ball of a_2, $B(a_2, 2^{i<=i*})$.



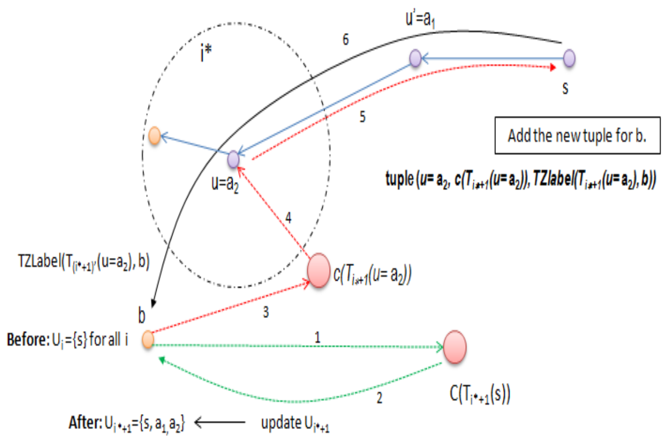

u'=a_1
6
i*
s
5
Add the new tuple for b.
u=a_2
tuple(u=a_2, c(T_{i*+1}(u=a_2)), TZlabel(T_{i*+1}(u=a_2), b))
4
TZLabel(T_{(i*+1)'}(u=a_2), b)
c(T_{i*+1}(u=a_2))
b
3
Before: U_i={s} for all i
1
2
C(T_{i*+1}(s))
After: U_{i*+1}={s, a_1, a_2}
update U_{i*+1}

Figure 9.6: The arrival of a new leaf a_2 and the U_i set update procedure.

Figure 9.8: A new leaf node b is outside the ball of a_2, $B(a_2, 2^{i<=i*})$.

information is passed to the source node s (it will not keep a tuple for leaf node b). On the contrary, if leaf node b is outside the ball of a_2, b will trigger the normal process. As $U_i = \{s\}$, it queries its $C(T_{i*+1}(s))$, which will update leaf node b's U_i sets with the new nodes u that have joined before the tree (i.e., other leaf nodes plus the source), giving it the chance to go for a closer (i.e. lower index i) center node $c(T_{i*+1}(u))$, e.g. $u \neq s = a_2$.

Acronyms

A-PPC	AnyTraffic PPC
ACK	Acknowledge
AS	Autonomous System
ASON	Automatically Switched Optical Network
BCP	Burst control packet
BCT	Background Control Task
BE	Best Effort
BGP	Border Gateway Protocol
BLP	Burst Loss Probability
BSC	Burst Switching Capable Interface
CP	Control Plane
DP	Data Plane
DWDM	Dense Wavelength Division Multiplexing
E2E	end-to-end
ERO	Explicit Route Object
FIB	Forwarding Information Base
FT	Forwarding Table
FTTx	Fiber to the x
GELS	GMPLS Ethernet Label Switching
GLP	Generalized Linear Preference
GMPLS	Generalized Multi-Protocol Label Switching
GOBS	GMPLS-OBS
GPON	Gigabit Passive Optical Network
HP	High Priority
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISC	Interface Switching Capable
ITU-t	International Telecommunication Union - Telecommunication Standardization Sector
LMP	Link Management Protocol
LOBS	Labeled OBS
LSP	Label Switched Path
LSU	Link-State Update
LTE	Long Term Evolution
MAN	Metro-Area Network
MDT	Multicast Distribution Tree
MIB	Management Information Base
MILP	Mixed Integer Linear Programming
MLN	Multi-Layer Network
MP2MP	multipoint-to-multipoint
MPLS	Multi-Protocol Label Switching
MRIB	Multicast Routing Information Base
MRN	Multi-Region Network
NACK	Negative acknowledge
NGN	Next Generation Network
NNI	Network-to-Network Interface
OAM	Operation, Administration and Maintenance
OBS	Optical Burst Switching
OCS	Optical Circuit Switching

OPEX	Operational Expenses
OPS	Optical Packet Switching
OSPF-TE	Open Shortest Path First
OTN	Optical Transport Network
P2P	point-to-point
P2MP	point-to-multipoint
PBB	Provider Backbone Bridge
PBT	Provider Backbone Transport
PCE	Path Computation Element
PIM	Protocol Independent Multicast
PPC	Pedroso, Papadimitriou, Careglio
QoE	Quality of Experience
QoS	Quality of Service
RSVP-TE	Resource Reservation Protocol
RT	Routing table
RWA	Routing and Wavelength Assignment/Allocation
SCT	Specific Control Task
SDH	Synchronous Digital Hierarchy
SNMP	Simple Network Management Protocol
SONET	Synchronous optical networking
SP	Shortest Path
SPT	SP Tree
ST	Steiner Tree
TE	Traffic Engineering
TED	TE-database
TCP	Transmission Control Protocol
TDM	Time Division Multiplexing
TIB	Tree Information Base
TSPEC	Traffic Specifications
UNI	User Network interface
URIB	Unicast Routing Information Basis
VT	Virtual Topology
WDM	Wavelength Division Multiplexing
WAN	Wide Area Network
WiMax	Worldwide Interoperability for Microwave Access

List of Figures

List of Tables

Technical Acknowledgements

I would like to thank to several entities and people that somehow have contributed to this work:

· to the Portuguese Government Entity, Fundação para a Ciência e a Tecnologia (FCT), for the granted PhD Fellowship SFRH / BD / 36950 / 2007.

· to Jordi Perello and Salvatore Spadaro for there valuable contribution and collaboration in the preparation of several scientific articles. Also, many thanks to Fernando Agraz and Ramon Casellas for their valuable comments.

· to Anna Manolova, who received me in Copenhagen for an amazing month. I really enjoyed the intensity of our discussions and the flow of ideas.

· to the people from ISCTE-IUL in Lisbon where I graduated as a Telecom Engineer before joining UPC. In particular, to Rui Lopes who has been there from the beginning of this journey; to Luís Cancela for his availability to co-coordinate two Master Thesis from ISCTE-IUL students at UPC, and finally to Professor Francisco Cercas for his precious help in order to solve the problem with FCT and the PhD fellowship at the early beginning of this journey.

· to the Master students João Baião and Rui Batalha from ISCTE-IUL in Lisbon, Portugal. Their works were important contributions to the deployment of the complex event-driven Java simulator of a GMPLS-OBS controlled network (so-called JA(G)OBS). I would like also to thank to the PhD. candidate and my friend Oscar Pedrola from UPC, Barcelona, Spain for his JAVOBS simulation tool and to his unmeasured technical support during my adaptation phase to such tool. Last but not least, I would like to thank to Dr. Sebastien Rumley from EPFL, Lausanne, Switzerland to his outstanding work in the creation of such flexible framework JAVANCO, the basis of both JAVOBS and JA(G)OBS. Without all these people, this project would not have been possible.

· to FP7 ICT EULER (Experimental UpdateLess Evolutive Routing), EU 7th Framework Program (FP7) for Research and Technological Development - Information and Communication Technologies (ICT) work Programme. Project Duration: 01/10/2010 - 30/09/2013.

· to FP6 IST IP NOBEL2 (Next Generation Optical Networks for Broadband European Leadership Phase 2), EU 6th Framework Program (FP6) for Research and Technological Development - Information Society Technologies (IST) Work Programme. Integrated Project (IP). Project Code: IST-FP6-027305. Project Duration: 01/03/2006 - 31/3/2008.

· to FP7 ICT NoE BONE (Building the Future Optical Network in Europe: The e-Photon/ONe Network), EU 7th Framework Program (FP7) for Research and Technological Development - Information and Communication Technologies (ICT) Work Programme. Network of Excellence (NoE). Project Code: ICT-FP7-216863. Project Duration: 01/01/2008 - 28/2/2011.

· to the Spanish Ministry of Science and Innovation through the CATARO (TEC2005-08051-C03-01) and DOMINO (TEC2010-18522) projects.

Bibliography

[Abraham06] I. Abraham, C. Gavoille, and D. Malkhi. "On space-stretch trade-offs: Lower bounds.", In SPAA, 2006

[Abraham08] I. Abraham et al., "Compact name-independent routing with minimum stretch", ACM Transactions on Algorithms, vol.4(3), Jun. 2008

[Abraham09] I. Abraham, D. Malkhi, D. Ratajczak, “Compact multicast routing,” Proc. 23rd Int. Symp. DISC’09, Elche, Spain, pp.364–378, Sep. 2009

[Akar10] N. Akar, E. Karasan, K. G. Vlachos, E. A. Varvarigos, D. Careglio, M, Klinkowski, and J. Solé-Pareta, “A survey of quality of service differentiation mechanisms for optical burst switching networks,” *Optical Switching and Networking*, vol. 7, no. 1, pp. 1-11, Jan. 2010.

[Awerbuch89] B. Awerbuch, A. Bar-Noy, N. Linial, D. Peleg, “Compact distributed data structures for adaptive routing,” Proc. 21st annual ACM STOC’89, Seattle, WA, United States, pp. 479–489, May 1989

[Barakat06] N. Barakat, E. H. Sargent, “Separating resource reservations from service requests to improve the performance of optical burst-switching networks,” *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 4, pp. 95-107, Apr. 2006.

[BGPReports] BGP Routing Table Analysis Reports, http://bgp.potaroo.net/

[Bondi00] A. B. Bondi, ’Characteristics of scalability and their impact on performance’, Proceedings of the 2nd international workshop on Software and performance, Ottawa, Ontario, Canada, 2000, ISBN 1-58113-195-X, pages 195 - 203

[Blazevic99] L. Blazevic, Y.-L- Boudec, "Distributed core multicast (DCM): a multicast routing protocol for many groups with few receivers", Proc. Networked Group Communication Workshop, Pisa, Italy, November 1999.

[Bu02] T. Bu, D. Towsley, “On distinguishing between Internet power law topology generators,” Proc. IEEE Infocom’02, pp. 638–647, New York, NY, USA, Jun. 2002

[CAIDA] CAIDA, http://www.caida.org

[CISCO10] Cisco, "Cisco Visual Networking Index: Forecast and Methodology, 2009Ű2014", white paper, July 2010

[Claffy93] K.C. Claffy, G.C. Polyzos, H.W. Braun, “Traffic characteristics of the T1 NSFNET backbone,” in *Proc. IEEE INFOCOM93*, San Francisco, USA, Apr. 1993.

[Chen04] Y.Chen, C. Qiao, Y. Yu, “Optical burst switching: a new area in optical networking research,” *IEEE Network*, vol. 18, no. 3, pp. 16-23, May-Jun. 2004.

[Croxton02] K. L. Croxton, B. Gendron, and T. L. Magnanti, "A comparison of mixed-integer programming models for non-convex piecewise linear cost minimization problems", Massachusetts Institute of Technology, Tech. Rep. OR 363-02, July 2002. [Online]. Available: http://dspace.mit.edu/handle/1721.1/5233

[Deering90] S. Deering, D. Cheriton, "Multicast routing in datagram internet-works and extended LANs", ACM Trans. Compt. Syst., vol. 8, May 1990.

[Dolzer04] K. Dolzer, "Mechanisms for quality of service differentiation in optical burst switched networks", Ph.D. dissertation, Institut fr Kommunikation- snetze und Rechnersysteme der Universitat Stuttgart, 2004

[Düser02] M. Düser and P. Bayvel, "Analysis of a Dynamically Wavelength-Routed Optical Burst Switched Network Achitecture", IEEE J. of Lightwave Technology, (2002).

[Farahmand07] F. Farahmand, V. M. Vokkarane, J. P. Jue, J. Rodrigues, M. Freire, "Optical burst switching network: A multi-layered approach", Journal of High Speed Networks, vol. 16 pp. 105Ű122, 2007.

[Farinacci11] D. Farinacci et. al., "Locator/ID Separation Protocol (LISP)", work in progress, draft-ietf-lisp-12, April 26, 2011

[Farrel06] A. Farrel, I. Bryskin, *GMPLS: architecture and applications*, Morgan Kaufmann Series in Networking, 2006.

[Fei02] Z. Fei, M. Ammar, E. Zegura, "Multicast server selection: problems, complexity, and solutions", IEEE J. Select. Areas Commun., vol. 20, no. 7, Sep. 2002.

[Garey77] M.R. Garey, R.L. Graham, D.S. Johnson, "The complexity of computing Steiner minimal trees", SIAM J. Appl. Math., vol. 32, no. 4, pp. 835–859, 1977

[Gavoille96] C. Gavoille and S. Perenn'es. Memory requirement for routing in distributed networks. In PODC, 1996

[Gauger02] C.M. Gauger, "Performance of converter pools for contention resolution in optical burst switching", in: Proc. OptiComm 2002, July 2002.

[Ghani00] N. Ghani, S. Dixit, and T. S. Wang, "On Ip-over-WDM Integration", IEEE Commun. Mag., vol.38, no. 3, pp. 72-84, March 2000.

[Gopalan03] K. Gopalan, ŞEfficient provisioning algorithms for network resource virtualization with QoS guarantees,Ť Ph.D. dissertation, Stony Brook University, 2003.

[Gunreben07] S. Gunreben and G. Hu, "A multi-layer analysis on reordering in optical burst switched networks", IEEE Communications Letters, vol. 11, no. 12, pp. 1013Ű1015, December 2007

[Guo07] H. Guo, et.al., "Proposal of a Multi-layer Network Architecture for OBS/GMPLS Network Interworking", Proc. Network Architectures, Management, and Applicstions V, SPIE, November 2007.

[Huston03] Huston, G., "Analyzing the Internet's BGP Routing Table", http://www.potaroo.net/papers/ipj/ 2001-v4-n1-bgp/bgp.pdf, 2003.

[Hwang92] F.K. Hwang, (Ed.), "The Steiner Tree Problem", Annals of Discrete Mathematics, vol. 53, Amsterdam, North-Holland Editions, 1992.

[IEEE02] IEEE Communication Magazine, Special Issue on Packet-Oriented Photonic Networks, September 2002

[ITU04] International Telecommunications Union (ITU-t), "Global information infrastructure, Internet Protocol aspects and next-Generation networks", Rec. Y.2001: Series Y, December 2004.

[IEEE09] IEEE 802.1Qay-2009 - "IEEE Standard for Local and metropolitan area networks– Virtual Bridged Local Area Networks Amendment 10: Provider Backbone Bridge Traffic Engineering(PBB-TE)", August 2009

[Inkret03] R. Inkret, A. Kuchar, B. Mikac, "Advanced infrastructure for photonic networks European research project", Extended Final Report of COST 266 Action, ISBN 953-184-064-4, 2003.

[ITU04] International Telecommunications Union, "Terms and definitions for Automatically Switched Optical Networks (ASON)", Recommendation G.8081/Y.1353, June 2004.

[ITU06] International Telecommunications Union (ITU-t), "Architecture for the automatically switched optical network (ASON)", Recommendation G.8080/Y.1304, June 2006

[Izal02] M. Izal and J. Aracil, "On the influuence of self-similarity on optical burst switching traffic", in Proceedings of IEEE Global Communications Conference (GLOBECOM 2001), San Antonio, TX (USA), November 2001.

[Jajszczyk05] A. Jajszczyk, "Automatically Switched Optical Networks: Benefits And Requirements," *IEEE Commun. Mag.*, Feb. 2005.

[Jinno07] M. Jinno, Y. Miyamoto and Y. Hibino, "Optical-Transport network in 2015", Nature Photonics, vol.1, March 2007.

[Kaheel03] A. Kaheel, H. Alnuweiri, "A strict priority scheme for quality-of service provisioning in optical burst switching networks," *Proc. of ISCC 2003*, 2003.

[Kim08] H. Kim, (Ed.), "On realizing all simple graphs with a given degree sequence", Discrete Mathematics, Apr. 2008.

[Kleinberg05] R. Kleinberg, J. Kleinberg, "Isomorphism and embedding problems for infinite limits of scale-free graphs", 15th ACM-SIAM SODA, Jan. 2005.

[Kleinroch77] L. Kleinrock and F. Kamoun, "Hierarchical routing for large networks: Performance evaluation and optimization", Computer Networks, 1:155–174, 1977.

[Klinkowski09a] M. Klinkowski, D. Careglio, J. Sole-Pareta, and M. Marciniak, "A Performance Overview of Quality of Service Mechanisms in Optical Burst Switching Networks (Chapter 1) in Current Research Progress of Optical Networks", M. Ma, Ed. Springer-Verlag, 2009.

[Klinkowski09b] M. Klinkowski, M. Pióro, and M. Marciniak, "Optimization of Routing in Optical Burst Switching Networks: a Multi-Path Routing Approach", (Chapter 5) in Graphs and Algorithms in Communication Networks - Studies in Broadband, Optical, Wireless, and Ad Hoc Networks, A. Koster and X. MuŸ noz, Eds. Springer, 2009.

[Klinkowski09c] M. Klinkowski, D. Careglio, J. Solé-Pareta and M. Marciniak, "Performance Overview of the Offset Time Emulated OBS Network Architecture", JOURNAL OF LIGHTWAVE TECHNOLOGY, VOL. 27, NO. 14, JULY 15, 2009.

[Klinkowski10] M. Klinkowski," An Overview of Routing Methods in Optical Burst Switching Networks", Optical Switching and Networking, vo.7, number 2, April 2010, pp.41-53.

[Krioukov04] D. Krioukov, K. Fall, and X. Yang, "Compact routing on Internet-like graphs", in IEEE INFOCOM, 2004

[Krioukov07] D. Krioukov, "On Compact Routing for the Internet", ACM SIGCOMM Computer Communication Review, July 2007.

[Krioukov08] D. Krioukov et al., "Efficient navigation in scale-free networks embedded in hyperbolic metric spaces", CoRR, May 2008.

[Lim06] H. Lim, S. Ahn, E. Kim, H. Park, "A QoS-Based Adaptive Resource Sharing Protection for Optical Burst Switching Networks," *Proc. ICOIN 2006*, 2006.

[Liu03] J. Liu, N. Ansari, T. J. Ott, "FRR for latency reduction and QoS provisioning in OBS networks," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 7, pp. 1210-1210, Sep. 2003.

[Long03] Keping Long, Rodney S. Tucker, Chonggang Wang, "A New Framework and Burst Assembly for IP DiffServ over Optical Burst Switching Networks", proceeding of IEEE Globecom 2003,vol6. pp3159-3164, 1-5 Dec. 2003

[Long05a] X. Yang, K. Long, X. Yang, Y. Zhang, and G. Liu: A General Framework for GMPLS-Based OBS Networks. Proceedings of SPIE, Vol. 6022, December 2005.

[Long05b] Long et. al., "Generalized MPLS (GMPLS) architecture's extensions for Optical Burst Switch network", draft-long-gmpls-obs-00.txt (expired), November 2005.

[Long06] K. Long, X. Yang, S. Huang, and Y. Kuang: A GMPLS-based OBS Architecture for IP- over-WDM Networks. In Proc. of SPIE Network Architectures, Management and Applications IV, Vol. 6354, 2006.

[MCE10] "Telecom Services Industry", Management Centre Europe (MCE), the executive issue, n36, Quarter1 2010

[Maesschalck03] S. De Maesschalck, D. Colle, I. Lievens, M. Pickavet, P. Demeester, C. Mauz, M. Jaeger, R. Inkret, B. Mikac and J. Derkacz, "Pan-European Optical Transport Networks: An Availability-based Comparison," *Phot. Netw. Commun.*, vol.5, no.3, May 2003.

[Mahony06] M.J. O'Mahony, C. Politi, D. Klonidis, R. Nejabati, D. Simeonidou, "Future Optical Networks," *IEEE/OSA J. Lightw. Technol.*, vol. 24, no. 12, pp. 4684-469, Dec. 2006.

[Manolova07] A. Manolova, S. Ruepp, J. Buron, L. Dittmann and L. Ellegaard.: Advantages and Challenges of the GMPLS/OBS Integration. In Proc. VI GMPLS Workshop, pp.133-144, Girona, Spain, April 2007.

[Manolova10] A. Manolova, S. Ruepp, "The OBS Control Plane: GMPLS Integration or Not?". In Proc. IX GMPLS Workshop, Girona, Spain, May 2010.

[Morrow05] Monique J. Morrow, Mallik Tatipamula, Adrian Farrel, "GMPLS: THE PROMISE OF THE NEXT-GENERATION OPTICAL CONTROL PLANE", Guest Editorial, IEEE Communication Magazine, July 2005

[Minoux86] M. Minoux, "Mathematical Programming: Theory and Algorithms", John Wiley and Sons, 1986.

[Minoux01] M. Minoux, "Discrete cost multicommodity network optimization problems and exact solution methods", Annals of Operations Research, vol. 106, no. 1, pp. 19-46, 2001

[Miguel04] I. De Miguel, "Polymorphic Architectures for Optical Networks and their Seamless Evolution towards Next Generation Networks",Photonic Network Communications

[Pasqualini05] Sandrine Pasqualini et.al., "Influence of GMPLS on Network Providers' Operational Expenditures: A Quantitative Study", IEEE Communications Magazine, July 2005

[Pedrola10] O. Pedrola, M. Klinkowski, D. Careglio, J. Solé-Pareta, S. Rumley, C. Gaumier, "JAVOBS: a flexible simulator for OBS network architectures," *J. Networks*, vol. 5, no. 2, pp. 256-264, Feb. 2010.

[Pedroso08] P. Pedroso, D. Careglio, R. Casellas, M. Klinkowski, and J. Solé-Pareta, "An integrated GMPLS/OBS Control Plane: RSVP and OSPF extensions proposal", IEEE CSNDSP, Graz, Austria, July2008

[Pedroso11a] P. Pedroso et. al., "JA(G)OBS: an event-driven Java simulator for a GMPLS-controlled OBS network", Technical Report, July 2011.

[Pedroso11b] P. Pedroso and D. Papadimitriou, "PPC: a Name-Independent Compacy Multicast Routing", available as Technical Report, UPC-DAC-RR-CBA-2011-15, March 2011

[Phuritatkul07] J. Phuritatkul, Y. Ji, and S. Yamada, "Proactive wavelength pre-emption for supporting absolute QoS in optical-burst-switched networks", IEEE Journal of Lightwave Technology, vol. 25, no. 5, pp. 1130Ű1137, May 2007.

[Pioneer02a] Pioneer Consulting, "An Economic Assessment of Gmpls ", July 2002.

[Pioneer02b] Pioneer Consulting, "Gmpls and the Optical Control Plane: An Analysis of Profitability and Performance in Optical Networks", July 2002.

[Pioro04] M. Pióro and D. Medhi, Routing, Flow, and Capacity Design in Communication and Computer Networks. Morgan Kaufmann, 2004

[Peleg89] D. Peleg and E. Upfall, A trade-off between space and efficiency for routing tables," J. ACM, vol. 36, no. 3, pp. 510–530, Jul. 1989

[Qiao99] 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, March 1999.

[Qiao00] C. Qiao: Labeled Optical Burst Switching for IP-over-WDM Integration. IEEE Communications magazine, Vol. 38, No. 9, pp. 104 - 114, September 2000.

[RWAinfo11] Y. Lee et.al., "Routing and Wavelength Assignment Information Model for Wavelength Switched Optical Networks", work in progess: draft-ietf-ccamp-rwa-info-11.txt, March 14, 2011

[Rosberg03] Z. Rosberg, Hai Le Vu, M. Zukerman, J. White, "Performance Analyses of Optical Burst Switching Networks," *IEEE J. Sel. Areas in Commun.*, vol. 21, no. 7, Sep. 2003, pp. 1187-1197.

[SAGE] Sage's Graph Library. Available at http://www.sagemath.org/.

[Sahara03] A. Sahara, K. Shimano, Y. Takigawa and M. Koga, "Optical burst data switching utilising GMPLS signalling", ELECTRONICS LETTERS 21st August 2003 Vol. 39, No. 17, pp. 1267-1269.

[Sriram03] K. Sriram, D. Griffith, et.al., "OBS: Benefits and Challenges", The First International Workshop on Optical Burst Switching (WOBS), Dallas, USA, October 2003.

[Syski60] R. Syski, "Introduction to Congestion Theory in Telephone Systems", North-Holland, 1960.

[Takacs08] A. Takács, H. Green, and B. Tremblay, "GMPLS Controlled Ethernet: An Emerging Packet-Oriented Transport Technology", IEEE Communications Magazine, September 2008

[Takahashi80] H. Takahashi, A. Matsuyama, “An approximate solution for the Steiner problem in graphs”,Math. Japonica, pp. 573–577, 1980

[Teng05] J. Teng and G. N. Rouskas, Traffic Engineering Approach to Path Selection in Optical Burst Switching Networks", Journal of Optical Networking", vol.4,n.11,pp.759-777, 2005

[Thaler00] D. Thaler, M. Handley, "On the aggregation of multicast forwarding state", Proc. Infocom 2000, Tel Aviv, Israel, March 2000.

[Thorup01] M. Thorup, and U. Zwick, “Compact routing schemes,” Proc. 13th Annual ACM SPAA’01, Heraklion, Crete, Greece, pp. 1–10, Jul. 2001.

[Tian98] J. Tian, G. Neufeld, "Forwarding state reduction for sparse mode multicast communications", Proc. Infocom 1998, San Francisco, CA, March 1998.

[Radoslavov99] P.I.Radoslavov, R. Govindan, D. Estrin, "Exploiting the bandwidth-memory tradeoff in multicast state aggregation", Technical Report 99-697, Computer Science Department, University of Southern California, July 1999.

[Yang06] L. Yang, G.N. Rouskas, “Adaptive path selection in OBS networks,” *IEEE/OSA J. Lightw. Technol.*, vol. 24, no. 8, pp. 3002-3011, Aug. 2006.

[Yang07a] L. Yang and G. N. Rouskas, "Generalized Wavelength Sharing Policies for Absolute QoS Guarantees in OBS Networks", IEEE Journal of Selected Areas in Communications, vol 25, number 4, pp93-104, April 2007

[Yang07b] L. Yang and G. N. Rouskas, "Optimal Wavelength Sharing Policies in OBS Networks Subject to QoS Constraints", IEEE Journal of Selected Areas in Communications, vol 25, number 9, pp. 40-49, December 2007

[Yang08] D.-N. Yang, W. Liao, "Optimal state allocation for multicast communications with explicit multicast forwarding", IEEE Trans. Parallel and Distr. Sys., vol. 19, no. 4, April 2008.

[Yao03] S. Yao, B. Mukherjee, S.J.B Yoo, “A Unified Study of Contention-Resolution Schemes in Optical Packet-Switched Networks,” *IEEE/OSA J. Lightw. Technol.*, vol. 21, no. 3, pp. 672-683, Mar. 2003.

[Yoo00] M. Yoo, C. Qiao, S. Dixit, “QoS performance of optical burst switching in IP-over-WDM networks,” *IEEE J. Sel. Areas Commun.* , vol. 18, no. 10, pp. 2062-2071, Oct. 2000.

[Wei00] J. Y. Wei and R. I. McFarland, "Just-in-Time signaling for WDM optical burst switching networks", Journal of Lightwave Technology, Vol. 18, no. 12, December 2000, pp. 2019-2037.

[Wong00] T. Wong and R. Katz, "An Analysis of Multicast Forwarding State Scalability," Proc. Eighth IEEE Int’l Conf. Network Protocols (ICNP ’00), pp. 105-115, 2000.

[Xin04] Y. Xin, J. Teng, G. Karmous-Edwards, G. N. Rouskas, D. Stevenson, “Fault management with fast restoration for optical burst switched networks,” *Proc. of BROADNETS 2004*, 2004.

[Xu03] J. Xu, C. Qiao, J. Li, G. Xu, “Efficient channel scheduling algorithms in optical burst switched networks,” *Proc. of IEEE INFOCOM 2003*, 2003.

[Zalesky09] Andrew Zalesky, "To Burst or Circuit Switch?", IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 17, NO. 1, pp.305-318, FEBRUARY 2009

[Zhang04] Q. Zhang, V. M. Vokkarane, J. P. Jue, and B. Chen, "Absolute QoS differentiation in optical burst-switched networks", IEEE Journal on Selected Areas in Communications, vol. 22, no. 9, pp. 1781-1795, November 2004.

[RFC2209] R. Braden, L. Zhang, “Resource ReSerVation Protocol (RSVP) – Version 1 Message Processing Rules,” *IETF RFC 2209*, Sept. 1997.

[RFC2328] J. Moy, "OSPF Version 2", IETF RFC 2328, April 1998

[RFC3209] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, “RSVP-TE: Extensions to RSVP for LSP Tunnels,” *IETF RFC 3209*, Dec. 2001.

[RFC3417] J. Case et.al., "A Simple Network Management Protocol (SNMP)", IETF RFC 1157, May 1990

[LongDraft05] "GMPLS extensions to OBS network", IETF draft v0, 2005, K. Long et. al.

[RFC3471] L. Berger, "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description", IETF RFC 3471, January 2003

[RFC3473] L. Berger, "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", *IETF RFC 3473*, Jan. 2003.

[RFC3477] K. Kompella, Y. Rekhter, "Signaling Unnumbered Links in RSVP-TE," *IETF RFC 3477*, Jan. 2003.

[RFC3945] E. Mannie et. al, "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", IETF RFC 3945, October 2004

[RFC4202] K. Kompella and Y. Rekhter, "Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", IETF RFC 4202, October 2005

[RFC4203] K. Kompella and Y. Rekhter, "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", IETF RFC 4203, October 2005

[RFC4206] K. Kompella, Y. Rekhter, "Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE)", October 2005

[RFC4601] B. Fenner, M. Handley, H. Holbrook, I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM)," Internet Engineering Task Force (IETF), RFC 4601, Aug. 2006

[RFC4632] V. Fuller, T. Li, "Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan", IETF 4632, August 2006

[RFC4657] J. Ash and J.L. Le Roux, "Path Computation Element (PCE) Communication Protocol Generic Requirements", September 2006

[RFC4875] R. Aggarwal et. al., "Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs), IETF RFC4875, May 2007

[RFC4974] D. Papadimitriou, A. Farrel, "Generalized MPLS (GMPLS) RSVP-TE Signaling Extensions in Support of Calls", RFC 4974, August 2007.

[RFC4984] D. Meyer et. al., "Report from the IAB Workshop on Routing and Addressing", IETF RFC 4984, September 2007

[RFC5212] K. Shiomoto, D. Papadimitriou, JL. Le Roux, M. Vigoureux, D. Brungard, Requirements for GMPLS-Based Multi-Region and Multi-Layer Networks (MRN/MLN), July 2008.

[RFC5307] K. Kompella, Y. Rekhter, "IS-IS Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", October 2008

[RFC5440] JP. Vasseur and JL. Le Roux, "Path Computation Element (PCE) Communication Protocol (PCEP)", March 2009

[RFC5828] D. Fedyk, L. Berger, L. Andersson, "Generalized Multiprotocol Label Switching (GMPLS) Ethernet Label Switching Architecture and Framework", *IETF RFC 5828*, Mar. 2010.

[RFC6001] D. Papadimitriou et al., "Generalized MPLS (GMPLS) Protocol Extensions for Multi-Layer and Multi-Region Networks (MLN/MRN)", *IETF RFC6001*, Oct. 2010.

[RFC6002] L. Berger, D. Fedyk, "Generalized MPLS (GMPLS) Data Channel Switching Capable (DCSC) and Channel Set Label Extensions", Internet Engineering Task Force (IETF), October 2010

Scientific Production

Journal Papers:

2011 - M. Klinkowski, P. Pedroso, D. Careglio, M. Pióro, and J. Solé-Pareta, "Joint Routing and Wavelength Allocation subject to Absolute QoS Constraints in OBS Networks", IEEE/OSA Journal of Lightwave Technology (JLT)

under submission:

2011 - P. Pedroso, J. Perello, D. Careglio, S. Spadaro and J. Solé-Pareta, "Optimized Burst LSP Design for Absolute QoS in GMPLS-controlled OBS network", Journal of Optical Communications and Networking (JOCN) - major revisions (waiting for final decision)

in progress:

2011 - P. Pedroso, A. Manolova, D. Careglio, J. Perello, S. Spadaro and J. Solé-Pareta, "Fully Dynamic GMPLS-OBS network operation", Computer Network (being prepared to be submitted before Nov. 2011)

2011 - P. Pedroso, A. Manolova, D.Careglio, J. Perello, S. Spadaro and J. Solé-Pareta, "GMPLS-OBS Interoperability: pushing it forward", IEEE Communication Magazine (being prepared to be submitted before Dec. 2011)

2011/2012 - P. Pedroso, D. Papadimitriou and D. Careglio, "Name-independent Compact AnyTraffic Routing", IEEE/ACM ToN (scheduled to be submitted by the end of the year 2011, beginning of 2012)

Conference Papers:

2011 - Pedro Pedroso, Dimitri Papadimitriou and Davide Careglio, "Dynamic Compact Multicast Routing on Power-Law Graphs ", IEEE Globecom, Houston 2011.

2011 - Pedro Pedroso et. al., "Functional Model of a Routing System Architecture", 1st Workshop Future Internet Efficiency in high-speed networks (W-FIERRO 2011) - joint work resulting from the EULER project

2011 - Dimitri Papadimitriou, Pedro Pedroso and Piet Demeester, "Performance Analysis of a Dynamic Compact Multicast Routing Scheme", 13es Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel), France, May 2011.

2011 - Pedro Pedroso, Jordi Perelló, Miroslaw Klinkowski, Davide Careglio, Salvatore Spadaro, Josep Solé-Pareta, "A GMPLS/OBS Network Architecture Enabling QoS-aware End-to-End Burst Transport", IEEE Conference on High Performance Switching and Routing (HPSR), Cartagena, Spain, July 2011

2010 - Miroslaw Klinkowski, Pedro Pedroso, M. Pióro, Davide Careglio, Josep Solé-Pareta,"Virtual Topology Design in OBS Networks," in *Proc. IEEE ICTON2010*, Munich, Germany, Jun. 2010.

2010 - Dimitri Papadimitriou, Pedro Pedroso, Davide Careglio, "AnyTraffic Labeled Routing", IEEE ICC, Cape Town, South Africa, May 2011

2009 - Pedro Pedroso, Oscar Pedrola, Dimitri Papadimitriou, Miroslaw Klinkowski and Davide Careglio, "Anytraffic routing algorithm for label-based forwarding", IEEE Globecom, Honolulu, Nov. 2009

2008 - Pedro Pedroso, Davide Careglio, Ramon Casellas, Miroslaw Klinkowski, and Josep Solé-Pareta, "An integrated GMPLS/OBS Control Plane: RSVP and OSPF extensions proposal", in Proc. IEEE CSNDSP, July 2008

2007 - Pedro Pedroso, Josep Solé-Pareta, Davide Careglio, Miroslaw Klinkowski, "Integrating GMPLS in the OBS Networks Control Plane", in Proc. IEEE ICTON2007, Rome, Italy 2007

under submission:

2011 - Dimitri Papadimitriou, Pedro Pedroso and Davide Careglio, "Design and Performance Analysis of Dynamic Compact Multicast Routing", IEEE Infocom, Orlando 2012

Technical Report:

2011 - P. Pedroso and D. Papadimitriou, "PPC: a Name-Independent Compact Multicast Routing", available as Technical Report, UPC-DAC-RR-CBA-2011-15, March 2011