UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Bringing cognition to multilayer transport networks

Alba Pérez Vela

Universitat Politècnica de Catalunya

Optical Communications Group

Bringing Cognition to Multilayer Transport Networks

Alba Pérez Vela

apvela@ac.upc.edu

A thesis presented in partial fulfillment of the requirements for the degree of

Philosophy Doctor

Advisor: Dr. Luis Velasco

Co-advisor: Dr. Marc Ruiz

September 2018

ISBN 978-84-09-02269-4

Registration date May 11, 2018

Optical Communications Group (GCO)

Universitat Politècnica de Catalunya (UPC)

C/ Jordi Girona, 1-3

Campus Nord, D4-213

08034 Barcelona, Spain

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola de Doctorat

Acta de calificación de tesis doctoral

Curso académico:

Nombre y apellidos

Programa de doctorado

Unidad estructural responsable del programa

Resolución del Tribunal

Reunido el Tribunal designado a tal efecto, el doctorando / la doctoranda expone el tema de la su tesis doctoral titulada _____

_____.

Acabada la lectura y después de dar respuesta a las cuestiones formuladas por los miembros titulares del tribunal, éste otorga la calificación:

☐ NO APTO ☐ APROBADO ☐ NOTABLE ☐ SOBRESALIENTE

(Nombre, apellidos y firma) Presidente/a	(Nombre, apellidos y firma) Secretario/a

(Nombre, apellidos y firma) Vocal	(Nombre, apellidos y firma) Vocal	(Nombre, apellidos y firma) Vocal

_____, _____ de _____ de _____

El resultado del escrutinio de los votos emitidos por los miembros titulares del tribunal, efectuado por la Escuela de Doctorado, a instancia de la Comisión de Doctorado de la UPC, otorga la MENCIÓN CUM LAUDE:

☐ SÍ ☐ NO

(Nombre, apellidos y firma) Presidente de la Comisión Permanente de la Escuela de Doctorado	(Nombre, apellidos y firma) Secretaria de la Comisión Permanente de la Escuela de Doctorado

Barcelona a _____ de _____ de _____

Agradecimientos

En primer lugar quiero agradecer mis directores de tesis Luis Velasco y Marc Ruiz; gracias por haberme abierto las puertas de la investigación y haberme brindado la oportunidad de desarrollar este trabajo a vuestro lado. Gracias por vuestra atención, paciencia, dedicación y esos largos ratos frente a la pizarra. Para mí, ha sido un privilegio poder contar con vuestra ayuda y compartir estos años en el grupo. En especial a Luis; gracias por creer desde el comienzo en mis capacidades y darme esa confianza que me faltaba en los momentos más críticos.

A todos los compañeros del Grupo de Comunicaciones Ópticas, en especial a Lluís; gracias por estar siempre dispuestos a ayudar y a compartir vuestro conocimiento. El camino ha sido más fácil con vosotros! A Alberto y Ángela; gracias por vuestra afectuosa acogida y apoyo durante mi enriquecedora estancia en la Universidad de California, Davis.

Gracias a todos los familiares y amigos que por extensión sería difícil citar. En particular quiero recordar a mis antiguos compañeros y compañeras de la Universidad de Zaragoza con los que empecé mi andadura por el mundo de la física. En especial a Laurita, gracias por enseñarme la necesidad de llegar siempre a la raíz del problema y disfrutar en el proceso.

Gracias a todas las personas que, de una forma u otra habéis sido claves en mi vida profesional y personal y que os he ido encontrando a lo largo de mi camino. A Fran y Silvia, gracias por vuestro sincero interés durante toda mi andadura. Fran, ánimo con la tesis!

Gracias a las que han llegado, Elina y Maia, que preguntáis por mi trabajo sin entender bien qué significa. Algún día leeréis este agradecimiento y sabréis que valioso ha sido vuestro estímulo.

Y gracias a los que ya no están a mi lado, pero permanecen en mi pensamiento y en mi corazón, mis abuelos, Ramona, Celestino...tan imprescindibles para mí, desde pequeña me habéis apoyado y seguido mi trayectoria y sé que estáis muy orgullosos. En especial a mi abuelo Luis, se la satisfacción tan grande que hubieras tenido al verme llegar hasta aquí.

Gracias a mi familia por estar siempre a mi lado y seguir de cerca mi camino. Por apoyarme incondicionalmente durante todos estos años y siempre. Gracias por celebrar conmigo mis éxitos y hacer menos amargos los fracasos. Mamá, Papá, Tito, Pablo y Lyolya gracias de corazón.

A Daniel, gracias por tu apoyo incondicional, por transmitirme siempre serenidad y tus palabras de ánimo cuando más las he necesitado. Y sobre todo, gracias por tu paciencia y tu cariño.

Esta tesis es un éxito compartido. Sin vosotras y vosotros no habría sido posible. Gracias.

We do not know a truth without knowing its cause.
Aristotle, Nicomacheian Ethics, I.1.

Abstract

Operator's transport networks are becoming increasingly more complex due to the large number of network layers needed to support the ever increasing number of new services (e.g., video-on-demand, social media, on-line gaming, or video and voice calls) with stringent requirements like very low latency and high throughput, as well as the number of users of those services. As a result, innovative approaches for operating the networks is mandatory in order to fulfill those tight requirements, while reducing costs.

The main objective of this PhD thesis is improving network operation by introducing *autonomic networking* capabilities. To this end, we study algorithms targeting network healthiness by monitoring both, the optical (L0) and the packet (L2) layers. An in depth study concerning centralized vs distributed architectures is carried out for anticipating anomaly or degradation detection before enough to give time to re-optimization algorithms. This will allow to plan the most adequate re-optimization that will end in, e.g., re-routing those affected demands according to their Service Level Agreement (SLAs) aiming at reducing the traffic affected by the detected degradation.

This main goal is achieved by the following five specific goals:

1. *Traffic Anomalies at the packet layer*. A *score-based* anomaly detection method is proposed for improving single Origin-Destination (OD) traffic anomalies detection. In addition, a method is devised to deal with the case of multiple related traffic anomalies triggered by an external event. By anticipating whether other ODs are anomalous after detecting one anomalous OD pair, the number of network reconfigurations, total reconfiguration time, as well as traffic losses are improved.
2. *Failure detection and localization/identification at the optical layer* based on bit error rate (BER) monitoring. BANDO and LUCIDA algorithms are proposed to, first, detect significant BER changes in optical connections, and then, to identify the most probable failure pattern. Devoted to soft failure localization, two techniques for active monitoring during commissioning testing and for passive in-operation monitoring are proposed.

3. *Network Reconfiguration*. Two reconfiguration algorithms were devised after anomalies at L2 and degradation at L0 are detected. The ODEON optimization problem is proposed to reconfigure the VNT, whereas the SCULPTOR algorithm is proposed to be triggered for demand re-routing after receiving certain BANDO notifications regarding significant BER change.

4. *Cognitive Architecture*. A monitoring and data analytics (MDA) architecture is devised aiming to reduce the amount of data to be conveyed and to minimize anomaly and degradation detection times. Representative use cases for autonomic networking in multilayer scenarios experimentally validate the distributed MDA architecture presented in this PhD thesis.

5. *Visualization Techniques*. Visualization techniques with specific task-oriented charts are proposed to help operators. For such visualization to be useful for human operators, an overwhelming amount of monitoring data needs to be pre-processed. A use case for failure localization is utilized as a guiding thread.

It shall be mentioned that part of the work reported in this thesis has been done within the framework of European and National projects. Specifically, the *METRO High bandwidth, 5G Application-aware optical network, with edge storage, compUte and low Latency* (METRO-HAUL), H2020-ICT-2016-2. (G.A. 761727) funded by the European Commission, and the *Service-oriented hYbrid optical NEtwork and cloud infrastructuRe featuring high throuGhput and ultra-low latency* (SYNERGY). Ref: TEC2014-59995-R, 2015-2017 and the *cogniTive 5G application-aware optical metro netWorks Integrating moNitoring, data analyticS and optimization* (TWINS) Ref: TEC2017-90097-R, 2018-2020, both funded by the Spanish Ministry of Economy, Industry and Competitiveness (MINECO).

Resumen

Las redes de transporte de los operadores son cada vez más complejas debido al gran número de capas que son necesarias en la red para poder dar soporte tanto al creciente número de servicios (por ejemplo, video bajo demanda, redes sociales, juegos on-line, o vídeo llamadas) que requieren estrictos valores de baja latencia o alta conectividad, así como la creciente cantidad de usuarios que reclaman estos nuevos servicios. Por lo tanto, para poder satisfacer estos rigurosos requisitos y reducir costes, es necesario desarrollar enfoques innovadores para la operación de estas redes.

El objetivo principal de esta tesis doctoral se centra en mejorar la operación de la red mediante la introducción de capacidades *autónomas*. Con este objetivo, se proponen algoritmos para la evaluación del estado de la red mediante la monitorización tanto de la capa óptica (L0), como de la capa de paquetes (L2). Asimismo, se compara el uso de una arquitectura centralizada frente a una distribuida para anticipar la detección de anomalías o degradaciones con suficiente antelación, de forma que se disponga de tiempo suficiente para poder aplicar algoritmos de re-optimización. Esto permitirá planificar la re-optimización en el mejor momento resultando en, por ejemplo, el re-enrutamiento de las demandas en función del Acuerdo de Nivel de Servicio (SLA), de forma que se reduzca significativamente el tráfico afectado por la degradación detectada.

Este objetivo principal se logra mediante los cinco siguientes objetivos específicos:

1. *Anomalías de tráfico en la capa de paquetes*. Se propone un método de detección de anomalías basado en *puntuaciones* para mejorar la detección de anomalías de tráfico Origen-Destino (OD). Además, se diseña un método para tratar el caso de múltiples anomalías de tráfico relacionadas y desencadenadas por un evento externo. Al ser capaces de anticipar otros OD también anómalos después de haber detectado un primer par OD anómalo, se puede rebajar el número de reconfiguraciones que se tienen que realizar en la red, con la consiguiente reducción del tiempo total de reconfiguración, minimizando finalmente las pérdidas de tráfico.
2. *La detección de fallos y localización/identificación en la capa óptica* se basa en la monitorización de tasa de error de bits (BER). Para abordar este objetivo se

proponen dos algoritmos denominados BANDO y LÚCIDA, donde en primer lugar BANDO detecta cambios de BER significativos en las conexiones ópticas, y luego LÚCIDA identifica el patrón más probable de los fallos. Con el objetivo de detectar estos fallos, se proponen dos técnicas para dos momentos diferentes en la vida de una conexión óptica. La primera, está basada en monitorización activa y está indicada para el momento de puesta en servicio de una conexión. La segunda, se centra en la monitorización pasiva durante el tiempo en que la conexión está operativa en la red.

3. *Reconfiguración de la Red.* Se han diseñado dos algoritmos para ser ejecutados cuando se detectan anomalías en la capa L2 o degradaciones en la capa L0. Se propone el problema de optimización para reconfigurar la topología de red virtual, denominado ODEON, y el re-enrutamiento de las demandas cuando se hayan recibido ciertas notificaciones del algoritmo BANDO tras la detección de un cambio significativo en la BER algoritmo, denominado SCULPTOR.

4. *Arquitectura Cognitiva.* Se diseña una arquitectura de monitorización y análisis de datos (MDA) con el doble objetivo de reducir la cantidad de datos que se tienen que enviar al repositorio central y a la vez minimizar los tiempos de detección de anomalías y degradaciones. Se proponen ciertos casos de uso representativos en escenarios multicapa para redes autónomas y que se utilizan para validar experimentalmente la arquitectura MDA distribuida presentada en esta tesis doctoral.

5. *Técnicas de Visualización*. Se proponen técnicas de visualización que incluyen gráficas diseñadas para tareas específicas, con el objetivo de guiar a los operadores. Para que estas técnicas sean útiles, se han de procesar una abrumadora cantidad de datos de monitorización. Un caso de uso de localización de fallos se utiliza como hilo conductor.

Cabe destacar que parte del trabajo reportado en esta tesis doctoral se ha realizado en el marco de proyectos europeos y nacionales. En concreto, *METRO High bandwidth, 5G Application-aware optical network, with edge storage, compUte and low Latency* (METRO-HAUL), H2020-ICT-2016-2. (G.A. 761727) que ha sido financiado por la Comisión Europea, el *Service-oriented hYbrid optical NEtwork and cloud infrastructuRe featuring high throuGhput and ultra-low latency* (SYNERGY). Ref: TEC2014-59995-R, 2015-2017 y el *cogniTive 5G application-aware optical metro netWorks Integrating moNitoring, data analyticS and optimization* (TWINS) Ref: TEC2017-90097-R, 2018-2020, ambos financiados por el Ministerio de Economía, Industria y Competitividad (MINECO).

Table of Contents

List of Figures

List of Tables

Page

Chapter 1

Introduction

1.1 Motivation

Transport networks have been traditionally deployed as multilayer networks, where the optical layer provides connectivity to packet nodes adapting the traffic generated by the users to the huge capacity of the optical connections. This creates Virtual Network Topologies (VNT), since links connecting packet nodes are supported by optical connections. However, those optical connections were statically configured, as traffic aggregation at the packet layer translated into an almost constant capacity requirement.

However, such scenario has radically changed in the last few years since the difference between the traffic requested by the users has significantly increased by the introduction of new services, like video-on-demand, social media, video calls, on-line gaming, etc. In consequence, traffic dynamicity has arrived at the optical layer and optical connections need to be established dynamically; here the concept of Software Defined Networking (SDN) has facilitated such dynamic connection provisioning.

Another recent change for the transport networks is the concept of the *telecom cloud* [Ve15], where network operators are deploying data centers connected to their networks thus, opening the possibility to the introduction of new services, as well as reducing Capital Expenditures (CAPEX) by Virtualizing Network Functions (NFV).

Both, traffic dynamicity and the telecom cloud have added more complexity to transport networks infrastructures, making also their operation much more complex. In fact, the panorama is getting worse, as new players (e.g., Netflix, Amazon or even Google Play Films & TV) have recently arrived at the market forcing network operators to both, reduce prices and increment the quality of the

services and users' experience, as a result of the extreme competition. In addition, 5G networks, already becoming a reality, will bring even more new players to the scene and impose even more stringent requirements.

In this context, reducing CAPEX is not enough to survive as a player and much less to be a winner. Then, innovative solutions are needed to keep such CAPEX reduction, and also to significantly reduce Operational Expenditures (OPEX) coming from adding cognitive capabilities to the network to facilitate their self-configuration and self-adaptation, paving the way toward a true *autonomic networking*.

The very first step in this path is understanding what is going on in the network infrastructure, making better planning decisions and more efficient resource utilization, anticipating anomalies and degradations that might affect the committed Service Level Agreements (SLA), and localizing failures at their incipient phase thus, leaving plenty of time to reconfigure the network, while facilitating its maintenance.

Considering the network at the packet level, unexpected traffic increments can create network congestion and stress resource utilization in packet nodes. Those anomalous volumes of traffic do not follow expected patterns and hence, its prompt detection becomes essential since it allows preparing the network to minimize their impact on the running services.

Looking at the optical layer, a failure can impact on a huge number of services causing disruptions, which entails money losses for the network operators. Therefore, detecting failures in advance makes that services can be moved before any disruption happens.

In this PhD thesis, we face those challenges by devising data analytics algorithms that are used to help human operators and provisioning and planning tools by detecting anomalies and degradations and by supplying useful data that might improve their work. Data analytics need data that can be obtained by monitoring the network and their results can be used to do reconfiguration of the nodes or the network. Then, an additional challenge to be faced is to come up with a monitoring and data analytics architecture to support Observe-Analyze-Act (OAA) *control loops* at the node and the network level developing thus, the main ingredients for autonomic networking.

1.2 Goals of the Thesis

In light of the above, this Ph.D. thesis is focused on applying control loops as a way to bring cognition to multilayer transport networks. Specifically, detection of different incidences impacting the network, whenever directly involving the optical connections (*lightpaths*) at the optical layer or affecting traffic volume such as

traffic anomalies at the packet layer, will be tackled by proposing algorithms and control and management architectures. Anticipating their detection will provide additional time to plan adaptation to those unexpected events by reconfiguring the network aiming at reducing signal and packet losses. Not only detection, but also identification and localization, wherever possible, is targeted. Moreover, network reconfiguration triggered by the detection of incidences will be studied for the sake of completeness.

Five specific goals are defined to achieve this main goal:

G.1 –Traffic Anomalies at the packet layer

This goal focuses on detecting traffic anomalies at the packet layer and consists of two sub-goals:

- **G.1.1**: Study methods for the detection of single traffic anomalies.
- **G.1.2**: Study detection and reconfiguration methods in scenarios with multiple traffic anomalies.

G.2 – Failure detection and localization/identification at the optical layer

This goal targets at detecting and localizing/identifying soft failures that may impact the optical layer. The goal is divided into three sub-goals:

- **G.2.1**: Methods for soft failure detection.
- **G.2.2**: Methods for soft failure localization/identification.

G.3 – Network Reconfiguration

This goal aims at reconfiguring either the optical or the packet layer, or both after a failure or an anomaly is detected.

G.4 – Cognitive Architecture

This goal concentrates on studying different architectures for the control and management planes to bring cognition to multilayer transport networks.

G.5 – Visualization techniques to assist network operator

This goal focalizes in developing visualization techniques to help human operators to understand what is going on in the network.

A summary of the goals of the thesis is presented in Table 1-1.

Table 1-1: Thesis goals.

Goals	Sub-goals
G.1 **Traffic Anomalies at the packet layer**	**G.1.1** Single traffic anomalies.
	G.1.2 Multiple traffic anomalies.
G.2 **Failure detection and localization/identification at the optical layer**	**G.2.1** Failure detection.
	G.2.2 Failure localization/identification.
G.3 **Network Reconfiguration**	
G.4 **Cognitive Architecture**	
G.5 **Visualization techniques**	

1.3 Methodology

To carry out the studies needed to meet the goals of this thesis, the methodology illustrated in the next figure will be followed.

As the starting point of each study, an idea related to a thesis objective is conceived and formally stated. Then, due to the nature of this thesis' goals, both a data analytics algorithm and/or an optimization problem (mathematically formulated as a Mixed Integer Linear Programming (MILP)) are devised. The data analytics algorithms are implemented in R in RStudio framework, whereas the mathematical model is solved using IBM's commercial solver CPLEX [CPLEX]. Because the realistic scenarios tackled lead to large problem sizes in most of the studies, MILP formulations require long computation times; therefore, for those scenarios requiring very short computation times (e.g., once in operation or to manage dynamic connection requests) heuristic algorithms are required. To that end, once a MILP formulation is validated, an algorithm is designed.

Data analytics and optimization algorithms are together considered as our proposed solution to the problem. These algorithms are integrated in an event-

driven OMNeT++ [OMNet] simulator, which allows evaluating the performance of the solution and, if required, revising and improving the algorithms. Then, the performance of the solution is compared against a certain benchmark (e.g., other solutions based on state-of-the-art procedures). Relevant results are eventually disseminated and considered for the conception of a new idea requiring further research.

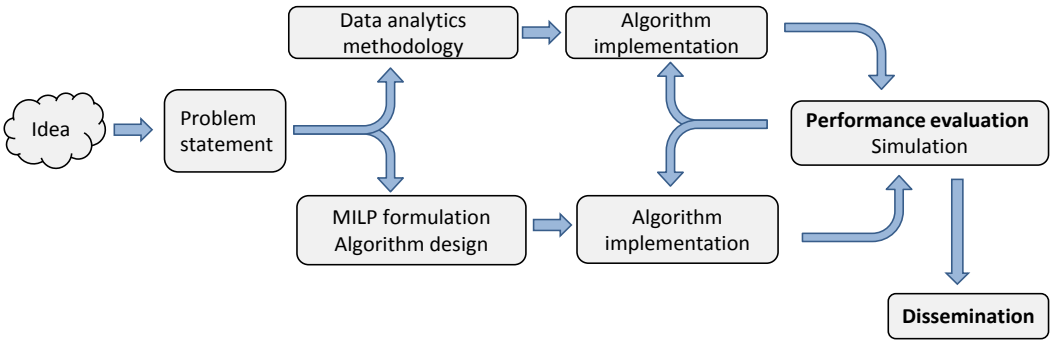


Fig. 1-1. Diagram detailing the followed methodology.

Finally, aiming at experimentally demonstrating the feasibility of the proposals in this PhD thesis, some of the algorithms have been integrated in an experimental platform.

1.4 Thesis Outline

The remainder of this thesis is organized as follows.

Chapter 2 introduces the needed background in networking and Chapter 3 in optimization, statistics and machine learning. That background is required to follow the main concepts of this thesis.

Chapter 4 briefly reviews the state-of-the-art related to the objectives of this Ph.D. thesis, focusing on detaching and highlighting the niches to be covered.

Chapter 5 deals with traffic anomalies and re-configuration at the packet layer targeting goals G.1.1 and G.3. Also, it sketches a cognitive architecture focusing on goal G.4. This chapter is based on the following journal and conference publications: [ComCom17], [JLT17.1], [ECOC16.2], [ACP16] and [ICTON17.2].

Continuing with anomaly detection, Chapter 6 targets goal G.1.2 of multiple traffic anomalies detection; it is based on the journal [ComCom17] and conference [ACP16] publications.

Focused at the optical layer, Chapter 7 aims at achieving bit error rate (BER) degradation detection and failure identification covering goals G.2.1 and G.2.2. The following publications are related to this chapter [JLT17.2], [ICTON17.1], [OFC17.1], and [ECOC16.1].

Continuing with soft failure detection, Chapter 8 suggests performing network reconfiguration after such detection, thus targeting goal G.3. This chapter is related to the conference [OFC17.1] publication.

Localization after soft failure detection is targeted in Chapter 9, related to goals G.2.1 and G.2.2. This chapter is based on the journal [JOCN18] and conferences [OFC18.2], [ICTON18.1] and [ICTON18.2].

Chapter 10 validates the distributed data analytics architecture targeting goal G.4, by demonstrating three representative use cases. It is based on the journal [JLT18] and conference [OFC17.2].

Chapter 11 relates to goal G.5 and devises several visualization techniques as a tool to guide operators in the process of failure localization. This work is based on the already published conference [OFC18.1] and the just submitted conference [ECOC18.1] and demonstration [ECOC18.2].

To conclude, Chapter 12 draws the main contributions of this PhD thesis.

1.5 Contributions and References from the Literature

For the sake of clarity and readability, references contributing this thesis are labeled using the following criteria: [<conference/journal acronyms> <Year (*yy*)[.autonum]>], e.g., [OFC17] or [JLT18]; in case of more than one contribution with the same label, a sequence number is added.

The rest of the references to papers or books, both auto references not included in this thesis and other references from the literature are labeled with the initials of the first author's surname and year of publication, e.g., [Bi06]. Finally, references to norms or standards are labeled with its identification, e.g., [RFC7491].

Chapter 2

Background in Networking

As a starting point, let us introduce in this chapter the main concepts that are used over the rest of this PhD thesis. Firstly, an overview of telecom networks introducing the access, metro and optical segments is presented. Multilayer networks are considered, where virtual network topologies created on top of the optical layer. A brief introduction to control and management of telecom networks is presented. The optical layer is then examined, and related concepts are introduced. Since optical components can experience degradations that could affect the optical signal, the optical layer should be monitored, including the spectrum of the optical signals.

2.1 Telecom Networks

Optical networks are a type of networks using light to convey data between two (or more ends) and optical fibers as a transmission medium. A *laser* is an optical device that converts electrical signal received into light pulses and sends them through an optical fiber. An optical fiber consists of a cylindrical core of silica with a refractive index μ_1, surrounded by cylindrical silica cladding with a lower refractive index μ_2 in order to confine light.

Optical fiber cables are currently widely deployed and used in all telecommunications networks. Compared with copper cables previously used, optical fiber offers huge bandwidth and reduced losses and sensitivity to electromagnetic interferences among other undesirable effects [Ra10]. Optical networks satisfy demand for bandwidth due to the tremendous growth of connected users and devices, and the use of popular bandwidth hungry applications, like video on demand.

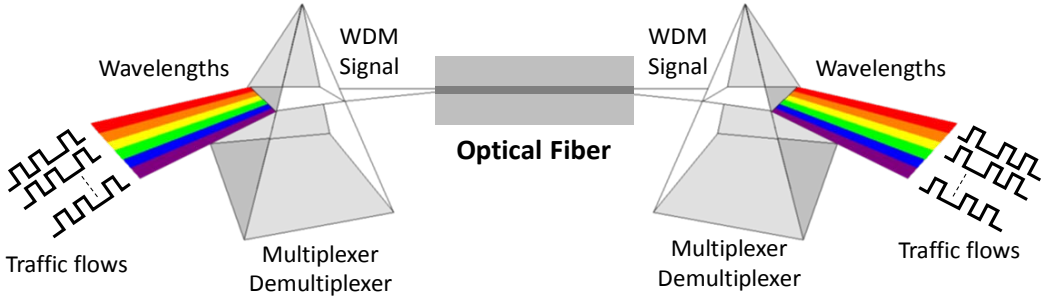


Fig. 2-1. Point-to-point optical transmission system.

The first generation of optical systems was based on point-to-point transmission (Fig. 2-1), where only lasers, and Optical Amplifiers (OAs) to amplify the optical signal at intermediate points, were used to provide capacity between two distant locations. Optical signals can be modulated in a specific central frequency in the optical spectrum and passive multiplexors allow multiplexing several optical signals to create a bundle that is eventually injected to the optical fiber; this is named Wavelength Division Multiplexing (WDM). In this generation, switching was performed electronically in the *packet nodes*.

In the second generation, optical switching was introduced. Wavelength Selective Switching (WSS) enable switching signals between optical fibers on a per-wavelength basis. As a result of the introduction of switching capabilities, real optical networks can be created, and some intelligence is needed to create and route optical connections (*lightpaths*) traversing several nodes between the source to the to the destination node [Ra10]. Such optical nodes performing optical switching are named (Remotely configurable) Optical Add-Drop Multiplexers ((R)OADMs) or Optical Cross Connects (OXCs). Remote configuration became a must to be able to *program* the optical nodes from a remote center, to automatically establish/release lightpaths in/from the network.

In general, network operators split their transport networks into several segments, to better capture the specific need of each of them. Fig. 2-2 presents an example of such split, which include:

- *Access Network* that connects users in a small area to their network operator and aggregate traffic to fit users' traffic flows into network connections.
- *Metro Network* interconnects and aggregates access networks, while connecting to the core network. Metro nodes are conceived as a combination of optical transmission and switching, a packet switching, and small data center, together with Passive Optical Networks (PON) and other access specific technologies.

- *Core Network*, or backbone network, is the part that performs the highest level of aggregation. It consists of OXCs and packet nodes, in addition to large data centers (DC) placed at some strategic locations.

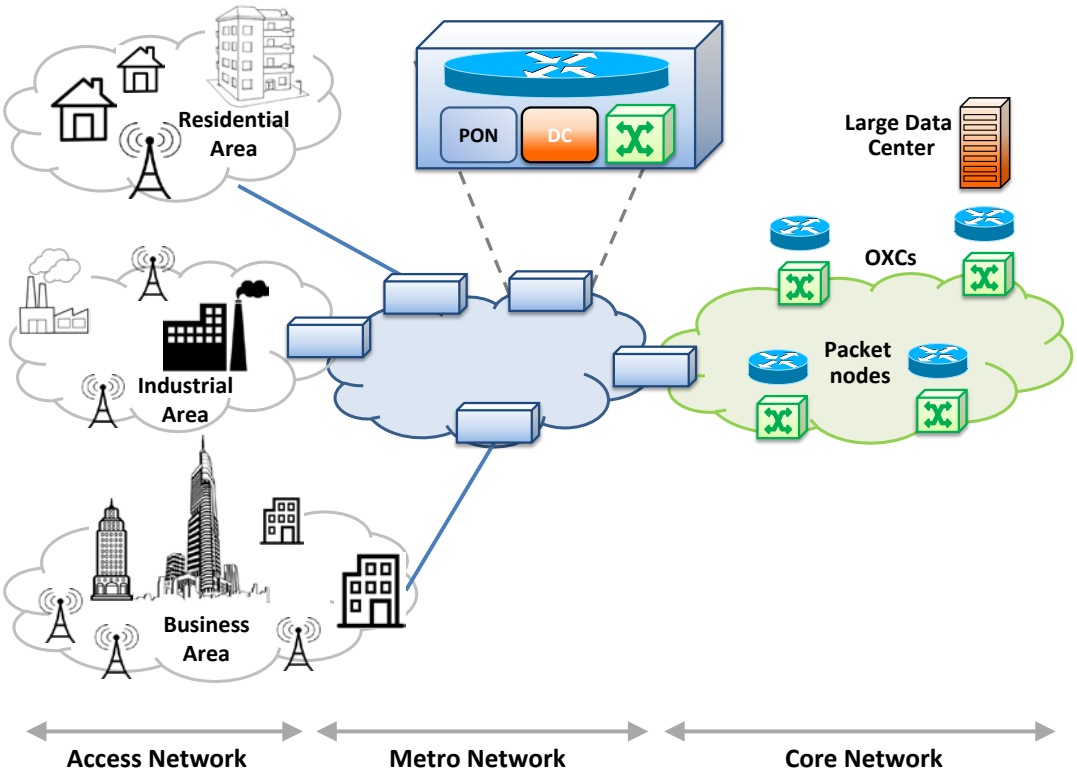


Fig. 2-2. Access, metro and core networks.

Metro and core networks are the focus of this PhD thesis, as they include optical and packet networks.

2.1.1 Optical Networks

Let us now focus on the most important physical components that can be found in optical networks [No11], [EON16]:

- *Transponder* is the element that sends and receives the optical signal from a fiber; it adapts the signal coming from a client layer (e.g., a packet switch) into an optical signal ready to be multiplexed in a WDM bundle. Similarly, in the reverse direction, it adapts the optical signal from the network into a signal for the client. A transponder is characterized by its data rate and the maximum distance that the signal can span (see Fig. 2-3). It has two parts, the *transmitter* that includes a semiconductor laser diode as light source, and a *receiver* where the optical signals are converted to the electrical domain using a photodetector.

In WDM, optical signals where conceived with a fixed spectrum width, e.g., 50 GHz, which limits the amount of data that can be conveyed. As an evolution, the concept of Elastic Optical Networks (EON) allow a variable spectrum width for the lightpaths, and therefore, transponders can generate elastic optical paths; with variable bitrates according to the current need [Ge12].

- *Optical Amplifiers* (OA) amplify optical signals without converting them the electrical domain. Although several types of optical amplifiers exist, erbium-doped fiber amplifiers (EDFAs) are the ones commonly used in intermediate locations to amplify WDM signals in long distance optical links.
- *Wavelength Selective Switching* (WSS) devices switch signals. Nowadays, WSSs are mostly based on Liquid-Crystal-On-Silicon (LCoS); they offer flexible passband filtering in conjunction with the wavelength switching and routing functionalities [EON16]. The specific functionalities of WSSs, include: multiple wavelength routing and switching, power equalization of different wavelength channels, dispersion compensation/mitigation, variable channel bandwidths, polarization-independent operation, compactness, and millisecond reconfiguration times.

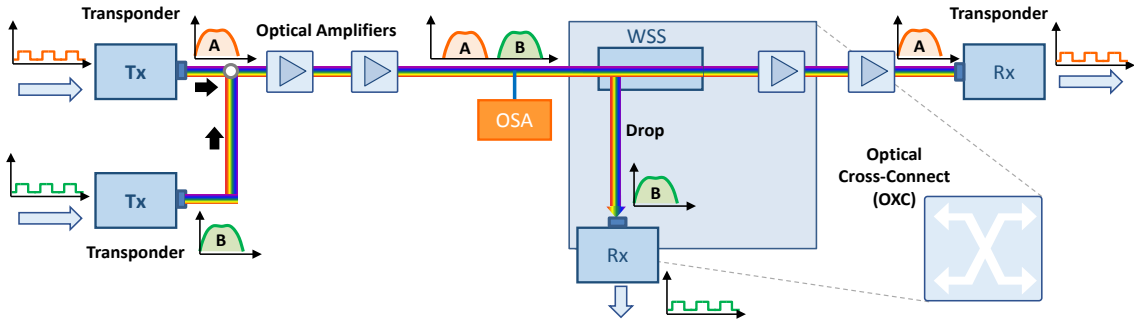


Fig. 2-3. Example of optical network, where optical components are illustrated.

A measure of the performance of the optical layer is that of the Quality of Transmission (QoT), which can be quantified in terms of BER of the lightpaths.

2.1.2 Virtual Network Topologies (VNT)

We target multilayer networks, where the optical layer includes OXCs and optical links with OAs, whereas the packet layer includes packet nodes and links are supported by lightpaths in the optical layer (Fig. 2-4).

The topology at the optical layer is statically configured, as optical links are created using physical components such as optical fibers and OAs. However, the topology at the packet layer can be dynamically configured, as the links are supported by lightpaths that can be dynamically established and released. In these conditions,

packet links are said to be virtual and denoted *virtual links* (vlinks), whereas the packet topology is said to be a VNT.

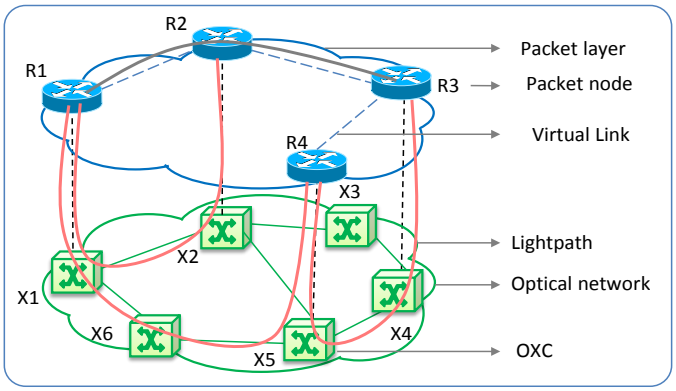


Fig. 2-4. Scheme for a VNT in a multilayer network.

In this PhD thesis, we assume that packet switches are based on Multiprotocol Label Switching (MPLS), which means that they support connections to be established at the packet layer, each of them between an *origin* (O) and a *destination* (D) packet switch and with some maximum capacity.

Increasing packet traffic dynamicity entails large overprovisioning, which increases costs and reduces margin to network operators. The solution to reduce overprovisioning lies in monitoring traffic to be able to predict future traffic conditions that, because known in advance, allow optimization algorithms to find optimal resource allocation, which can be implemented in the network in a proactive manner.

In addition, aiming at quantitatively measure the Quality of Service (QoS) at the packet layer, several metrics apart from pure number of bytes/packets can be considered, like packet loss, one-way or two-ways delay and differential delay or jitter. Note that QoS can be related to the Quality of Experience (QoE) as view from the service consumers'.

2.1.3 Control and Management Plane

As introduced above, transport networks consist of different network segments (metro and core) and layers (MPLS and optical), possibly from different vendors and using different technologies.

To enable automatic provisioning and management of heterogeneous services over such multi- vendor/domain/layer networks, a *control and management plane* is needed (Fig. 2-5). The control plane is in charge of network operation by automatically provisioning connections and reacting against failures in the network devices, while the Network Management System (NMS) at the management plane is responsible for providing Fault, Configuration,

Accounting/Administration, Performance and Security (FCAPS) management to the network. To enhance network operation, a robust infrastructure capable of performing complex operations, such as a SDN controller, is required at the control plane.

The SDN controller uses two differentiated databases to store operational data. The topology database (also known as the Traffic Engineering Database -TED) contains the state of network resources, whilst the connections database (or Label Switched Path (LSP) database) maintains information regarding current connections in the network; including the route, the bitrate or capacity, the spectrum allocation, the switching types, and other constraints.

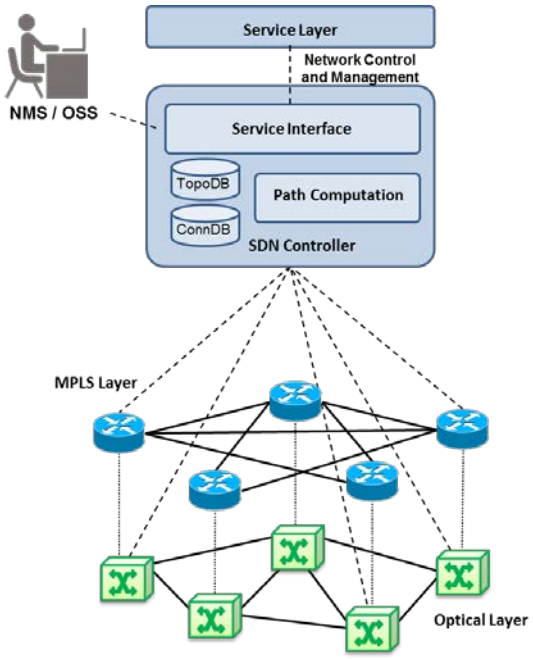


Fig. 2-5. Control and Management Plane

2.2 The Optical Layer

In this section, we concentrate on the specifics of the optical layer, including optical signals, monitoring, and soft failures.

2.2.1 The Optical Signal

Fig. 2-6 illustrates the spectrum of a WDM signal bundle (e.g., in an optical link) where lightpaths use a flexible spectrum allocation depending on the bitrates requirements; the width of the spectrum allocation, named *frequency slot*, is adapted to meet such requirements. Note that a frequency slot is created as a

group of neighboring frequency slices of fixed width, e.g., 6.23 GHz. In the example, spectrum allocations range from 37.5 GHz to 75 GHz.

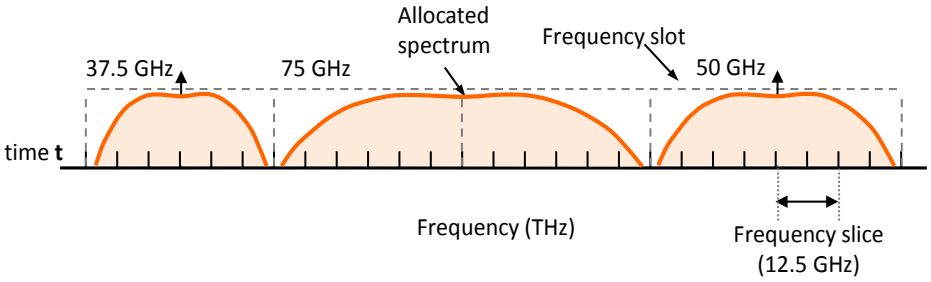


Fig. 2-6. Example of flexible spectrum allocation.

2.2.1.1 Modulation Formats and Optical Detection

Several modulation formats are available; its choice impacts on the reach and capacity of lightpaths and therefore, choosing the correct modulation format is primordial to build a flexible and cost effective high capacity optic-fiber network [Ja13]. The different modulation formats convey data by changing some particular aspect of a base signal, the *carrier*, in response to input data. There are three major classes of digital modulation techniques used for transmission of digitally represented data: ASK, FSK and PSK.

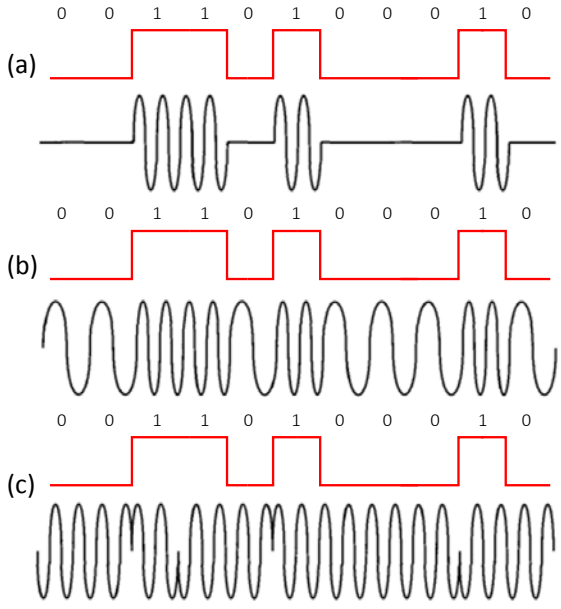


Fig. 2-7. Examples for modulation formats: (a) ASK, (b) FSK, (c) PSK.

- Amplitude-Shift-Keying (ASK), where digital data are represented as variations in amplitude of the carrier wave, also known as On-Off-Keying (OOK) (see Fig. 2-7(a). OOK will be later used in Chapter 9.

- Frequency-Shift-Keying (FSK) is based on switching the frequency of a laser light between two different frequencies to represent 0s and 1s (Fig. 2-7(b)).
- Phase-Shift-Keying (PSK) uses the phase of the signal to modulate a signal. The modulation happens by varying the sine and cosine inputs at a precise time Fig. 2-7(c).

Later, in Chapter 7, the concept of Polarization-Multiplexed (PM) or Dual-Polarization (DP) is a method that allows extending the bandwidth of optical signals transmitting two waves using orthogonal polarization states over the same carrier frequency.

Different optical detection techniques can be considered (e.g., direct, coherent). Although the particular type of signal detection is irrelevant for this thesis, the experiments entailing optical transmission use coherent detection. Coherent detection is a method developed for detecting information encoded as modulation of the phase and/or frequency of electromagnetic radiation ranging from infrared to visible light. It was first developed for radio and microwaves. The receptor receives a weak signal as input, which is later mixed with a strong reference light from a local oscillator by means of a nonlinear device e.g., rectifier. Finally, the mixed result is detected [wrp].

2.2.1.2 Spectrum of Optical Signals

As above introduced, the optical spectrum can be represented as the decomposition of the power (dBm) as a function of the frequency (Hz). A related concept is that of the *optical bandwidth*, which is the width of the optical spectrum.

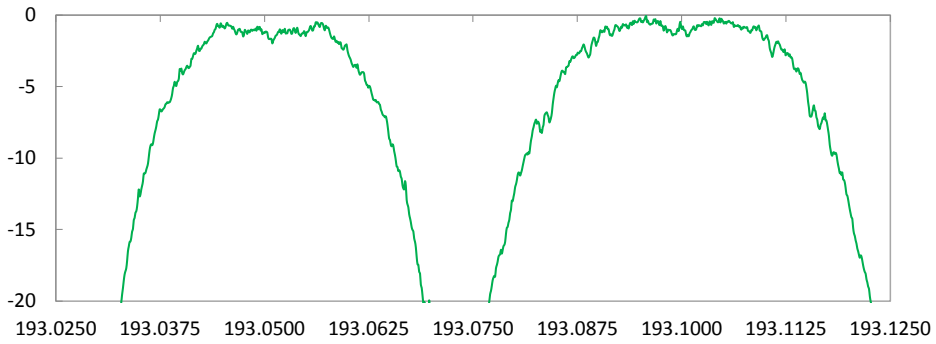


Fig. 2-8. Real QPSK optical spectrum acquired by an OSA.

Fig. 2-8 presents an optical spectrum with two neighboring Quadrature Phase-Shift Keying (QPSK) 100Gb/s signals (s1 and s2) acquired by an Optical Spectrum Analyzer (OSA). OSAs are devices that can be used to acquire the optical spectrum of a WDM bundle; the spectrum measurements can be also exported from the OSA as ordered lists of <*frequency*, *power*> pairs. In particular, signal s1 in Fig. 2-8 was generated using an experimental system while signal s2 was generated by a commercial one. Note that a band guard is left between both signals to avoid them

to overlap. In general, QPSK optical signals present a flat spectral region around the central frequency, sharp edges, and a round region between the edges and the central frequency.

2.2.2 Monitoring

Monitoring the optical layer has been traditionally limited to metering optical power at intermediate points, to detect whether some optical signal is received or in the contrary, to detect fiber cuts, and at the receiver side as an indirect way to determine whether the quality of the transmission is enough.

Digital Signal Processing (DSP) module of coherent receivers has posed new possibilities for monitoring [Do15], [Do16], since they enable get measurements of several parameters related to the optical signal. Such monitoring capability is attracting increasing interest for several reasons such as: *i*) the reduction of system margins (which derives in reducing capital expenditures) might induce more frequent degradations at the optical layer [Po17], [So17]; *ii*) a more accurate estimation of the QoT and an optimization of transmission parameters, routing, and spectrum assignment [Ch15].

Monitoring the optical layer is vital to verify SLAs fulfillment. Considering a multilayer scenario like the one depicted above, packet losses could come from errors affecting the optical layer therefore, leading to unacceptable QoS and to SLA violations. Such violations represent money losses for the network operator.

Monitoring the ends of the lightpaths allows detecting, and somehow identifying failures affecting lightpaths. Such failures can be classified as *hard* and *soft* failures, where hard failures are unexpected events that suddenly interrupt the established connections, whereas soft failures are events that progressively degrade the quality of the signal. It is key to localize any failing element in order to make operational decisions, like reroute lightpaths. In fact, in general, it is not enough monitoring the QoT at the ends of the lightpaths, since this does not allow localizing the element that causes the failure.

Several parameters can be measured at the optical layer; above all, pre-FEC BER, Optical Signal to Noise Ratio (OSNR), Q-factor, and electrical SNR can be monitored by currently available commercial transponders. Moreover, other parameters can be monitored: e.g., chromatic dispersion through equalizer taps [Co13], the central frequency of the signal thanks to an automatic frequency control [Cu13], polarization channel characteristics and the state of polarization [Ciena]. Such measurements can be performed with a time period of 10 ms and can also be used for failure prediction applications allowing operators to pre-empt outages [Ciena].

QoT can be roughly estimated as a function of the links and nodes traversed by each optical connection. This is useful information since it can be used to configure

a BER threshold at connection set-up, which would help to detect BER degradation by comparing the actual measured BER against it. However, if the threshold value is set to a value too close to the actual BER, many threshold-crossing notifications would be raised because of small BER changes, which, in addition to add control overhead, do not give useful information. On the contrary, if the threshold value is relaxed, e.g., closed to the equipment max BER, degradation detection could not be anticipated early enough the transmission is totally disrupted.

Another useful way to monitor the performance of the network, especially for in-line optical amplifiers, is the Optical Supervisory Channel (OSC). The OSC is an additional data channel that uses a specific wavelength outside the ones used for carrying the actual traffic. It is used to verify that an optical link is working properly by performing control and management functions along with monitoring the performance of amplifiers along the link for failure detection [Ra10].

2.2.3 Soft Failures

This section presents four different soft failures that could affect the QoT of an optical connection (Fig. 2-9):

- *Signal Overlap* (SO) happens when the spectrum allocation of an optical connection invades that of a neighboring one. This might be caused by the inaccuracy in the central frequency of the laser and/or the filters of one of the connections.
- *Filter Tightening* (FT) appears when there exists a central frequency misalignment or a width inaccuracy in the filters along the route of an optical connection. Besides, Fig. 2-10 presents four different possible causes for FT, where Fig. 2-10(a) shows filter F2 is misaligned, F2 width is narrower than required frequency slot width in Fig. 2-10(b), filters F2 and F3 are misaligned in Fig. 2-10(c), and the central frequency of the signal is misaligned in Fig. 2-10(d).
- *Gradual Shift/Drift* in Fig. 2-9(c) appears when either the optical signal in the case of a *Laser Drift* (LD), or the filter in the case of a *Filter Shift* (FS), gradually deviate from the central frequency determined at set-up time;
- *Cyclic Shift/ Drift* in Fig. 2-9(d) occurs when a gradual drift describes a cyclical movement with time, *Cyclic Laser Drift* (cLD) or *Cyclic Filter Shift* (cFS).

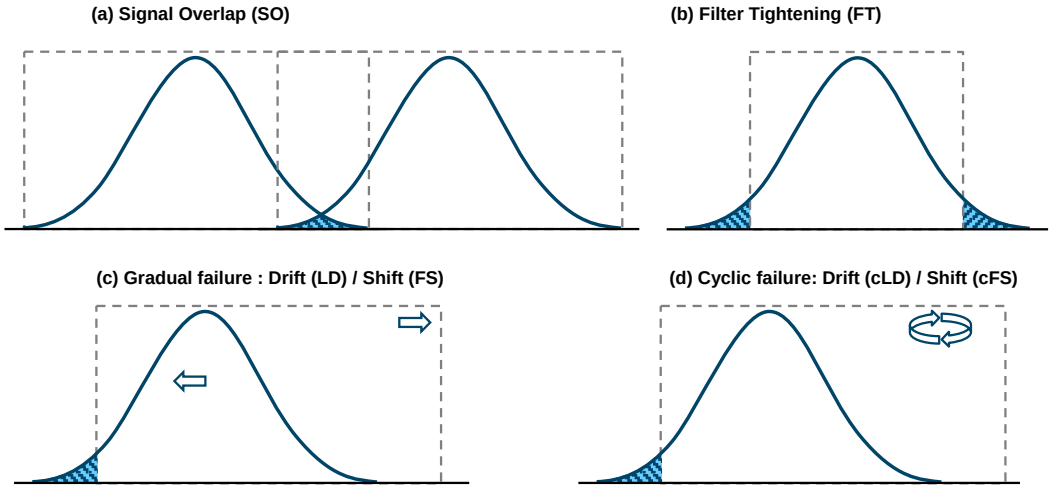


Fig. 2-9. Four failures affecting the signal of an optical connection. (a) Signal Overlap, (b) Filter Tightening, (c) Gradual Drift/ Shift, and (d) Cyclic Drift/ Shift.

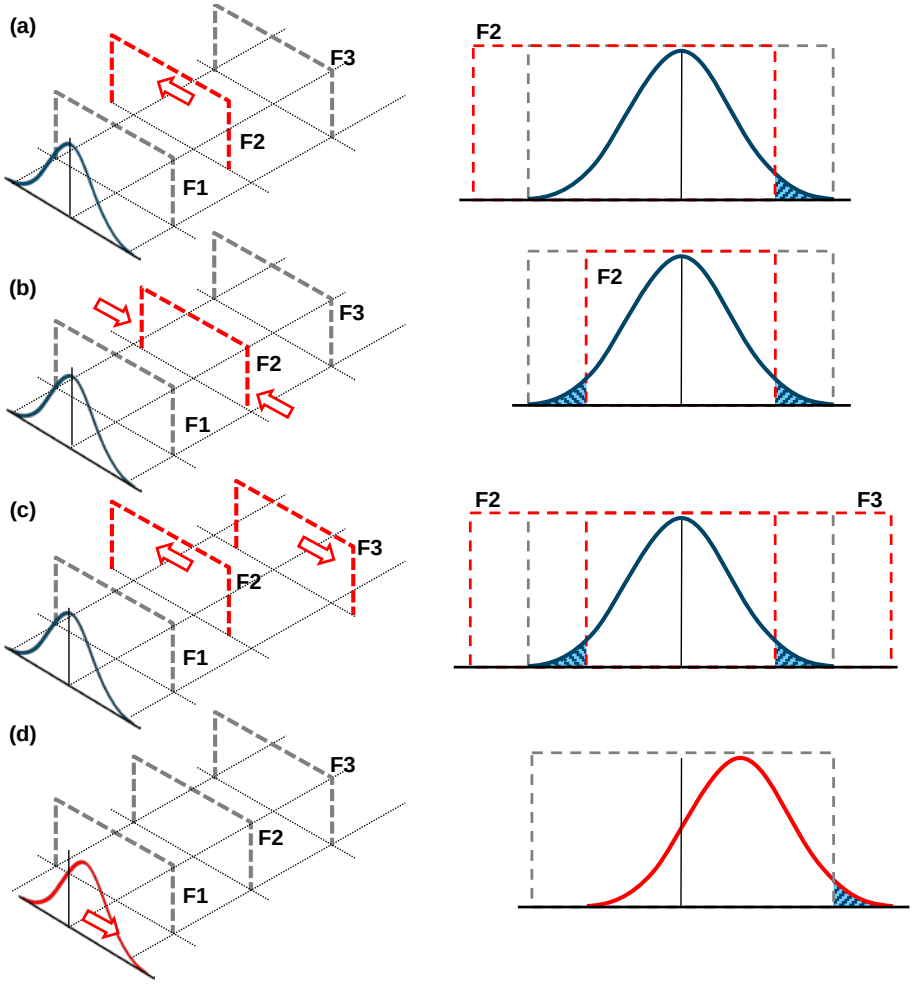


Fig. 2-10. Causes of Filter Tightening: (a) F2 misalignment, (b) F2 narrower, (c) F2 and F3 misalignment, and (d) central frequency misalignment.

2.2.4 Optical Spectrum Analysis

In this section, we focus on obtaining the most illustrative features of a QPSK signal optical spectrum. As real measurements are not always available (such as Fig. 2-8), simulators such as VPI [VPI] allow to generate and measure the optical spectrum at different points of a testbed. Besides, different OSA granularities can be emulated, as it can be seen in the different plots of Fig. 2-11.

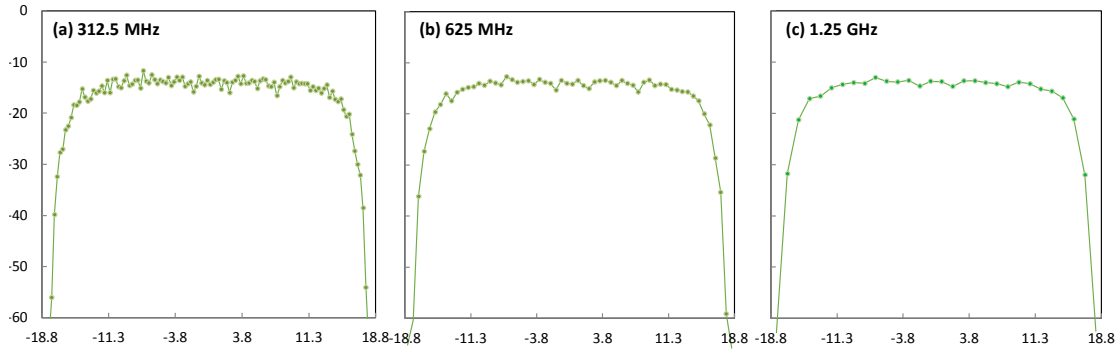


Fig. 2-11. VPI Simulation of a QPSK optical spectrum acquired by an OSA for different granularities: (a) 312.5 MHz, (b) 625 MHz, and (c) 1.25 GHz.

Let us consider the characterization of an optical spectrum by extracting its main features. In particular, we examine the optical spectrum of a simulated 100Gb/s Dual Polarization - Quadrature Phase Shift Keying (DP-QPSK) modulated signal acquired by an OSA with 625 MHz granularity presented in Fig. 2-13(a). In general, QPSK optical signals present a flat spectral region around the central frequency, sharp edges, and a round region between the edges and the central frequency. When the signal is properly configured, its central frequency should be around the center of the assigned spectrum slot to avoid filtering effects, and it should be symmetrical with respect to its central frequency. On the contrary, when a misconfiguration occurs, the optical spectrum is affected as shown in Fig. 2-12. In the case of filter shift (Fig. 2-12(a)), the optical spectrum would be asymmetrical, in the case of filter tightening, (Fig. 2-12(b)) the edges of the optical spectrum would get noticeably rounded due to the degradation in both sides of the slot, and in the case of laser drift, the central frequency of the signal would be shifted with respect to the assigned slot; (Fig. 2-12(c)).

In order to detect the above distortions, a module, named FeX in [JOCN18], primarily pre-processes the optical spectrum of the signal, which formally consists of an ordered list of frequency-power (<*f*, *p*>) pairs. The first pre-processing step consists in equalizing power, so the maximum power to be 0 dBm. Then, the derivative of the power with respect to the frequency is computed. Fig. 2-13(b) illustrates the derivative of the example optical signal; note that sharp convexity is observed close to the edges.

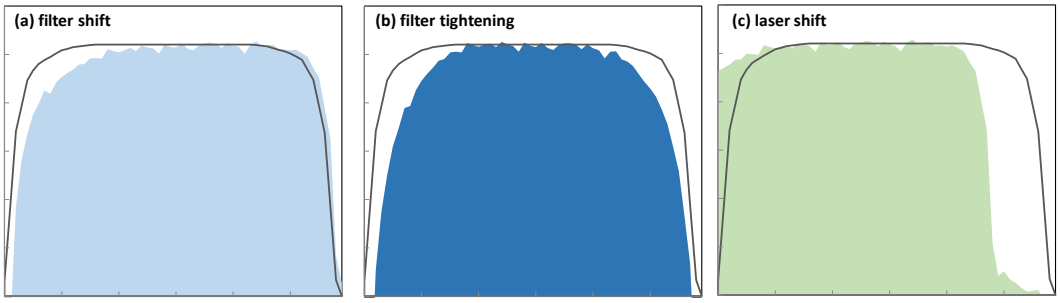


Fig. 2-12. Solid black line represents the spectrum of a non-degraded signal. Solid areas represent the spectrum of signals affected by different soft failures: (a) filter shift, (b) filter tightening, and (c) laser shift.

After pre-processing, the FeX module characterizes the mean (μ) and the standard deviation (σ) of the power around the central frequency ($fc \pm \Delta f$), as well as a set of primary features computed as cut-off points of the signal with the following power levels: *i*) equalized noise level, denoted as *sig* (e.g., -60dB + equalization level); *ii*) edges of the signal computed using the derivative, denoted as ∂; *iii*) a family of power levels computed with respect to μ minus $k\sigma$, denoted as $k\sigma$; and *iv*) a family of power levels computed with respect to μ minus a number of dB, denoted as *dB*. Each of these power levels generates a couple of cut-off points denoted as $f1_{(\cdot)}$ and $f2_{(\cdot)}$. In addition, the assigned frequency slot is denoted as $f1_{\text{slot}}$, $f2_{\text{slot}}$.

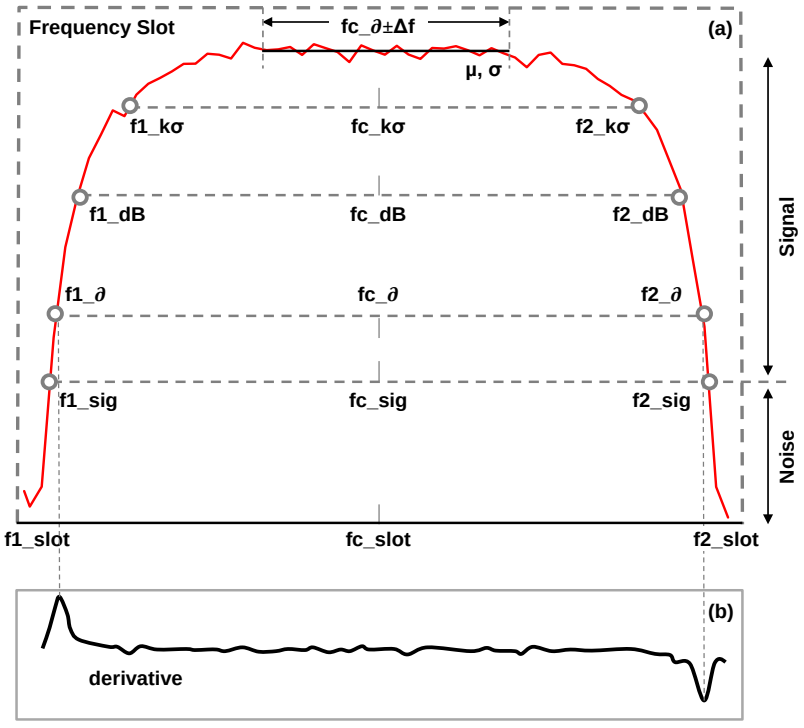


Fig. 2-13. Relevant signal points (primary features).

Although features have been computed from an equalized signal, note that signal distortion due to filter cascading effect has not been corrected yet. As previously introduced, this effect might induce to a wrong diagnosis of a filter problem for a normal signal. To overcome this drawback, a *filter mask* can be used to compensate the effect that a normal signal would suffer after passing a defined number of filters before computing its features. Filter masks can be easily obtained by means of the theoretical signal filtering effects or experimental measurements taken for a distinct number of cascaded filters.

Other features are computed as linear combinations of the primary features focus on characterizing a given optical signal; they include: *i*) bandwidth, computed as $bw_{(\cdot)}=f2_{(\cdot)}-f1_{(\cdot)}$; *ii*) central frequency, computed as $fc_{(\cdot)}=f1_{(\cdot)}+0.5*bw_{(\cdot)}$, as well as the shifting of the central frequency $\Delta fc_{(\cdot)}=fc_{(\cdot)}-fc_{(slot)}$; and *iii*) symmetry with respect to a reference (frequency slot or derivatives), computed as $sym_{(\cdot)\text{-ref}}=(f1_{(\cdot)}-f1_{ref})-(f2_{ref}-f2_{(\cdot)})$.

These features can be used as input for algorithms that can identify soft failures and their magnitude.

2.3 Conclusions

The purpose of this chapter has been giving the needed background on networking to ease the comprehension of some concepts used in this PhD thesis. Starting from a quick survey of ideas related to telecom networks are introduced, a deep study of the optical signal concerning the optical spectrum and modulation formats was carried out. Likewise, the concept of soft failure and possible causes is explained to facilitate the comprehension of further Chapter 7, Chapter 8 and Chapter 9.

The following chapter is focused on reviewing optimization problems continuing within an exhaustive presentation of Machine Learning (ML) algorithms that are used through this PhD thesis.

Chapter 3

Background in Optimization, Statistics, and Machine Learning

Basic concepts related to optimization and statistics are introduced in this chapter, as they are used to tackle specific problems through this PhD thesis and are also an essential part of ML algorithms subsequently introduced. Last but not least, visualization techniques are motivated, and some visualization charts needed in further chapters are presented.

3.1 Optimization

Mathematical programming or *optimization* is a mathematical method to find an *optimal* point x^* that results into the minimum (or maximum) value of a function $f(x)$ while satisfying a set of constraints [Ch83]; such point x^* is said to be optimum. More formally, an optimization problem can be defined as follows:

$$\textit{subject to} \left| \begin{array}{l} min \quad z = f(x) \\ g_i(x) \geq b_i, \quad \forall i \in C \end{array} \right. \tag{3.1}$$

where x represents a vector of variables, $f(x)$ is the objective function, and C represents the set of constraints. A constraint is an inequality defined by a function $g_i(x)$ and a constant b_i.

Let us define X as the set of all possible x vectors. Then, we can define the set S of feasible solutions of the problem, as follows:

$$S = \{x' \in X \mid g_i(x') \geq b_i, \forall i \in C\} \tag{3.2}$$

i.e., S contains all elements in X satisfying the whole set of constraints. Note that a problem could have alternative optimal solutions, i.e., several x^* with the same z^* value. Therefore, we can define the set of optimal solutions X^* as:

$$X^* = \{x' \in S \mid f(x') \leq f(x''), \forall x'' \in S\} \tag{3.3}$$

The problem, however, could have no feasible solution, i.e., $S = \emptyset$; in such case the problem is unfeasible. Finally, when $f(x^*) = -\infty$ the problem is unbounded.

A *Linear Programming* (LP) problem is a special case of mathematical programming, where $f(x)$ and $g_i(x)$ are linear functions of real variables. When variables are restricted to be integer, the problem is called *Integer Linear Programming* (ILP), whereas if the problem combines integer and real variables, the problem is defined as *Mixed Integer Linear Programming* (MILP).

Exact procedures have been developed to solve mathematical programming problems. For example, the *simplex algorithm* is used for solving LPs, whereas *Branch&Bound* and *Branch&Cut* algorithms are used to solve ILPs.

Although the output of these exact methods is the optimal solution, the required computation time tends to be too high for practical purposes when real-life instances need to be solved, even in the case of using powerful solver engines, such as CPLEX [CPLEX]. Thus, heuristic algorithms have been proposed to provide near optimal solutions to optimization problems. A heuristic is an algorithm to obtain feasible solutions, where the guarantee of finding an optimal solution is sacrificed for the sake of getting good solutions in a significantly reduced amount of time.

3.2 Statistics

Some concepts related to statistics are needed not only to introduce concepts related to problems addressed in this PhD thesis, but also for Machine Learning (ML). Further details of the contents presented in the following section can be found in [Sm08] and [Bi06].

3.2.1 Random Variables

A random variable X is a probabilistic variable whose possible values are outcomes of a random phenomenon, e.g., casting a dice or in coin tossing experiment; different outcomes depend on some physical effects yet not understood. In the first case, there are six possible outcomes $X = \{1,2,3,4,5,6\}$ equally likely to occur. By means of probability theory, we can model uncertainty in the outcome of such experiments stating that 1 would occur with probability 1/6. In contrast, the outcome when flipping a coin is not numerical, e.g., heads or tails, so it is useful to

associate numerical values to the outcomes via a random variable. For example, let us consider a random variable X take on value +1 whenever the coin lands heads and -1 otherwise.

Let us now introduce some important concepts to characterize a random variable: the mean, the variance, and the standard deviation.

The *mean* (μ), also known as the expected value (E($\cdot$)), refers to a single measure of the central tendency of a probability distribution. If we consider X as a random variable, f: $\mathbb{R} \rightarrow \mathbb{R}$, $f(X)$ is also a random variable with mean:

$$\mathrm{E}\left(f\left(X \right) \right) = \int f\left(x \right) dp\left(x \right) \tag{3-4}$$

If X is a discrete random variable, then:

$$\mathrm{E}\left(X \right) = \sum_{x} xp\left(x \right) \tag{3-5}$$

The *variance* (σ^2 or Var($\cdot$)) of a random variable X is defined as follows:

$$Var\left(X \right) = \mathrm{E}\left[\left(X - \mathrm{E}\left[X \right]^2 \right) \right] \tag{3-6}$$

Measuring how much on average X deviates from its expected value. Note that the concept of *standard deviation* (σ) is often used, which is defined as the square root of the variance.

3.2.2 Probability Distributions

Besides the mean and the variance, random variables can be deeply characterized by probability distribution functions. Roughly speaking, these are functions that return the expected probability for every possible value of the variable under study. A very common continuous probability distribution is the *normal* or *Gaussian* distribution, ($\mathcal{N}$ (μ,σ^2)) which is important in statistics and are often used to represent real-valued random variables. In fact, normal distributions with zero mean and unitary variance ($\mathcal{N}$ (*0,1*)) are used several times along this PhD thesis.

The *domain* of the random variable defines the type of function that could be obtained.

Discrete: taking on a finite number of values, where the assignment of probabilities is called a *Probability Mass Function* (PMF) characterized by the fact that they are non-negative and must sum to one. Considering the above example of tossing a coin, if heads and tails are equally likely, then the random variable χ described above takes on values +1, -1 with probability 0.5, i.e., P(X =+1) = 0.5, and P(X =-1) = 0.5.

Continuous: taking on an infinite number of values, where the assignment of probabilities is called *Probability Density Function* (PDF), defined as:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{3-7}$$

In particular, in this PhD thesis we use the *Cumulative Distribution Function* (CDF) that gives us the probability that random variable X will take a value less than or equal to x'. For a real valued random variable X with PDF $f(x)$, the associated Cumulative Distribution Function F(x') is given by:

$$F(x') = \mathrm{P}\{X \le x'\} = \int_{-\infty}^{x'} df(x) \tag{3-8}$$

The CDF $F(x')$ allow us to efficiently perform range queries on p. For instance, by integral calculus, we obtain:

$$\mathrm{P}(a \le X \le b) = \int_{a}^{b} df(x) = F(b) - F(a) \tag{3-9}$$

In particular, the values of x' for which $F(x')$ assumes a specific value, such as 0.1 or 0.5 have a certain name, *quantile* of the distribution p.

In some special cases we are interested in obtaining the inverse F^{-1} defined when F is strictly increasing and continuous, then $F^{-1}(p)$ is equal to the unique real number x that satisfies $F(x) = p$ such that $p \in [0,1]$.

As an illustrative example, Fig. 3-1 shows the PDF (see Fig. 3-1(a)) and the CDF (Fig. 3-1(b)) for the normal distribution $\mathcal{N}(0,1)$.

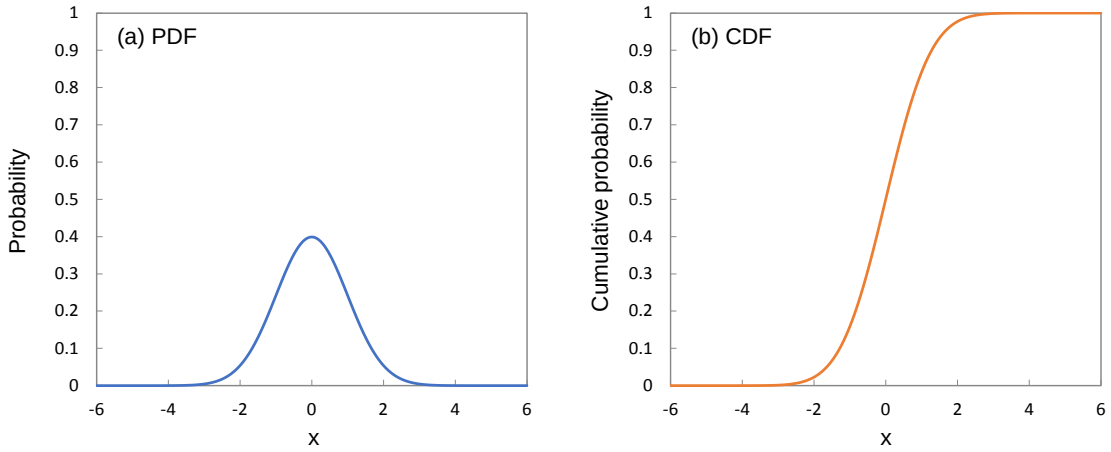


Fig. 3-1. (a) PDF and (b) CDF for the normal distribution with zero mean and unit variance.

3.2.3 Statistical Modeling

So far, we have focused on characterizing a random variable, using well known functions. However, it is also possible to characterize a random variable by obtaining a certain model that explains its behavior. The most basic model is *linear regression*; this model aims at finding the relationship between a dependent variable (y) and a single or multiple independent predictor variable(s) (x_i).

Consider the statistical model presented in eq. (3-10) to predict the dependent variable y. The general model consists on a deterministic term computed as a function of the *independent variables,* x_i, and a random term ε.

$$y = f(x_1, x_2, ..., x_n) + \varepsilon \tag{3-10}$$

Two good properties that a statistical model should satisfy are the following:

- *Unbiased estimation:* it is satisfied when the expected value of the dependent variable is the deterministic term (e.g., $E(y)=f(x_1,x_2,\ldots,x_n)$). Equivalently, we can consider the expected value of the random component as 0, $E(\varepsilon)=0$.
- *Known error distribution*: the random term of the model, ε, should fit with a known distribution such as the Gaussian distribution.

Note that in the definition in eq. (3-10) we are considering that y and x_i are different types of variables; however, there are particular situations, such as *time series*, where y and x_i variables represent the same variable at different time moments. Let us reformulate eq. (3-10) to cover such case:

$$y(t) = f(y(t-1), y(t-2), .., y(t-n)) + \varepsilon \tag{3-11}$$

Simple Linear Regression

As previously stated, the simplest form of linear regression relates the independent and dependent variables is a straight line [La13], i.e.,

$$y = a_1 \cdot x + a_0, \tag{3-12}$$

where y is the dependent variable, the one that we want to model according to the input data x; the unknowns are then, coefficients a_1 and a_0. Fig. 3-2 presents an illustrative example of simple linear regression.

Let us denote $\hat{y}_i$ the predicted value for data point i. Then, the difference between the training and predicted values, i.e., (y_i - $\hat{y}_i$), represents the error between the real and the modeled value.

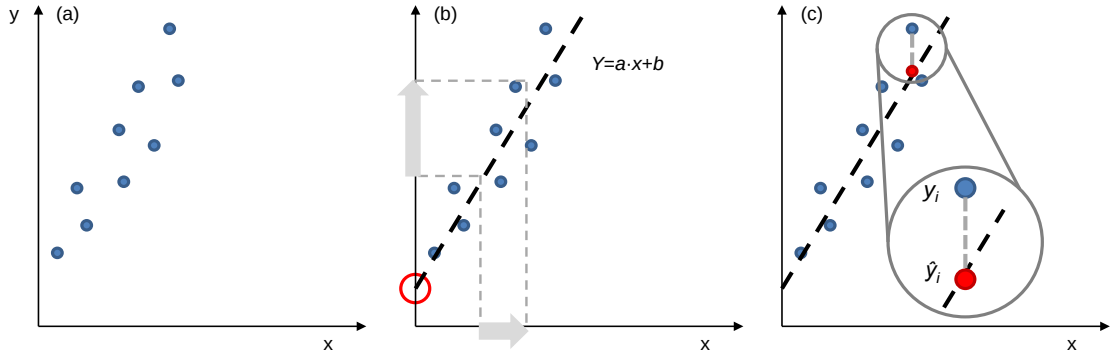


Fig. 3-2. (a) Real data points, (b) linear regression is applied obtaining a linear model, and (c) distance between real and predicted data.

The regression algorithm is based on solving an optimization problem to minimize the squared error, i.e., the sum of the square of the differences between the predicted and the real points:

$$\min\left(\sum_i \left(y_i - \widehat{y_i}\right)^2\right) \tag{3-13}$$

Multiple Linear Regression

The case where there is more than one single independent variable is known as *multiple* linear regression; it normally follows the form:

$$y = \sum_{i=1}^{n} a_i \cdot x_i + a_0 \tag{3-14}$$

Note that non-linear relationships between dependent and independent variables can be considered using multiple linear regression. For instance, the linear relation between x and y in eq. (3-12) can be extended to a k-th degree polynomial in x by creating k variables each one representing one power of x from 1 to k:

$$y = \sum_{i=1}^{k} a_i \cdot x^i + a_0 \tag{3-15}$$

Generalized Linear Model (GLM)

The traditional linear models predict the expected value of a random variable as a linear combination of a set of observed values. Thus, a constant change in a predictor leads to a constant change in the response variable; note that this is only satisfied when the response variable has a normal distribution.

For other more complex scenarios, these assumptions are not satisfied, hence, the GLM arises as extension of traditional linear models. Eq. (3-16) shows the generic

form of a GLM, where g is named *link function* and relates the linear predictor to the response variable [Cu89].

$$y = g^{-1}\left(\sum_{i=1}^{n} a_i \cdot x_i + a_0\right) \tag{3-16}$$

For response variables distributed according to distributions different than the Gaussian, (e.g., binomial, Poisson, gamma, etc.), link functions range from the simple identity function in the case of a simple linear regression to complex relation of logarithmic transformations such as the logit function or the Box-Cox transformation.

3.2.4 Goodness of Fit

The goodness of fit describes how well a statistical model fits a set of observations. Although several measures of goodness of fit have been defined, in the following we only introduce the ones used in this PhD thesis.

Coefficient of Determination R^2

The coefficient of determination, also denoted as R^2, is a statistic used in the context of statistical models that shows the proportion of the variance in the dependent variable (y_i) that is explained from the independent variables (x_i). It aims at providing a measurement of how well observed outcomes are replicated by the model.

Consider a dataset with n values: x_1, x_2,…, x_n, where each of them is associated with a predicted value $\hat{y}_i$. We define the residuals as:

$$e_i = x_i - \hat{y}_i \tag{3-17}$$

Besides, we define the mean of the observed data as:

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{3-18}$$

Finally, the variability of the dataset can be measured using the following formulas: *i*) total sum of squares eq. (3-19), *ii*) sum of squares of residuals eq. (3-20).

$$SS_{tot} = \sum_i (x_i - \overline{x})^2 \tag{3-19}$$

$$SS_{res} = \sum_i (x_i - \hat{y}_i)^2 = \sum_i e_i^2 \tag{3-20}$$

Then, the most general definition of R^2 is:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{3-21}$$

Likelihood Function

The *likelihood* is a function of the parameters from a statistical model given some specific observed data. Likelihood functions are useful to estimate a parameter given specific observed data. In particular, the likelihood acts as an indicator of how much the data contributes to the probability of the parameter value or of the model.

For the sake of clarity, consider an example where a supposedly *fair* coin is tossed 6 times, obtaining the following results: HHHTTH, where H and T stand for heads and tails respectively. Fig. 3-3 shows the likelihood of the coin *fairness*; it can be observed how the maximum value of the likelihood is obtained around 0.65, far from the expected result around 0.5. Hence, we could presume that the coin has been slightly tricked.

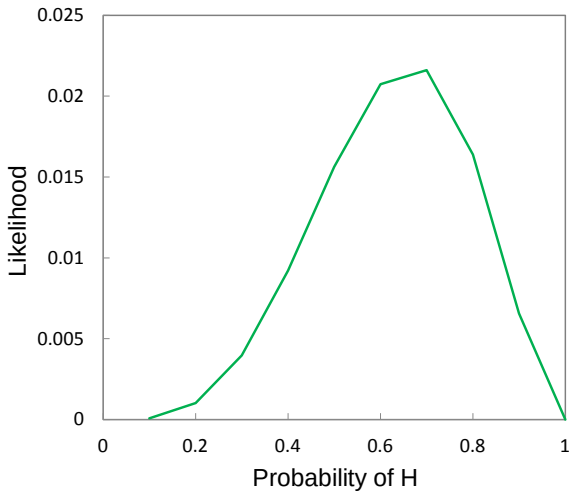


Fig. 3-3. Likelihood function example.

In some applications is more convenient the logarithm of the likelihood (*Log-likelihood*) as it is a strictly increasing function. Note that the logarithm of a function achieves its maximum value at the same points as the function itself, and hence the log-likelihood can be used in place of the likelihood in maximum likelihood estimation and related techniques. In addition, finding the maximum of a function involves taking derivatives and it is often easier to work with the log-likelihood rather than the original likelihood function.

In addition, the *marginal likelihood* is a particular likelihood function where the likelihood is based on only part of the data, some variables have been marginalized (integrated out).

Akaike Information Criterion (AIC)

Eq. (3-22) presents one of the forms to compute the AIC, an indicator that provides an estimation of the quality of a statistical model compared to others for a given data. The quantity *maxL* is the maximum of the likelihood function and *M* is the number of parameters in the model [Bi06].

$$AIC = 2M - 2\ln(maxL) \tag{3-22}$$

Hence, AIC weights the accuracy of the model and the number of parameters required. We aim at minimizing AIC values by either increasing the maximum likelihood of the model or decreasing the amount of parameters. In other words, considering two models providing the same accuracy, the one with less parameters (parsimonious model) will be preferred.

3.2.5 Bayesian Statistics

So far, the *frequentist* inference approach has been considered thus, conclusions from sample data were based on the frequency or proportion of the data. In contrast, Bayesian inference uses the Bayes' theorem to update the probability for a hypothesis as more evidence or information becomes available.

Bayesian inference derives the posterior probability P($A|B$) from two antecedents A and B: a prior probability P($B|A$) and a likelihood function derived from a statistical model for the observed data. Bayesian inference computes the posterior probability according to *Bayes' Theorem*:

$$P(A|B) = \frac{P(B|A)\cdot P(A)}{P(B)} \tag{3-23}$$

3.3 Machine Learning and Data Mining

Machine learning is omnipresent nowadays as a technique to find relations between inputs and outputs, where no specific analytical equations (or algorithms) can be found to perform such transformation. In fact, as stated in [bAl10], "*What we lack in knowledge, we make up for in data*".

For illustrative purposes, let us consider a quick example where we aim at developing an algorithm to filter spam from legitimate emails where both: *i)* incoming emails as input and *ii*) decision: *yes*/*no* as output are known. For this particular example there is not a simple transformation from input to output,

however, an exhaustive study of thousands of *examples* of both type of messages could extract *automatically* the algorithm for this purpose [La13].

ML is an incredibly powerful tool for Big Data, predicting future behavior. One of the most well-known examples of ML in the technical industry is related to Amazon's or Netflix's algorithms, which can make suggestions based on your own choices in products or movies.

Algorithms for ML can be classified into three different categories according to the input received data [Bi06]:

- **Supervised learning**, when the training data contains examples (data points) of the input vectors and their corresponding target vectors; in other words, when a *label* is available for a subset of the dataset (Fig. 3-4(a)). The training set is used to generate a function to map inputs with the desired outputs until the model achieves a certain level of accuracy. This kind of algorithms is often used for *classification*; predicting to which category does a new data point belong to.
- **Unsupervised learning** is applied when the training data consists of a set of input vectors without any corresponding target value to predict (Fig. 3-4(b)). As an example, *clustering* targets at finding collection of data points with similar features among them but different from other data points in input data.
- **Reinforcement learning** is based on training the machine to make specific decisions. Initially it is continuously self-trained using trial and error (Fig. 3-4(c)). Then, it learns from experience and captures this knowledge to make accurate decisions, such as Markov decision process.

In this PhD thesis, only the two first techniques are considered. It is worth noting that the supervision refers to the fact that the target values provide a way for the algorithm to check how well is the learning achievement.

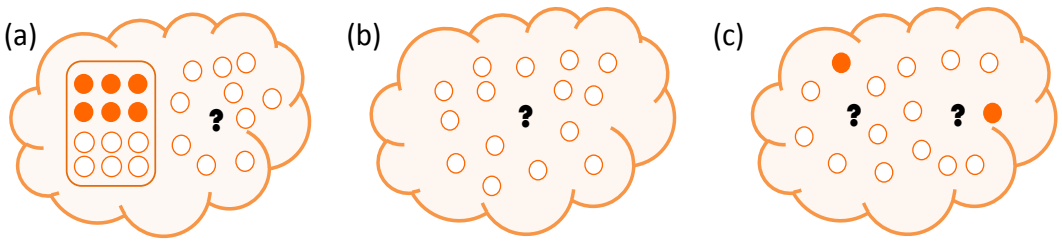


Fig. 3-4. (a) Supervised, (b) unsupervised and (c) reinforcement learning algorithms.

Another way to classify Machine Learning algorithms would be according to their function similarity:

- **Regression algorithms** deal with modeling the relationship between variables. Some algorithms are simple and multiple linear regression, as well as polynomial regression.
- **Classification algorithms** aim at deciding at which already known group does a new data point belong to. Some algorithms are Support Vector Machines (SVM), Naive Bayes, and decision tree classification.
- **Clustering algorithms** are focused on modeling as centroid-based. *K-Means* is the well-known clustering algorithm.

In the process of creating ML algorithms, datasets are divided into training and testing datasets. An approximation proportion could be the use of 60%-80% for training and the 40%-20% for testing. First, the model is obtained with the training dataset and then checked with the testing dataset.

Let us present some of the ML algorithms we are going to use through this PhD thesis.

3.3.1 Support Vector Machine

The SVM is a supervised learning technique used for classification and prediction. SVMs are easy to understand when are used for binary classification; a boundary called *hyperplane* separates data into groups of similar features; this *hyperplane* becomes a line in a 2D (Fig. 3-5(a)) and a plane in the case of 3D (Fig. 3-5(b)). When the two groups can be perfectly divided by the hyperplane, they are called *linearly separable* [La13].

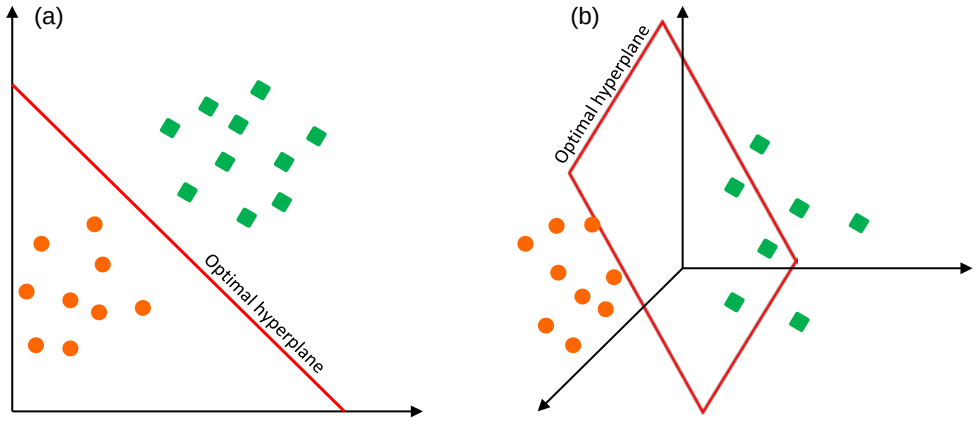


Fig. 3-5. Two-class problem where the classes are shown by dots and squares. (a) computes the optimal line in 2D, and (b) depicts the optimal hyperplane between both classes in 3D.

Given a set of training data, an SVM can be trained to build a predictive model to classify new data points. Each data point in the dataset belongs to the *n*-dimensional space of the problem; each coordinate of the n-dimensional vector representing the data point represents a particular feature.

The SVM algorithm is able to find several hyperplanes that separate the points into two different classes. The choice of the best hyperplane is based on maximizing the separation between the two considered classes (Fig. 3-6(a)). This problem will be tackled in the next subsection.

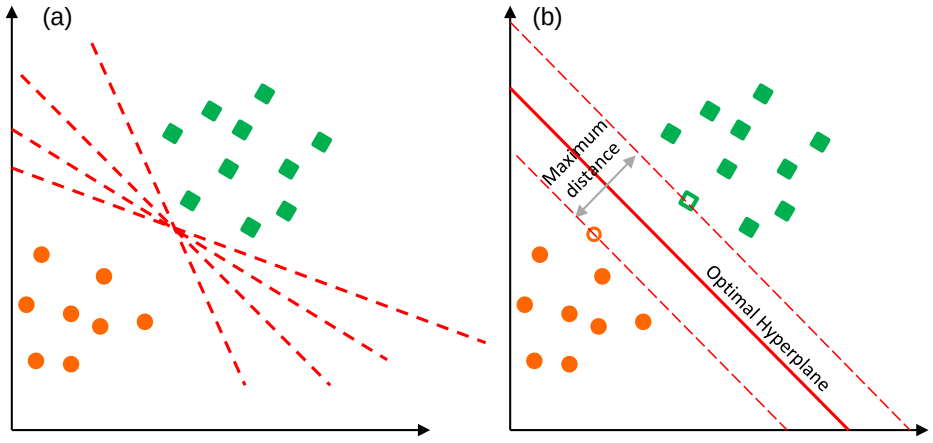


Fig. 3-6. Two-class problem where the classes are shown by dots and squares. (a) Different possible hyperplanes, and (b) the optimal hyperplane.

Linearly Separable Classes

Let us consider a binary classifier, where we are given a training dataset with a number m of input data points in the n-dimensional space $\boldsymbol{x}^i = (x^i_1, x^i_2, \ldots, x^i_n)$ and labels $y^i = \pm 1$:

$$\begin{matrix} \left(x^1, y^1\right) \\ \vdots \\ \left(x^m, y^m\right) \end{matrix} \tag{3-24}$$

In order to improve the chance of classification, we consider the search of the Maximum Margin Hyperplane (MMH) [La13], characterized by creating the greatest separation between the two considered classes (Fig. 3-6(b)).

The support vectors (the not filled markers in Fig. 3-6(b) are the data points that are the closest to the hyperplane; each class has at least one support vector, but it is possible to have more than one. It is possible to define the MMH using only the support vectors; the support vectors provide a very compact way to store a classification model, even if the number of features is extremely large.

In SVM, the error is minimized by maximizing the margin γ, i.e., the minimal distance between the hyperplane separating the two classes and the support vectors. The separating hyperplane can be written as:

$$w \cdot x - b = 0 \tag{3-25}$$

Where b is a scalar known as the bias; it is conceptually equivalent to the intercept term to specify lines in 2D space. While, $\boldsymbol{w}$ is the n-dimensional vector of weights and it is normal to the hyperplane.

Two parallel hyperplanes, called *margins,* can be selected to separate the two classes aiming at making the distance between them as large as possible (see Fig. 3-7). Then, the MMH is the hyperplane that lies in between them. These hyperplanes can be described as follows [Bi06]:

$$w \cdot x - b = 1 \tag{3-26}$$

$$w \cdot x - b = -1 \tag{3-27}$$

The geometrical distance between those two hyperplanes is computed as follows. If x^1 and x^2 are support vectors of each of the classes, then subtracting eq. (3-26) and eq. (3-27), we obtain:

$$w \cdot (x^2 - x^1) = 2 \tag{3-28}$$

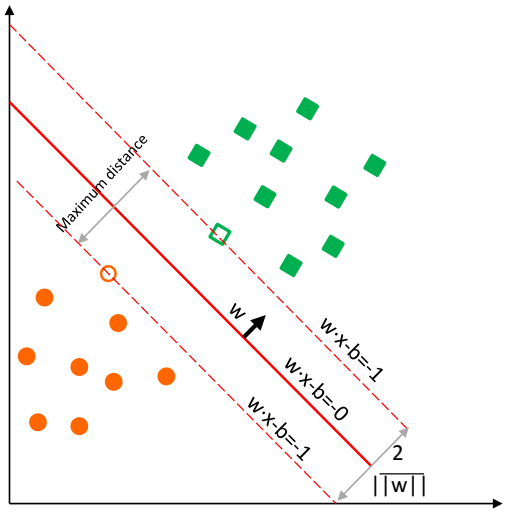


Fig. 3-7. Equations for the margins and the optimal hyperplane in a 2D example.

Using eq. (3-28), we project the n-dimensional vector x_2-x_1 (the blue dotted line in Fig. 3-8) onto the vector normal to the hyperplane (the black dashed line).

$$\left(x^2 - x^1\right) \cdot \frac{w}{\|w\|} = \frac{2}{\|w\|} \tag{3-29}$$

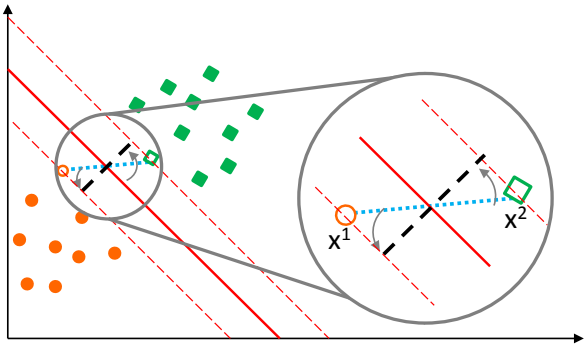


Fig. 3-8. Projection of the distance over the perpendicular margins' line.

We obtain twice the size of the margin γ, therefore the size of the margin is:

$$\gamma = \frac{1}{\|w\|} \tag{3-30}$$

An optimization problem arises to maximize the margin γ. Therefore, maximizing eq. (3-30) is completely equivalent to minimize the inverse of γ.

$$Maximize \quad \gamma = \frac{1}{\|w\|} \Leftrightarrow Minimize \quad \|w\| \Leftrightarrow Minimize \quad \frac{1}{2}\|w\|^2 \tag{3-31}$$

We need now to define the constraints for this optimization problem. These constraints state that each data point must lie on the correct side of the margin. The equations return 1 only for the support vectors, being greater than 1 for every other point.

$$\begin{aligned} &1 \cdot (w \cdot x - b) \geq 1 \quad hyperplane \quad above \\ &(w \cdot x - b) \leq -1 \quad hyperplane \quad below \end{aligned} \tag{3-32}$$

Working with eq. (3-26) we obtain the constraints for the hyperplanes from eq. (3-32), which can be grouped in eq. (3-34) thanks to eq. (3-33).

$$w \cdot x - b \geq 1 \Leftrightarrow w \cdot x - b \leq -1 \Leftrightarrow -1 \cdot (w \cdot x - b) \geq 1 \tag{3-33}$$

$$\left.\begin{aligned} &if \quad y = 1 \Rightarrow 1 \cdot (w \cdot x - b) \geq 1 \\ &if \quad y = -1 \Rightarrow -1 \cdot (w \cdot x - b) \geq 1 \Rightarrow (w \cdot x - b) \leq -1 \end{aligned}\right\} y \cdot (w \cdot x - b) \geq 1 \tag{3-34}$$

Then, the optimization problem reads:

$$\begin{aligned} &\textit{Minimize} \quad \frac{1}{2}\|w\|^2 \\ &\textit{subject} \quad \textit{to}: y_i \cdot (w \cdot x_i - b) \geq 1, \quad \textit{for} \quad i = 1,...,n \end{aligned} \tag{3-35}$$

Solving this problem, variables $\boldsymbol{w}$ and b are determined and as such our classifier, where:

$$x \rightarrow \text{sgn}(w \cdot x - b) \tag{3-36}$$

Non-Linearly Separable Classes

In many cases, however, data is not linearly separable by a hyperplane. In order to extend SVM to such cases, we consider the so-called *soft margins* and we introduce the concept of hinge loss; a loss function used for maximum margin classification.

For these special cases, the optimization problem needs to be revisited and a relaxation needs to be introduced to allow some points to fall on the incorrect side of the margin [La13]. Now, the optimization problem from eq. (3-35) is re-written as in eq. (3-37), where C represents the cost of allowing not perfect classification, i.e., the cost of the relaxation. The importance of the parameter C is illustrated in Appendix A.

$$\begin{aligned} &\textit{Minimize} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\zeta_i \\ &\textit{subject} \quad \textit{to}: y_i \cdot (w \cdot x_i - b) \geq 1 - \zeta_i \quad \forall x_i, \zeta_i \geq 0 \end{aligned} \tag{3-37}$$

3.3.2 Bayesian Network Classifiers

Probabilities play a central role in modern patter recognition. Classifiers based on Bayesian methods use a set of training data to compute the observed probability of each outcome based on the evidence provided by feature values. Once the classifier is applied to unknown data, the algorithm relies on the already observed probabilities to be able to predict the most likely class for the new features. To work with this kind of algorithm, data from numerous features should be simultaneously considered to estimate the overall probability of an outcome, even if some features have weak effects [La13].

Sometimes, it could be beneficial to improve probability analysis using probabilistic graphical models, such as Bayesian Network (BN). Generally speaking, a BN consist on a probabilistic directed acyclic graphical model that represents a set of variables and their conditional dependencies, e.g., relations between diseases and symptoms, cause and effects [Bi06].

3.3.2.1 Naïve Bayes

Recall that the Bayes theorem was introduced in Section 3.1, which is necessary for the Naïve Bayes approach. In particular, Naïve Bayes algorithm is a special case of a BN. The key assumption of the Naïve Bayes (NB) model is that, conditioned on the class to be predicted, the distributions of the input variables $x^1,\ldots,x^m$ are independent. In other words, NB assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. This assumption is rarely satisfied (e.g., considering cancer and age as variables, or the age and the salary of a person as variables, at some point in their lives, there would exist some kind of relation). Anyway, the exact reason why Naive Bayes works well in spite of its inaccurate assumptions has been the subject of much speculation [La13], [Bi06].

NB algorithm has many advantages, like it is an easy algorithm to build, it is fast for class prediction, it is very useful for very large datasets, it requires relatively few examples for training, but it works well with very large number of examples, and it is easy to obtain the estimated probability for a prediction. As well as some weaknesses, like it relies on the assumption of equally important and independent features, and it is not ideal for datasets with many numeric features, because in the case a categorical variable has a category which is not present in the training dataset the model will not be able to make a prediction for this case.

For the sake of clarity, let us consider an example. Note that these values are not experimental measurements, simply a set of points to illustrate the main concept of this algorithm. Consider the use of internet depending on the age and on people's way to go to work, e.g., walking or driving (see Fig. 3-9(a)). We want to use the NB algorithm to classify new data points and know whether the new person goes walking or by car (Fig. 3-9(b)).

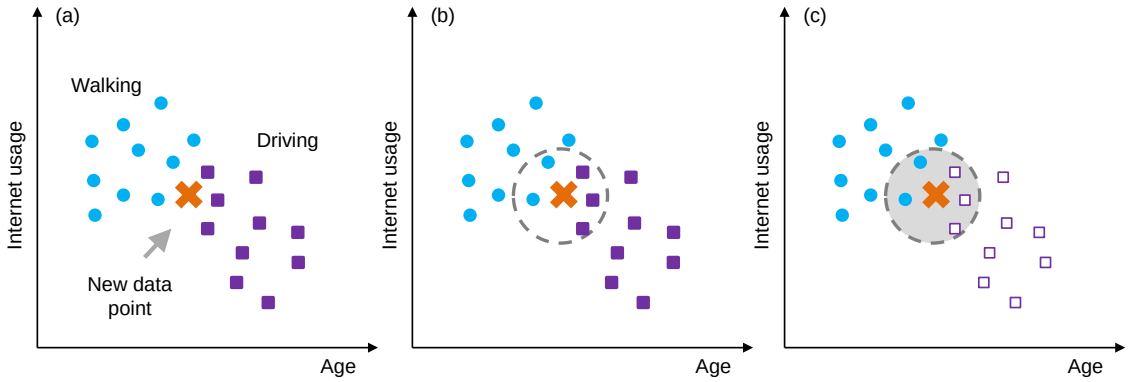


Fig. 3-9.(a) Input data and new point, (b) circle for computing marginal likelihood, and (c) only consider points for class "walking".

Let us compute the probability that a new person added to the dataset walks to work, P(*Walks* | *x*), according to their age and use of the internet. For this purpose, we use the following equations:

Prior probability	$P(Walks)$ = # *walks/total observations* =10/20	(3-38)
Marginal Likelihood	$P(x)$ = # *similar observations* / *total observations* = 5/20	(3-39)
Likelihood (Fig. 3-9(c))	$P(x \mid Walks)$ =# *similar observation among walk* / *total walkers* = 2/10	(3-40)

The result for the probability that the new point belongs to the "walking" group is:

$$P(Walks \mid x) = \frac{P(x \mid Walks) \cdot P(Walks)}{P(x)} = \frac{\frac{2}{10} \cdot \frac{10}{20}}{\frac{5}{20}} = 0.40 \tag{3-41}$$

3.3.3 Clustering

Although several clustering algorithms can be found in ML methods, in this PhD thesis we rely on the popular *K-means* clustering algorithm. *K-means* is designed to find a custom number K of clusters in a dataset. The algorithm assigns each of the m points to one of the K clusters aiming at minimizing the differences within each cluster and maximizing the differences between the clusters [La13].

For illustrative purposes, let us consider an example where only two variables are plotted (Fig. 3-10(a)). By inspection, several ways of grouping could be considered. The advantage of K-means algorithm lies in the fact that it takes out the complexity of this decision and easily finds clusters in the dataset (Fig. 3-10(b)).

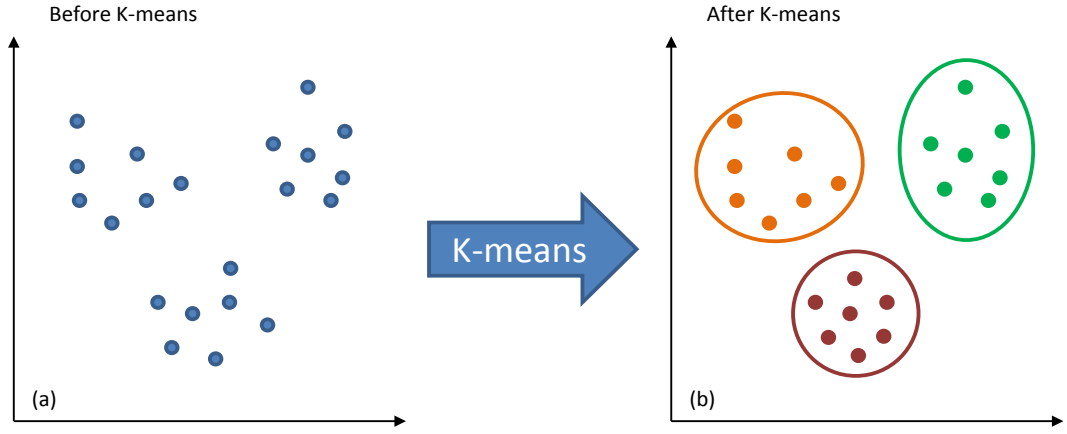


Fig. 3-10.(a) Input data, (b) after applying K-means data is grouped in clusters.

Let us analyze the process presented in Fig. 3-10 in more detail. For this purpose, let us describe the main steps of the K-means algorithm, which are presented in Table 3-1.

Table 3-1 Steps in K-means algorithm.

Step	Action
1	Choose a number K of clusters.
2	Select the centroids by selecting K points at random, not necessarily from the dataset.
3	Assign each data point to the closest (Euclidean) centroid.
4	Compute and place the new centroid of each cluster.
5	Reassign each data point to the new closest centroid. If any reassignment took place, go over steps 4 and 5, otherwise finish.

An optimization problem needs to be solved to minimize the distance between each point of the cluster, x_i, and its centroid, c_j:

$$\min\left(Dist(x_i, c_j)\right) \tag{3-42}$$

This minimization problem is considered in step 3 of Table 3-1 and repeated every step 5 if needed.

3.4 Data visualization

Data visualization is a technique that involves processing raw data aiming at communicating information in a clear way using graphical resources. It is worth noting that data visualization is extremely helpful to discover hidden behavior when points are presented graphically rather than checking data points in a table.

The process involving data visualization is far beyond the simplistic idea of seeing a plot. It is important to note, considering data visualization on a grand scale, that it is a tool that combines decision making, human interaction and data analysis [Ke08] (see Fig. 3-11).

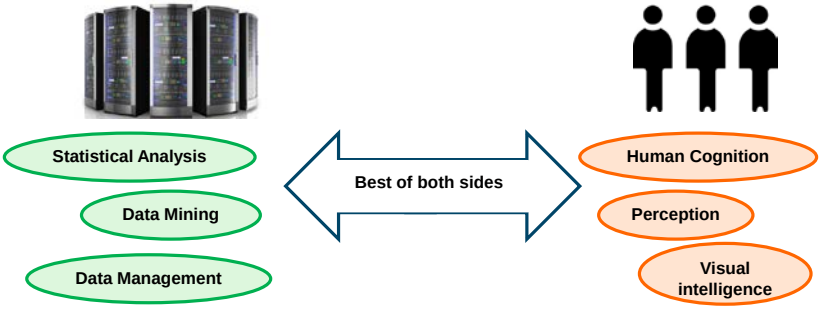


Fig. 3-11. Improving visual analytics by getting the best from machines and human knowledge (based on [Ke08]).

The main goals of data visualization include [Es17], [Sh96]: answering questions, synthesizing information to obtain an insight from massive data, making decisions, finding visual patterns, and seeing data in context.

According to author in [Sh96], there exist seven data types that reflect an abstraction of the reality, such as those in Table 3-2.

Table 3-2 Data types

	Data types	
1-Dimensional	Item:	Line of text containing a string of characters.
	Example:	Linear data types; documents, source code.
2-Dimensional	Item:	Has attributes such as name, owner, value, and features such as size, color, opacity, etc.
	Example:	Geographic maps, floorplans.
3-Dimensional	Item:	Volume and potentially complex relationship with other items.
	Example:	Real-world objects such as molecules, the human body, and buildings.
Temporal	Item:	Time lines, items a start and finish time.
	Example:	Medical records, project management.
Multi-dimensional	Item:	With n attributes become points in an n-dimensional space.
	Example:	Datasets with many dimensions.
Tree	Item:	Hierarchies that are a collection of items; each item has a link to one parent item (except the root).
	Example:	Tree-structured data.
Network	Item:	Items linked to an arbitrary number of other items when relationships among items cannot be conveniently captured with a tree structure.
	Example	Acyclic networks, lattices.

Visual structures are based on graphical properties effectively processed by human vision. In this sense, Fig. 3-12 presents the reference model for visualization [Chi00], which it can be described as the mapping of data to visual form supporting human interaction in a workspace for visual sense making [Ca99]. This model, breaks down the visualization technique into: *i*) three types data transformation (green boxes) and *ii*) four stages of human interaction (orange circles). In the first step, *data transformation*, data needs to be filtered and processed to transform them into a suitable format. Besides, in this scheme, arrows flow from *raw data* on the left towards the human experiencing multiple chained transformations. Moreover, arrows also flow from the human at the right to the transformation

themselves, showing that these transformations are user-operated controlled. Finally, *View Transformations* create Views where the user can control parameters of these transformations.

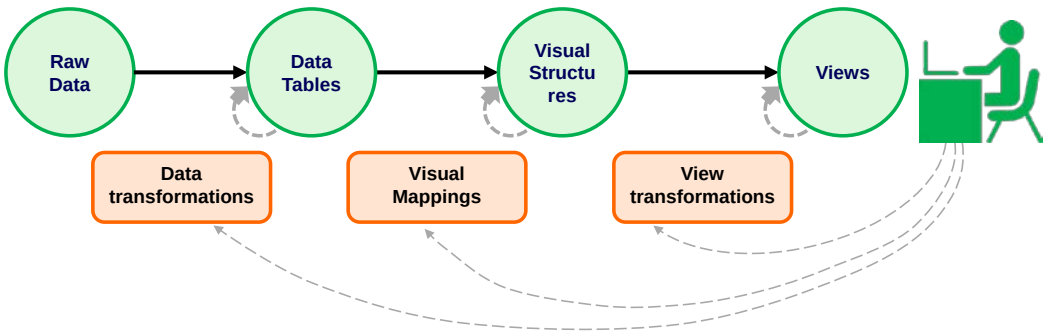


Fig. 3-12. Reference model for visualization [Ca99].

Let us now introduce the basic charts used in this PhD thesis, using an approach that considers the number of variables considered: *i*) univariate, *ii*) bivariate or *iii*) multivariate data [Sh96]:

Univariate data

In order to deal with univariate data, several graphical techniques are suitable such as:

- **Line graphs** plot data points on a Cartesian coordinate grid connected by a line. The direction of the lines on the graph eases the process of visualization, since an upward slope means that values have increased, while downward slope indicates that values have decreased. This kind of plot is used to display quantitative values over a continuous interval (see Fig. 3-13(a)).
- **Histogram**: This kind of chart helps to visualize the distribution of data over a continuous interval. Each bar represents the tabulated frequency at each interval. Thus, it helps to give an estimation to see where values are concentrated (see Fig. 3-13(b)).

Bivariate data

When dealing with bivariate data the best way of representing data would be

- **Scatter plot**: It is a mathematical diagram using Cartesian coordinates. The scatterplot uses a collection of points placed to display values from two variables. By plotting a variable in each of the axis, a relationship or correlation between the two variables can be found. They are very useful to check if one variable impacts the other. It is one of the best well-known and widely used plots (see Fig. 3-13(c)).

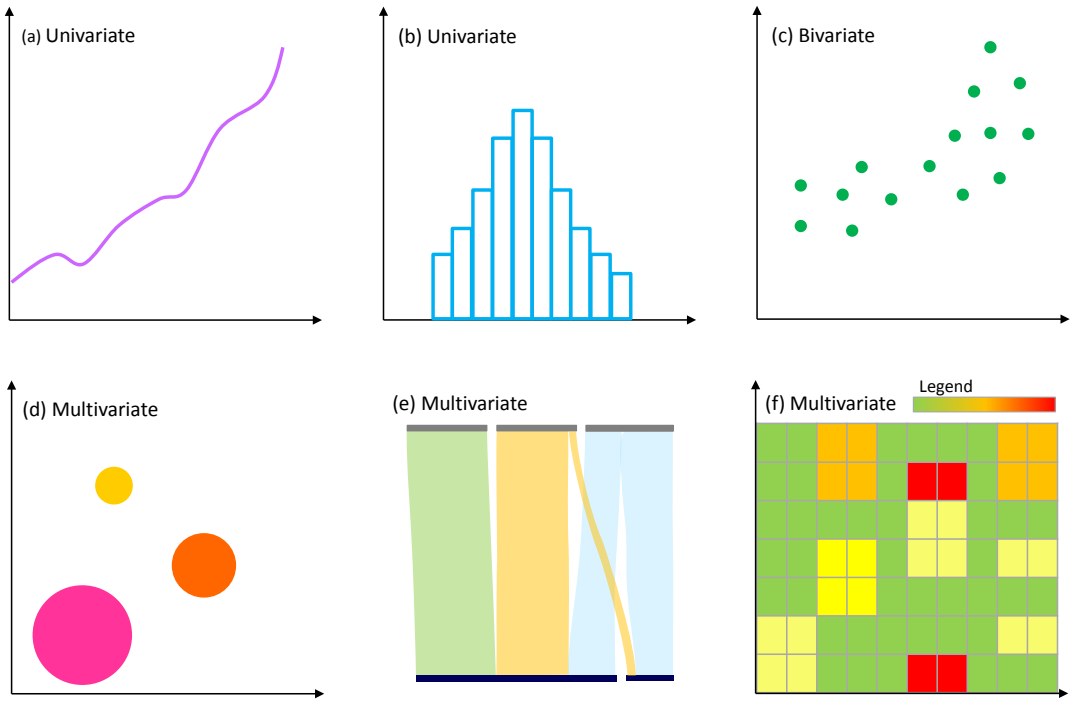


Fig. 3-13. Different schematic charts used in through this PhD thesis.

Multivariate data

Finally, for the case of multivariate data, the best techniques are:

- **Bubble chart**: It is a multi-variable graph that uses a Cartesian coordinate system to plot points along a grid, where *X* and *Y* axis represent different variables. In contrast to the scatter plot, here, each point is assigned a label or category. The main advantage of bubble charts is that several dimensions can be represented in a 2D plot. The graphic allows to plot points in two dimensions (*X* and *Y* axis), the third dimension is represented by the area of the bubbles, and colors are used to represent an additional variable. This chart can be used to seek for patterns/correlations (see Fig. 3-13(d)).
- **Parallel coordinates or Sankey diagram**: This diagram shows flow and their quantities in proportion one to another. The width of the lines reflects its magnitudes, so the bigger the line, the larger the quantity. Color may be used to divide the diagram into different categories. The function works as follows, for every single dimension, a horizontal (or vertical) bar is displayed for each of its categories. The width of the bar symbolizes the size of the absolute frequency or the proportional fraction of the category total (see Fig. 3-13(e)).
- **Heat map**: Heat maps are used to visualize data through variations in the colors; they are useful for showing variance across multiple variables and to reveal patterns. Normally, all the rows belong to a category and all the columns to a different one. The cells in the table are based on the

relationship between the two variables (row, column) and they contain color-coded categorical or numerical data. However, heat maps have some limitations due to the need of converting data into a matrix format. Besides, to successfully understand the graph, the choice of colors to represent the difference between high and low values is subtle, and a legend is required for detailing color meaning (see Fig. 3-13(f)).

3.5 Conclusions

This chapter has focused on giving the needed background of concepts related to optimization, statistics, and machine learning, to ease the comprehension of some concepts used in this PhD thesis. Starting with a short introduction to optimization and basic statistics, the chapter exhaustively presented the ML algorithms that are used through this PhD thesis.

Let us summarize the main strengths and weaknesses of the considered ML algorithms [La13].

Table 3-3. Strengths and weaknesses of different statistical and ML algorithms.

ML Algorithm	Strengths	Weaknesses
Linear Regression (Supervised)	• most common approach for modeling data • can be adapted to model	• makes strong assumptions about data • unable to handle missing data • useful for numeric features (categorical data needs pre-processing)
SVM (Supervised)	• can be used for classification or prediction • high accuracy • not influenced by noisy data	• requires many combinations of kernels and model parameters to find best model • can be slow to train (input dataset has large number of features)
Bayesian Networks (Supervised)	• simple, fast and effective • easy to obtain the estimated probability for a prediction	• based on the often-faulty assumption of equally important and independent features • not ideal for datasets with many numeric features
Clustering (Unsupervised)	• uses simple principles, • is very flexible being able to adapt with simple adjustments	• not guaranteed to find the optimal set of clusters because it uses random elements

A section focused on visualization techniques eventually pointed out the main uses of this technique.

The following chapter is focused on reviewing the State-Of-The-Art of the objectives of this PhD thesis, aiming at clearly identifying niches that may become research opportunities.

Chapter 4

Review of the State-of-the-Art

In this chapter, we review the state-of-the-art related to the different goals defined for this Ph.D. thesis with, the twofold objective of ensuring that these goals have not yet been covered in previous works in the literature and to serve as a starting point for this research work targeting to occupy the discovered niches.

4.1 Packet Traffic

In this section, we focus on the packet layer targeting to detect traffic anomalies, i.e., short-living events that do not follow expected traffic patterns (see a survey in [Ch09]). Since detecting traffic anomalies requires being able to forecast the traffic that is expected, this section also covers traffic modelling. Once those models are obtained, they can be used for several purposes, such as traffic prediction that can feed optimization models for preparing the VNT in a proactive manner [Mo17], as well as for traffic generation that can be used for ML algorithms training.

4.1.1 Traffic Anomalies

Traffic Anomalies can create network congestion and stress resource utilization in packet nodes and hence, its prompt detection becomes essential since it allows preparing the network e.g., by reconfiguring the VNT in multilayer network scenarios [Mo17].

Specifically, traffic anomaly detection can be used to trigger lightpath provisioning and network re-configuration, which entails analyzing monitoring data to anticipate congestion. It is clear that developing efficient techniques to detect traffic anomalies in real time would empower network operators to prevent grave consequences induced by such anomalies affecting end users. However, detecting anomalies is a difficult task because anomalous patterns need to be extracted and interpreted from large amounts of high-dimensional, noisy data.

In order to detect traffic anomalies, it is essential to: *i*) monitor traffic at the nodes and *ii*) to model such traffic [Li16]. For packet networks specifically, the traffic monitoring function allows identifying (classifying) the traffic belonging to a specific service or destination, so as to apply specific policies.

Monitoring traffic samples are produced at the packet nodes; according to the ITU-T [M.2120], performance events are counted second by second over every 15-minute period. At the end of a period, they are collected in a repository for further analysis [Mo17], [Mas14]. Data analysis can be used to create predicted traffic matrices for the near future; please refer to [RFC7536] for a list of use cases of traffic monitoring. Among use cases, that of identifying network failures and problems (or anomalies) is undoubtedly of the interest of many network operators. It is clear that when analytics are applied to data collected every 15 minutes, the expected traffic anomaly detection times will be as well in that order of magnitude. Consequently, the monitoring period should be reduced, which in turn increases the amount of monitoring data to be sent to the centralized data repository.

Many works in the literature can be found on intrusion and Denial-of-Service (DoS) detection (see, e.g., [Bh14]). However, traffic anomalies are a less common topic. Authors in [La04] proposed a general method that entails monitoring traffic in links and correlate monitoring time-series to detect volume anomalies, identify the anomalous Origin-Destination (OD) pair, and estimate the amount of traffic involved in that OD pair. The method is based on applying Principal Component Analysis (PCA) to separate the multidimensional space occupied by a set of network traffic samples into disjoint subspaces corresponding to normal and anomalous traffic conditions. Note that this method requires from previous collection and analysis of anomalous traffic, which is not always feasible since it is a very rare occurrence and that traffic anomaly might not follow a predictable pattern.

Regarding traffic periodicity, authors in [Cu15] analyzed the performance of several methods in detecting link traffic anomalies with respect to the traffic pattern on a 24h typical day. As traffic varies throughout the day, it is essential to consider the concrete traffic period in which the anomaly occurs. Authors in [Ma12] suggested a model based on splitting a 24 hours day period into 16 non-overlapping intervals of 90 min. Their on-line change detection algorithm identified relevant changes in link utilization and reported those links to a centralized controller for further analysis. Finally, authors in [So07] compared three traffic aggregation entities such as ingress routers, input links, and OD pairs and concluded that traffic aggregation level has a significant impact on the number of anomalies detected and on the false positive rate; they showed that aggregating traffic by OD pairs is the most appropriate choice.

Finally, different detection methods have been studied in the literature and applied to different contexts such as in traffic changes detection [Mo17] and anomaly

detection based on hypothesis testing [Ku00]. These methods are a good starting point for this thesis.

Besides, and according to our knowledge, no research has been performed towards the detection of multiple traffic anomalies. Accordingly, it is still an unexplored field to be studied.

In conclusion, considering the previous works, it is clear that a further study specially focused on traffic anomaly detection is suitable. Different OD traffic anomaly detection methods should be proposed combined with different monitoring strategies to efficiently detect OD traffic anomalies to minimize detection time and traffic losses. This is the work that we target in the specific objective G.1.1 and G.1.2.

4.1.2 Traffic Modeling

Traffic generation is a useful technique that enables studying and evaluating the network performance through simulation, when real traffic traces are not available. In fact, as pointed out by authors in [Ka02], there is an absence of public availability of real world network traces, specifically for traffic in the operators' and Internet service providers core transport networks.

Therefore, due this unavailability, one option is to attempt generating simulated traces which resemble to real ones. Authors in [Va14] state that it is quite easy to generate traffic, but it is far more difficult to produce traffic that exhibit real characteristics, such as the ones observed through the Internet. Several traffic generators have been developed until present but, to the best of our knowledge, literature is mainly focused on generating representative IP traffic for packet-based networks or connection arrival based on the Poisson distribution for circuit-switched networks [Ca12].

Model estimation requires *i*) new traffic samples arriving from the domain controller (e.g., every 15 min); *ii*) a time windows within which traffic samples will be considered for model estimation, and *iii*) the minimum number of samples to estimate the model. Once the model has been estimated, it can be evaluated to obtain a bitrate prediction. Given an absolute time t for which the bitrate needs to be estimated and a predictive traffic model, t is translated to a time within the model's period. Finally, a tuple containing the average bitrate and standard deviation prediction for time t is obtained.

For further details regarding traffic generation, please refer to Appendix B.

4.2 Failures at the Optical Layer

Faults or degradations occurring in the network may affect the fulfillment of SLAs, for that very reason, it is essential to monitor the physical layer to localize the failed elements and address the issue.

Besides, although commercially available optical equipment is able to correct degraded optical signals by means of Forward Error Correction (FEC) algorithms, a value of pre-FEC BER over the pre-defined limit (max BER) would imply a non-error-free post-FEC transmission and, as a result, communication would be disrupted. Therefore, a prompt detection of optical connections with excessive pre-FEC BER can greatly reduce SLA violations.

4.2.1 Failure Detection

Hard Failures Detection

For the purpose of detecting hard failures, an OSC can be implemented between every pair of optical nodes carrying information regarding the optical layer. In particular, when the signal from the adjacent node is lost, a failure in the link is detected. Note that to implement the OSC in a network, the number of additional transmitters and receivers that need to be installed is equal to the number of links. To reduce the number of additional transmitters and receivers, authors in [Ta15] proposed a solution based on monitoring trails (m-trails), which enables all-optical failure localization in the optical domain. M-trails can be implemented as a pair of lightpaths along a common physical route in opposite directions for sensing/monitoring the health of the links along the route.

A different option is proposed by the authors in [We05], consisting in sending optical probe signals sequentially along a set of designed lightpaths, where the network state can be inferred from the result of this set of end-to-end measurements.

Soft Failures Detection

As to soft failures detection literature is less extensive but still, some appealing work can be found. For example, the authors in [Ma00] proposed an algorithm to localize failures in the optical layer considering BER monitoring data. They included data from the following optical components in their algorithm: *i*) optical transmitters and receivers, *ii*) regenerators, *iii*) filters, and *iv*) amplifiers. All the received data is processed, and the algorithm is able to infer the possible cause of the failure. Note that this technique works when failures are persistent in the network, i.e., the failures remain over time.

However, intermittent failures, e.g., those that show a cyclic behavior appearing and disappearing cannot be localized by the above algorithm. In fact, to the best of

our knowledge, no works in the literature have focused on that type of failures. Therefore, our work will be focused on addressing the latter by implementing techniques for the time-series based data analytics algorithms.

4.2.2 Failure Localization

Failure localization is a very useful technique since it helps to greatly reduce failure repair times. In particular, regarding hard failure localization, considerable number of works has been proposed, however, as it happened with failure detection, soft failure localization has been less addressed in the literature.

Hard Failures Localization

Different works can be found in the literature for *hard* failure localization in optical networks, proposing methods for localizing hard link failures that affect a number of established connections, focused on reducing restoration times (see, e.g., [Ta12], [Ze06]). All the proposed methods basically consist on computing and establishing a number of auxiliary connections (m-trails or m-cycles). In the event of a link failure, one single connection would be affected thus, localizing the failed link.

For instance, authors in [Wu09] proposed an m-trail mechanism for fast link failure localization as a result of a fiber cut. Based on defining different m-cycles and analyzing a set of alarm signals generated in each monitor of the cycle, failure localization is achieved. Authors in [Ma05] presented a failure location algorithm to locate single and multiple failures in transparent optical networks by analyzing the received alarms. Finally, authors in [XLi16] proposed an m-trail allocation heuristic providing a simpler installation and smaller management cost. A different failure localization procedure was proposed by the authors in [Ta14] to help the restoration process to localize the failure, thus avoiding the failed resource. Related to m-trails, network kriging was proposed by the authors in [Ch16] to localize failures using alarm correlation.

A different approach for packet networks is presented by authors in [Ko16] based on defining control flows for link failure localization on SDN-based network.

When a hard failure occurs at the optical layer, the affected traffic needs to be immediately restored using currently available resources. For the restauration different algorithms can be devised, including multipath and bitrate squeezing [Pa14].

Nonetheless, some hard failures start as soft failures, and they can be detected as incipient degradations. Therefore, it would be desirable to anticipate and resolve hard failures in order to plan proper actions like traffic re-routing. Even though some soft failures evolution might take a long time, they can affect the quality of stablished optical connections.

Soft Failures Localization

Initially not very critical, they can become as harmful as hard failures, therefore, developing efficient techniques for soft failure localization is key to guarantee SLAs fulfillment and the correct network functioning.

Although, linear impairments (e.g., dispersion) can be compensated by the DSP itself, signal degradations in coherent systems are mainly dominated by amplified spontaneous emission, non-linear effects including also interference, and filters introducing signal distortion. Thus, the identification of such most relevant impairments is mandatory. A challenge is the analysis of monitoring data with the objective of identifying the nature of a problem (e.g., decide if a reduction of the OSNR is due to an amplifier malfunction or some other issue) and such topic still requires to be investigated to reach an adequate maturity. Regarding filtering effects, studies in [Sa12], [Gh13] evaluated the related induced penalties. However, work is still needed to correlate information related to end-to-end parameters such as OSNR and non-linear or filtering effects to identify the type of failure. Note that any of the techniques for hard-failure localization work for *soft* failures affecting individual lightpaths, such as Laser Drift, Filter Shift, or Filter Tightening, and thus in-line monitoring techniques to analyze and evaluate the quality of individual optical lightpaths are required. In this regard, although OSAs could be used to analyze the spectrum of optical signals, until recently, the use of OSA in the network was very limited due to the high cost of accurate OSAs. However, improvements in OSA technology are taking place, and a new generation of cost-effective OSAs with sub-GHz resolution is now available to be integrated into a new generation of optical nodes [Finisar]. Furthermore, OSA and other monitoring techniques require sophisticated algorithms able to identify and localize failures.

While being so critical, few works in the literature have been focused on soft failure localization that might affect a single or a reduced set of optical connections.

Some literature works also take advantage of the previously cited m-trails for soft failure localization. For instance, the authors in [Og14] propose a method to localize multiple link failures, including hard and soft failures.

However, vendors have lately commercialized products to track lightpaths along their route (e.g., [TAnalytics] from Nokia) or to predict network health [BluePlanet].

One step further, would involve the process of soft failure identification aiming at understanding the cause of a failure to accelerate maintenance thus reducing Mean-Time-To-Repair (MTTR). According to our knowledge, until now and since the beginning of this PhD thesis, no research has been performed in this decisive and unexplored field. Recently, this subject has gained more interest proven by the availability of some publications, like [To18]; where the authors agree on the challenges of this process, and the necessity of further research. For this very reason, this PhD thesis has been focused on this process since the very beginning.

In conclusion, considering the previous works, it is clear that a further study considering the whole process of soft failure detection, localization and identification is required. For this purpose, we address these challenges through goals G.2.1 and G.2.2.

4.3 Visualization

Data visualization is the process and ability of abstracting data in a significant way aiming at better understanding it. It is key for transforming huge amounts of raw data into meaningful information by the use of different type charts. The advantage of data visualization resides on the fact that exploits both, raw data analysis and human perception and expertise. Besides, visualization allows the detection of properties and patterns and it even reveals information about the way it was collected. Therefore, using an appropriate visualization, errors and spurious data would be straightforward identified. For this very reason, data visualization is a very valuable technique in quality control [Wa04].

Literature explores the development of more effective visual ways to present data, but it is mainly oriented towards detecting attacks, like [Co07] focused on visualizing vulnerabilities and intrusion detection.

According to the extensive survey presented by authors in [Gu16], research regarding security management is the most trending topic, around 71% of the total papers. However, only few are devoted to monitoring and measurements, where some of them are focused on failures at IP level (see [Ta13], [CAIDA]).

To conclude, taking into account the previous related work, it seems that data visualization at the optical layer has not yet been addressed and should be considered for further study. In particular, in this PhD thesis we consider the specific goal (G.2.3) for applying data visualization techniques to help network operators to detect and localize incipient optical degradation that could evolve on to a hard failure.

4.4 Network Reconfiguration

Network reconfiguration can be triggered periodically or by an unexpected event such as a failure at the optical layer or traffic anomalies at the packet layer. The most common network reconfiguration is in the case of a link failure, where an algorithm to recover the affected connections can be executed [Ve14.1].

Authors in [Ca13] presented a path-triggered spectrum defragmentation in flexgrid optical networks, particularly; in the case a requested connection could not be served using the currently available resources.

The result of the algorithm is a set of already established lightpaths that can be reallocated to make enough room to the incoming connection request. In this example, network reconfiguration is purely reactive against current network conditions.

To the best of our knowledge, network reconfiguration triggered by events such as traffic anomalies affecting the packet layer or degradation at the optical layer has been less explored. Although similar methods, based on optimization, can be applied, the triggering event is as a result of expected future scenarios, which might entail considerations that must be carefully taken into account. Note that this kind of reconfiguration is not reactive, but proactive aiming at minimizing the impact of future network conditions.

As an illustrative example, consider the detection of a traffic anomaly involving a single OD pair $o \rightarrow d$. In such case, a VNT can be immediately triggered to increase the capacity allocated for traffic between origin node $o \in R$ and destination node $d \in R$ (e.g., by setting up new connections on the underlying optical network), where R represents the set of nodes in the packet network. In this regard, authors in [Sr06] presented a VNT reconfiguration algorithm in the case of occurring traffic change.

However, not only single but also multiple anomalies can arise and affect the network; they can be caused by special circumstances, e.g., when a disaster affects the network [Na16]. Therefore, the fact of reconfiguring the VNT after an individual OD traffic anomaly is detected should be deeply studied thus it can result in both, an intolerable number of VNT reconfiguration and traffic losses and in a far from optimal network configuration in terms of resource utilization.

In addition, a prompt detection of optical connections with excessive pre-FEC BER can help to greatly reduce SLA violations, in particular when supporting vlinks in multilayer MPLS-over-optical VNTs. Therefore, focusing on localizing failures at the optical layer and identifying the most probable cause of failure after its detection is key to trigger a network reconfiguration algorithm. Developing an algorithm to be triggered for pro-actively re-routing those demands affected by QoS degradation is needed aiming at reducing the number of affected bandwidth and demands.

As presented above, efficient network reconfiguration considering future scenarios is an area not yet covered in the literature. Consequently, it represents one of the goals (G.3) of this PhD thesis.

4.5 Control and Management Architecture Supporting Autonomous Networking

Significant research and standardization effort has been made in defining control plane architectures and protocols to automate connection provisioning. Starting

from a distributed paradigm, the control and management plane has lately moved towards a centralized one led by the development of the SDN concept.

In a scenario where connection provisioning can be automated, network resources can be made available by reconfiguring and/or re-optimizing the network on-demand and in real-time. This was called as *in-operation network planning* [Ve14.2].

A new generation of internet services requiring stringent requirements, such as video on demand and live TV streaming, are changing the Internet traffic [Ru16]. The bandwidth that these services are demanding is continuously growing [Cisco16], and their allocation in metro and core networks is introducing an unprecedented dynamicity in the traffic involving changes over time not only in its volume, but also in its direction. On statically, and even on reactively -managed networks, such dynamicity entails large overprovisioning, and hence large costs for network operators. With this in mind, operators are seeking for architectures that allow adaptive resource allocation, while fulfilling services' requirements, aiming at minimizing the Total Cost of Ownership (TCO).

In this context, cognitive network architectures have been proven to adapt the network in a cost-effective manner and are gaining increasing importance [TAnalytics]. Specifically, by applying data analytics to monitored data, the *Observe-Analyze-Act* (OAA) loop [Bo76] can be enabled in the network, as proposed in [Mo17], where authors presented an architecture to allow collecting and storing data from monitoring at the network nodes and that was used to train predictive models for every OD pair. They also proposed an algorithm named as VENTURE to reconfigure the VNT based on predicted traffic matrices. In such centralized data analytics architecture, the monitored data is stored and processed in a central controller. Although that architecture provides adaptability to dynamic traffic and a lower TCO, it might not be suitable for the detection of traffic anomalies or degradations, as a result of both, the large amount of data to be conveyed to and analyzed in the centralized data analytics system, as well as the stringent times in which the detection needs to be performed.

Considering this reasoning, centralized data analytics architectures become rather limited, and therefore they must be upgraded to bring data analytics to the nodes and allow distributed data analytics schemes.

We propose to take advantage of novel network reconfiguration capabilities and new network management architectures to perform in-operation planning to reconfigure the network triggered by the results of analyzing monitoring data collected from the devices in the data plane.

In the literature, different architectures supporting cognition are considered for diverse purposes. However, the OAA loop can be applied to a variety of use cases, so specific requirements are needed to implement the OAA loop efficiently. Those requirements include, among others,

- monitoring as many devices as possible in the data plane,
- collecting large amount of data in a centralized repository,
- analyzing heterogeneous data to find patterns and discover knowledge,
- supporting distributed decision making to enable making decisions as closer as possible from the data,
- applying re-optimization techniques to proactively reconfigure the network based on the results of the data analysis to find optimal solutions for the predicted scenarios, and
- the ability to dynamically reconfigure the network data plane.

As far as we know, no previous works have been focused on studying different control and management, architectures to support the OAA loop. Particularly, regarding traffic anomalies affecting the packet layer, it is clear in view of the above requirements that architectures need to be studied to evaluate different data analytics algorithms placements.

Note that the storage required for traffic traces has greatly expanded due to increasing network speeds [Uc16]. Regarding data analytics placement, a meaningful study has been proposed in the context of global iceberg detection, e.g., distributed DoS attack; distributed monitors first measure local traffic for iceberg candidates and then, they report the measured datasets to a central server, which finds the most frequent ones [Hu11].

Research on distributed monitoring architectures is recently receiving great interest. As an example, authors in [Sa16] proposed a hierarchical monitoring architecture aiming at providing monitoring information gathering coming from different layers and network elements in a scalable way without overloading centralized controllers. Reactive strategies based on alarms pre-configured at different monitoring planes for several transmission parameters are proposed as the way to trigger network reconfiguration. This data analysis approach, however, results insufficient for detecting complex events such as traffic prediction. In fact, performing data analytics at the nodes requires from data stream mining algorithms [Me10]. For instance, authors in [Zh17] presented a data stream algorithm working in tight memory to detect hosts connecting to a large number of destinations (named to as *superpoints*). This algorithm provides a clear notion that data modeling is not only needed at a central controller but also distributed at the nodes. Effectively, the algorithm transforms IP packets into superpoints modeling data before sending it to the controller. In consequence, distributed data analytics must be complemented with a monitoring architecture enabling modeling and exporting data rather than just monitoring raw data.

In view of the above, we conclude that architectures to support the OAA loop are still a technological niche that yet needs to be covered. To this end, goal G.4 intends to effectively tackle this issue.

4.6 Conclusions

In this chapter, we have reviewed the state-of-the-art of relevant works related to the goals of this thesis. Table 4-1 summarizes the study.

Table 4-1: State-of-the-art summary

Goals	References
Traffic Anomalies at the packet layer	[Ch09], [Mo17], [Li16], [M.2120], [Ma14], [RFC7536], [Bh14], [La04], [Cu15], [Ma12], [So07], [Ku00]
Failure detection and localization / identification at the optical layer	[Ta14], [We05], [Ma00], [Ta12], [Ze06], [Wu09], [Ma05], [Sa12], [Gh13], [Pa14], [Ch16], [Ko16], [Og14], [XLi16], [Ta15], [Finisar], [ComCom17], [TAnalytics], [BluePlanet], [To18]
Network Reconfiguration	[Ve14.1], [Ca13], [Sr06], [Na16]
Cognitive Architecture	[Ve14.2], [Ru16], [Cisco16], [TAnalytics], [Bo76], [Mo17], [Mo17], [Si13], [Uc16], [Hu11], [Sa16], [Me10], [Zh17]
Visualization	[Wa04], [Co07], [Gu16], [Ta13]

In view of this study, we can conclude that, although some previous works have proposed algorithms and different methods for traffic anomaly detection at the packet layer and for failures detection and identification/localization at the optical layer, many enhancements can still be made, where data analytics combined with optimization can improve the network performance.

In addition, the state-of-the-art review of the architectures already proposed in the literature revealed that they have not been proposed with the aim of bringing cognition to the network or they do not provide the needed scalability.

In this and the previous chapters we have reviewed the state-of-the-art and the background concepts needed to fully understand this work. The following chapters present the essence and contributions of this PhD thesis.

Chapter 5

Traffic Anomaly Detection and VNT Reconfiguration

In this chapter, we focus on the L2 traffic and investigate methods to detect single traffic anomalies. As previously introduced, traffic anomalies can create network congestion, so its prompt and accurate detection would allow network operators to make decisions to guarantee the required network performance avoiding services to experience any perturbation. In particular, we focus on Origin-Destination (OD) traffic; to efficiently detect those traffic anomalies, we study two different detection methods based on data analytics and combine them with three monitoring strategies. In view of the short monitoring period needed to reduce anomaly detection, which entails large amount of monitoring data to be collected and analyzed in a centralized repository, we propose bringing data analytics to the network nodes, while keeping traffic estimation centralized. Once an OD traffic anomaly is detected, a network reconfiguration can be proactively triggered to adapt the network to the new traffic conditions. Exhaustive simulation results on a realistic network scenario show that the monitoring period should be as low as possible (e.g., 1 min) to keep anomaly detection times low, which clearly motivates placing traffic anomaly detection function in the network nodes. Finally, the benefits obtained from reconfiguring the Virtual Network Topology (VNT) after the anomaly has been detected are shown.

5.1 Motivation and Objectives

Two different methods for anomaly detection are studied in this chapter: *traffic-based* and *score-based*; both methods have already been proposed in the literature for traffic changes detection and other scenarios (see Chapter 4). In this chapter, we adapt them for OD traffic anomaly detection; assuming that monitoring data is stored in a centralized repository, we propose to dynamically configure the

monitoring period aiming at reducing the amount of data to be conveyed. Once a traffic anomaly involving a single OD pair $o \rightarrow d$ has been detected, a VNT can be immediately triggered to increase the capacity allocated for traffic between origin node $o \in V$ and destination node $d \in V$ (e.g., by setting up new connections on the underlying optical network), where V represents the set of nodes in the packet network.

Another option is to devise novel architectures where data analytics algorithms can be placed to reduce traffic anomaly detection time and the amount of monitoring data to be conveyed to the central repository. In this chapter, we study different monitoring strategies. Specifically, the contributions of this chapter are the following:

- Two different OD traffic anomaly detection methods together with three monitoring strategies are studied in section 5.2 to efficiently detect traffic anomalies. The traffic-based method uses predictive models to detect sequences of consecutive atypical traffic values, whereas the score-based method is a probabilistic classifier that considers both normal and atypical traffic data to measure how likely is to classify an OD as anomalous.
- Section 5.3 presents a VNT reconfiguration optimization problem to be solved when an OD traffic anomaly has been detected in order to adapt the VNT to the increase of traffic.
- Two architectural approaches are studied in section 5.4, where the anomaly detection algorithm is placed in a centralized controller or distributed inside packet nodes. The monitoring parameters to be configured in the nodes and the data in the notifications that they have to send towards the controller are specified.

The discussion is supported by the results from exhaustive simulation over a realistic scenario in section 5.5.

5.2 OD Traffic Anomalies

5.2.1 Overview

Before detecting OD traffic anomalies, traffic behavior needs to be firstly characterized. To this aim, OD traffic models need to be fitted, so as to generate predictions against which monitored values can be compared. As previously presented in Chapter 2, OD traffic models can be computed for the expected average by predicting two response variables: the *mean* (μ_{od}) and the *standard deviation* (σ_{od}). For the sake of simplicity, hereafter we use just μ and σ in the understanding that those refer to the mean and standard deviation, respectively for a specific OD pair.

Fig. 5-1 illustrates the main steps of the process on a sample network where V= {R_1, R_2, R_3, R_4, R_5}. Monitoring traffic values for every OD pair are collected from the packet nodes periodically, with a given *monitoring period*, denoted as δ. OD traffic models are fitted with these data (Fig. 5-1(a)). Upper and lower bounds, computed as $\mu \pm 3\sigma$, and traffic samples for a given *od* pair and for a typical day are shown in Fig. 5-1(b). Received monitored data can be now compared against its *od* traffic model and those out-of-bound values are considered as *atypical* (Fig. 5-1(c)). Notwithstanding, its detection does not entail a traffic anomaly evidence. In fact, the decision of whether an atypical sample is considered as a traffic anomaly cannot be based on just one single sample, but in observing some previous samples and computing a sort of likelihood of that *od* to be anomalous; we call this *score*. Score values are compared against a defined threshold value (ε_A) and those scores exceeding such threshold are considered anomalies. An example is depicted in Fig. 5-1(d), where an anomaly is detected after receiving two atypical values and considering some other previous within-bound samples.

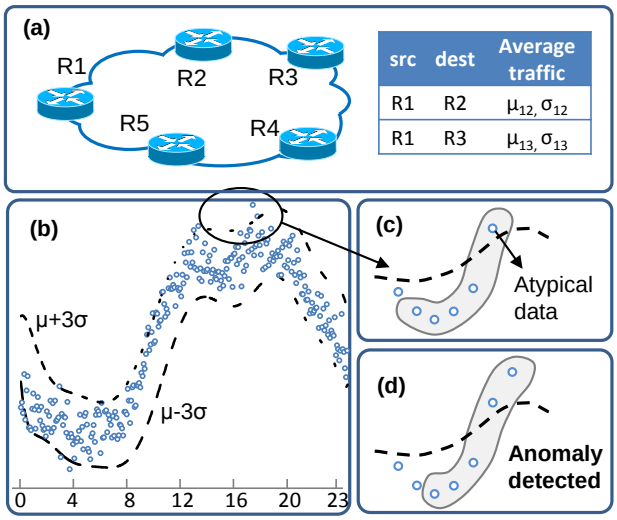


Fig. 5-1. (a) Monitoring samples and estimation boundaries. (b) Monitoring samples vs. estimation. (c) Atypical values over estimation. (d) Anomaly detection.

5.2.2 Notation

The following Table 5-1 presents the key notation used in this chapter.

Table 5-1. Relevant notation used in this chapter.

δ	monitoring period
μ	mean
σ	standard deviation
$s(t)$	score

ε_A	threshold value
$y(t)$	traffic monitoring data of a given OD pair
$\hat{y}(t)$	normalized value of $y(t)$ with respect to the average model
m	number of traffic data samples received
$\hat{Y}$	vector with normalized traffic data samples
$\mathcal{N}(\cdot)$	Gaussian distribution
c	coarse monitoring period in the reactive monitoring strategy
f	fine monitoring period in the reactive monitoring strategy

5.2.3 Detection Method

As stated above, two different methods for anomaly detection are adapted from the literature for OD traffic anomaly detection: *traffic-based* and *score-based*. The adapted *traffic-based* method consists in detecting anomalies after receiving a number of consecutive atypical monitoring data with respect to the $\mu \pm 3\sigma$ confidence interval.

Table 5-2. Score-based algorithm for traffic anomaly detection.

```
INPUT: y(t), μ(t) , σ(t), t, Ŷ(t)
OUTPUT: Anomaly
 1:  ŷ(t) ← Normalize (y(t), μ(t), σ(t)) (eq. (5-1))
 2:  remove oldest value from Ŷ
 3:  Ŷ(t) ← add (Ŷ, ŷ(t))
 4:  if ŷ(t) < 3 then
 5:    return false (atypical)
 6:  s(t) ← computeScore (Ŷ(t)) (eq. (5-3))
 7:  if s(t) < εA then
 8:    return false
 9:  return true
```

The *score-based* method is a probabilistic classifier with two labels for the response: *normal* and *anomaly*. The algorithm (see Table 5-2) is based on a multi-response model to predict whether a sequence of consecutive traffic data belongs to the normal class or, on the contrary, there is sufficient evidence to declare it anomalous. Since this method considers previous (not only atypical) traffic monitoring data, it has the capability of potentially anticipating traffic anomalies thus, reducing detection time compared to that of the traffic–based method.

The algorithm starts when a traffic monitoring data of a given OD pair $y(t)$ is received at time t; let $\hat{y}(t)$ be the normalized value of $y(t)$ with respect to the average model, i.e.,:

$$\hat{y}(t) = \frac{y(t) - \mu(t)}{\sigma(t)}, \tag{5-1}$$

where $\mu(t)$ and $o(t)$ are the mean and standard deviation, respectively, of the traffic model for such OD pair at time t.

Note that a normalized value equal to *K-means* that $y(t) = \mu(t)+k\cdot o(t)$. After normalization, $\hat{y}(t)$ is stored in a fixed-size data series $\hat{Y}$ containing the last m normalized traffic data received. Therefore, at a given time t, $\hat{Y}$ contains the following normalized traffic data:

$$\hat{Y}(t) = \{\hat{y}(t\text{-}i), \forall\ i \in 0..m\text{-}1\} \approx \mathcal{N}(0_{m\text{x}1}, \mathrm{I}_{m\text{x}m}), \tag{5-2}$$

where $\mathcal{N}(\cdot)$ represents the multivariate Gaussian distribution with mx1 zero vector mean and mxm identity covariance matrix. This multivariate distribution is the key result of normalizing traffic by means of μ and o models. Note that the identity covariance matrix indicates unitary standard deviation for every single $\hat{y}$ value and no correlation between any pair of elements in $\hat{Y}(t)$. Hence, we can conclude that $\hat{Y}(t)$ contains independent and identically distributed random variables each following the univariate standard Gaussian distribution $\mathcal{N}(0,1)$.

According to the aforementioned properties of the normalized traffic, let us define the probability $p(i)$=1-$P(z\leq|\hat{y}\ (t\text{-}i)|)$ as an indicator of how likely is to consider $\hat{y}$ as a normal traffic value. Therefore, we assume that smaller (i.e., less probable) $p(i)$ values will be observed in case of an anomaly. Based on these individual probabilities, we define a score function $s(t)$ to compute how likely is that a data series $\hat{Y}(t)$ does not belong to the normal class. The score $s(t)$ is defined as:

$$s(t) = \frac{1}{\sqrt[m]{\prod_{i=0..m-1}\left[1 - P\left(z \leq \left|\hat{y}\left(t-i\right)\right|\right)\right]}} \tag{5-3}$$

In view of eq. (5-3), it is worth noting that the lower the probabilities of $\hat{y}$ variables, the lower the product of probabilities and inversely, the higher the score. To decide whether an anomaly is detected, we simply compare $s(t)$ against the ε_A threshold that normal data series $\hat{Y}(t)$ do not practically exceed. Then, there is sufficient evidence to detect an anomaly in OD pair in time t if $s(t) \geq \varepsilon_A$.

5.3 VNT Reconfiguration

Once a traffic anomaly has been detected, the VNT can be reconfigured to cope with the traffic increment. In this regard, a prediction of the magnitude of the anomaly would be of great interest. However, in absence of such estimation we can consider that the magnitude of traffic anomies never exceeds e.g., double of normal traffic.

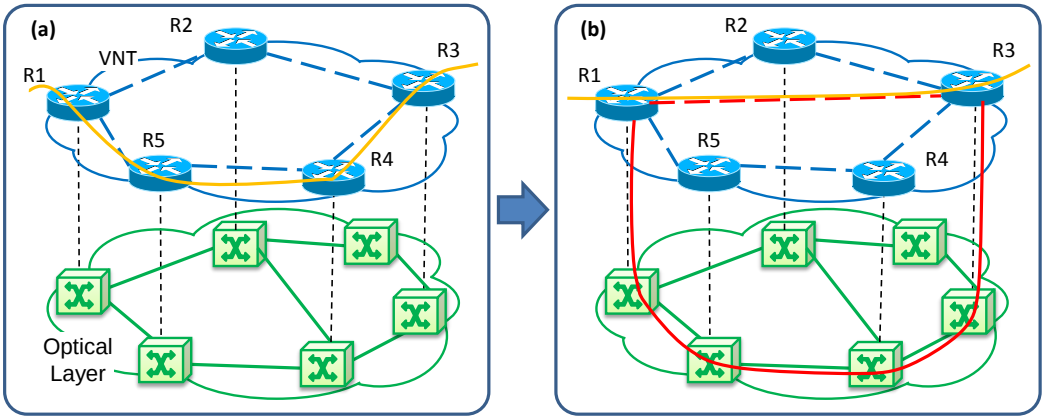


Fig. 5-2. (a) Initial VNT and OD 1→3 routing. (b) Reconfigured VNT and OD pair 1→3 routing after traffic anomaly detection.

Note that when the traffic experiments an abrupt increment in a short period of time vlinks capacity might be exceeded and thus, traffic losses appear until the new lightpath becomes available.

An example is shown in Fig. 5-2, where OD traffic is monitored (Fig. 5-2(a)) and a traffic anomaly in OD pair 1→3 is detected. The VNT is reconfigured by creating new vlink 1→3 and traffic is re-routed (Fig. 5-2(b)). As a result of the anticipated VNT reconfiguration, traffic losses might be decreased.

Based on the above, the OD traffic anomaly-triggered topology reconfiguration (ODEON) problem that we propose can be formally stated as:

Given:

- a graph $G(V, E)$, where V is the set of packet nodes and E is the set of directed vlinks connecting two nodes. The remaining capacity of each vlink e (b_e) is also known;
- a set $TP(v)$ of transponders installed in each node v. Some of these transponders can be currently used for the existing vlinks and some of them might be unused;
- the OD pair od affected by the anomaly; od is defined by the tuple $<o,d,b>$, where o and d are the source and target nodes, respectively, and b is the maximum expected bitrate during anomaly lifetime.

Output: The capacity increments in existing vlinks and the new vlinks to be created, so as to serve od. In addition, the new path for od needs to be specified.

Objective: Minimize the use of resources to serve od, including transponders utilization.

To tackle the ODEON problem, we propose the algorithm in Table 5-3. The resources currently allocated to pair od are first released (line 1 in Table 5-3) and the vlinks without enough capacity are removed from the graph (lines 2-3). Next,

the graph is augmented where new vlinks are created between routers with available transponders (line 4) and a shortest path is computed (line 5), assuming vlinks cost as follows:

$$c_e = \begin{cases} 1 & if\ e \in E \\ |E|+1 & otherwise\ (i.e., if\ e \in E' \backslash E) \end{cases} \quad (5\text{-}4)$$

Table 5-3 ODEON Algorithm

```
INPUT: G(V, E), od=<o,d,b>
OUT: <L, p>
 1:  deallocate(G, od)
 2:  for each e in E do
 3:    if availableCapacity(e) ≤ b then E ← E \ {e}
 4:  G'(V, E') ← augment(G)
 5:  p ← SP(G', <o, d>)
 6:  if p = Ø then return INFEASIBLE
 7:  L ← Ø
 8:  for each a in p do
 9:    if e=(v1,v2) ∈ E'\E then
10:      l ← RMSA(G', <v1, v2, b>, ∞)
11:      L ← L U {l}
12:  return <L, p>
```

Finally, the Routing, Modulation and Spectrum Assignments (RMSA) problem (see Chapter 2) is solved for every new vlink and for existing vlinks where the capacity needs to be increased.

5.4 Proposed Architecture and Monitoring Strategies

To efficiently implement OD-based traffic anomaly detection methods, we propose the modules depicted in Fig. 5-3 that are all of them, for the moment, assumed to be placed in the network control plane. We call this module Monitoring and Data Analytics System (MDA) In such *centralized* architecture, traffic samples are collected from packet nodes and stored in the Collected Data Repository (CR). Collected data can be conveniently summarized in modeled data, e.g., by computing average deviation, percentile, etc. The Estimator (E) module applies data analytics on samples from the Modeled Data Repository (MR) to estimate the specific models for every OD pair, which are stored in a model repository. Models predict response variables for the average OD traffic (i.e., $\mu(t)$ and $o(t)$). An Anomaly Detection (AD) module is in charge of detecting traffic anomalies; it first verifies whether a just arrived monitored OD traffic value is out-of-bounds and, only in such case, its current score is computed and compared against threshold ε_A. Upon the detection

of an anomalous OD pair, a VNT reconfiguration needs to be triggered as presented in the previous section.

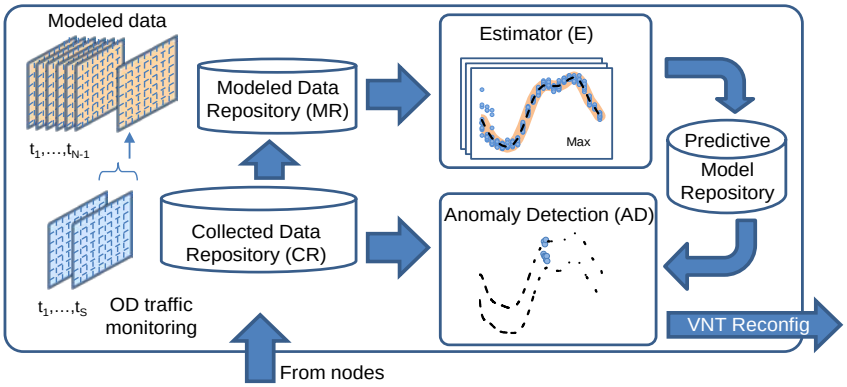


Fig. 5-3. MDA architecture for OD traffic anomaly detection.

Fig. 5-4 presents a general view of the centralized architecture with module placement, as well as the monitoring parameters that the MDA system can configure in the network nodes. Analyzing the placement, it is clear that repositories need to be centralized, since data can be used for several purposes that might require a global view of the network. Regarding model fitting, it can be carried out in the MDA from monitoring data collected every 15-minute period.

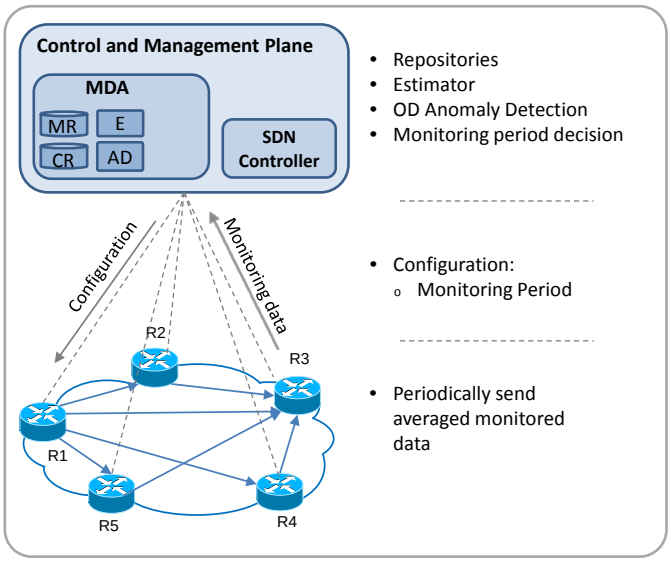


Fig. 5-4. Centralized monitoring architecture.

However, that monitoring period would impact on the time to detect OD traffic anomalies and hence, shorter monitoring period would be preferred from that viewpoint. For instance, if the target time to detect traffic anomalies is within the 5 minutes after they appear, the monitoring period cannot exceed that value (results are presented in section 5.5). In consequence, δ is a key parameter to study since

reducing it entails increasing the amount of monitoring data to be conveyed from the network nodes to the collected data repository in the MDA system.

Aiming at limiting the amount of collected data, in this chapter we propose studying the performance of the following monitoring strategies:

- the traditional *fixed* monitoring period strategy but reducing its period to accelerate anomaly detection;
- a *dynamic* monitoring strategy, where the monitoring period can be re-programmed during the day; and
- a *reactive* monitoring strategy (*c*:*f*) that uses a coarse monitoring period (*c*) and re-configures it to a finer period (*f*) after detecting the first atypical monitoring data.

From the possible combination of methods and strategies, we focus on studying the four most relevant approaches (Table 5-4):

- traffic-based with fixed monitoring (*traffic-fixed*);
- score-based with fixed monitoring (*score-fixed*);
- score-based with dynamic monitoring (*dynamic*);
- score-based with reactive monitoring (*reactive*)

Table 5-4. Anomaly detection methods and monitoring strategies

Monitoring Strategy	**Detection method**	
	Traffic-based	**Score-based**
Fixed	traffic-fixed	score-fixed
Dynamic	-	dynamic
Reactive	-	reactive

Another different option is to place the traffic anomaly detection functionality inside the network nodes, as depicted in Fig. 5-5; we call this the *distributed* architecture. This way, data analytics for anomaly detection has access to fine-grained monitoring data every δ period, which allows achieving low detection times while keeping a larger monitoring period (e.g., 15 minutes) for model fitting thus, reducing the amount of monitoring data to be collected.

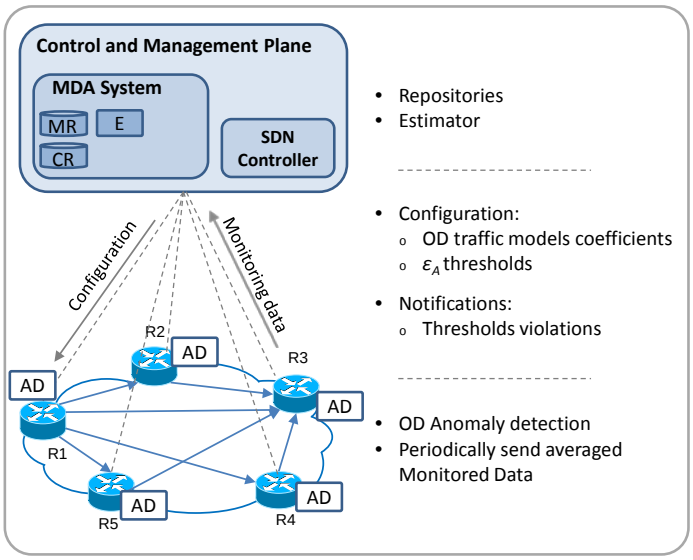


Fig. 5-5. Distributed monitoring architecture

5.5 Illustrative Results

In this section, we first study the performance of the proposed strategies and methods for traffic anomaly detection, where each anomaly affects one single OD pair. Next, the performance of the ODEON problem will be evaluated.

5.5.1 Scenario and Parameters Tuning

For evaluation purposes we developed an ad-hoc event-driven simulator in OMNET++ and considered a scenario with ten packet nodes (i.e., 90 OD pairs). OD traffic (see Appendix B for the details about traffic generation) and traffic anomalies were generated separately and subsequently combined. Traffic anomalies are generated following a pulse function (Fig. 5-6(a)), where the raising front consists of an exponential function and are used as a multiplicative factor over normal traffic (Fig. 5-6(b)). Anomalies can be configured to be triggered at any specific time and with any specific duration and scaling factor. As an example, in Fig. 5-6(c) an anomaly can be generated to multiply traffic by x1.5, last for two hours and reach 90% of its maximum value at the first 30 minutes.

The Estimator module was implemented in C++ and integrated in the simulator, whereas the AD module implementing the anomaly detector was developed in R and kept as a separated standalone module. Generated traffic values were used as input of an R function that computed the score of every OD pair and returned whether an OD exceeded a threshold.

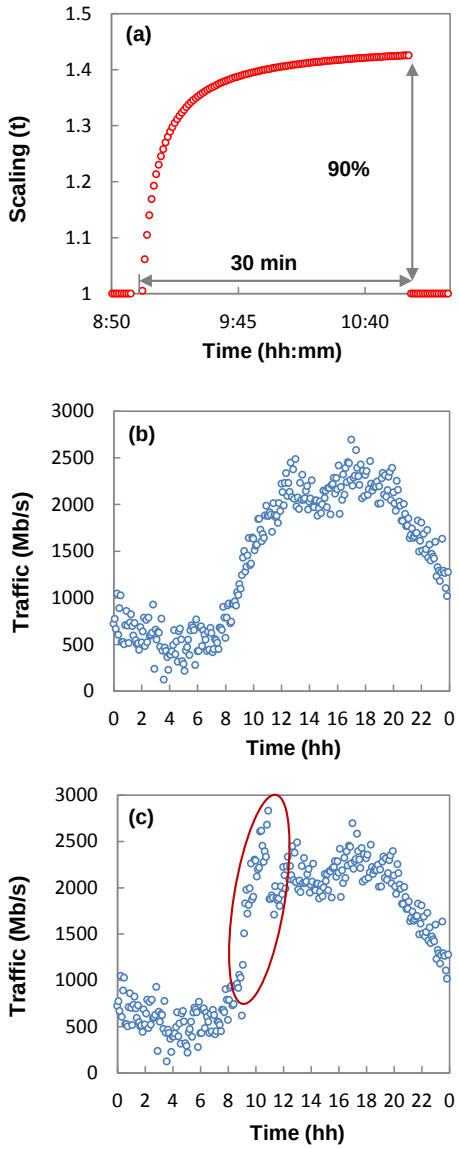


Fig. 5-6. (a) Exponential traffic anomaly. (b) OD traffic with a traffic anomaly at 9am. (c) OD normal traffic profile in a typical day

Before evaluating the performance of the previously proposed strategies and methods, some key parameters need to be determined. To this aim, we run some simulations without adding traffic anomalies, so as to produce monitoring data for the normal traffic. From those simulations, we observed that the maximum amount of consecutive atypical monitoring traffic data in an OD pair was 2. In consequence, anomalies will be detected by the *traffic-based* method when 3 or more consecutive atypical values are monitored. Regarding the *score-based* method, we considered the number of normalized traffic samples received, m=5. Finally, the value of ε_A was fixed to 1000. The rationale of such decision is left to the next chapter.

5.5.2 OD Traffic Anomaly Detection Methods

Graphs in Fig. 5-7 plot, for several hours of the day, the anomaly detection time for the considered detection methods and monitoring strategies, where the monitoring period is in the interval [1-5] minutes. We observe that although anomaly detection time varies for the different considered hours of day, detection time increases remarkably with the *traffic-fixed* approach when the monitoring period increases. This is in contrast to the moderated increment achieved by the s*core-fixed* one. In the case of the *reactive* approach, where we assume a (*c*=5:*f*=1) min. monitoring strategy, slightly lower detection times with respect to the previous approaches can be observed.

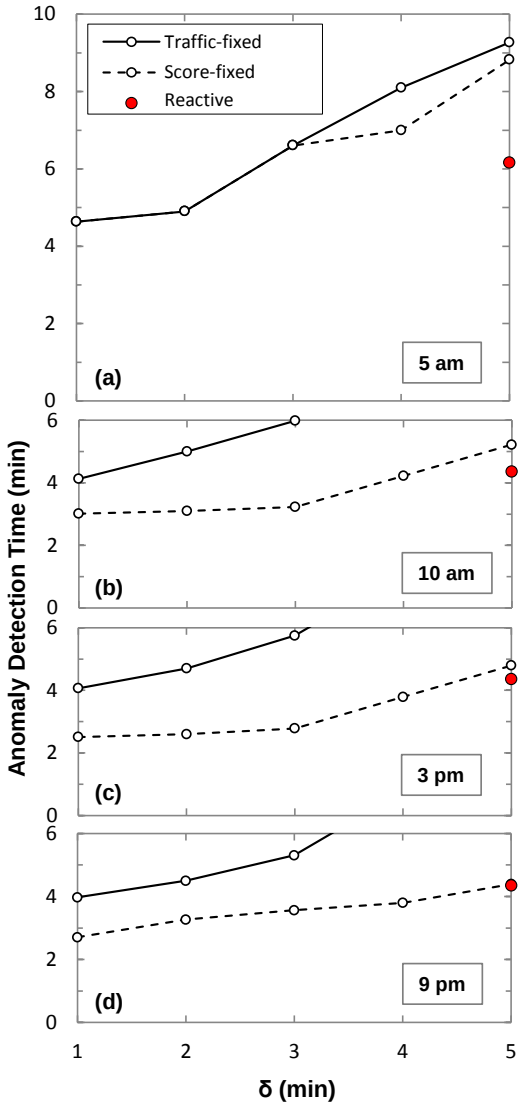


Fig. 5-7. Traffic anomaly detection time vs. monitoring period for different hours of a day at: (a) 5am, (b) 10am, (c) 3pm, and (d) 9pm.

Table 5-5. Gains from applying ALCOR (%).

Hour of day (h)	Detection Time (min)		Gain	
	Score-based		Abs. (min)	%
	Fixed	Reactive		
5am	8.8	6.2	2.6	30.2%
10am	5.2	4.4	0.8	16.4%
3pm	4.8	4.4	0.4	9.0%
9pm	4.4	4.4	>0	1%

The table in Table 5-5 reports the gains in detection time for the studied hours of day, where using a finer monitoring period after an out-of-bound traffic sample is detected provides gains between 1% and 30%.

Fig. 5-8 focuses on studying in depth the potentials of the score-fixed and illustrates how anomaly detection depends on different factors, such as the changes on traffic volume among the different hours of the day for the same monitoring period. This opens the opportunity to dynamically adapt the monitoring period for different hours of day and achieve the same anomaly detection times (Dynamic monitoring). Note that this is positive since to achieve low detection times, 1-minute period should be fixed. Hence, by relaxing the monitoring period, we are effectively reducing the amount of monitoring data to be collected in the centralized repository.

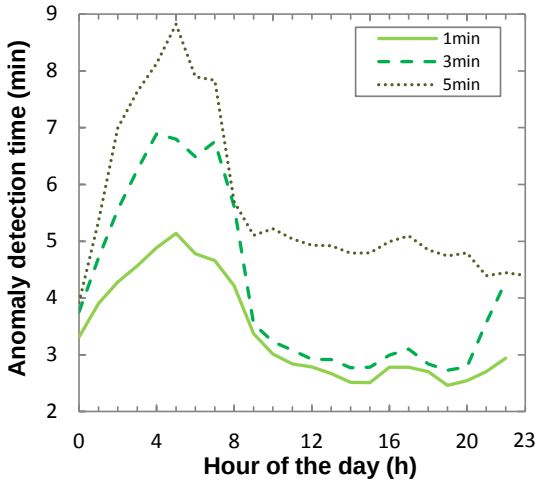


Fig. 5-8. Anomaly detection time vs. hours of day for different δ values.

Fig. 5-9 shows the amount of monitored data to be collected along the day when reducing δ. E.g., assuming a 5 minutes period, 288 monitoring samples per OD and day need to be collected achieving 5.14 and 4.62 min. detection times for the score-

fixed and the reactive approaches, respectively. Finally, Table 5-6 compares the amount of data to be collected in the centralized and the distributed architecture as a function of the required detection time for both, anomaly detection and traffic modelling. Note that the amount of data in the case of the distributed architecture is constant and equal to one sample every 15 min, since monitored data is exclusively used for traffic modelling purposes.

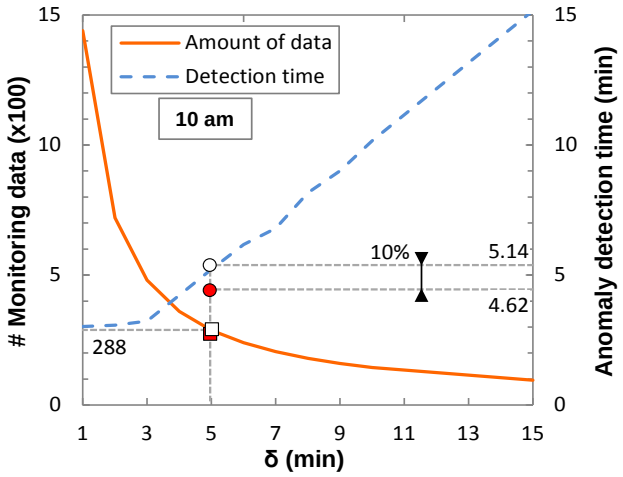


Fig. 5-9. Amount of collected data and anomaly detection time vs. δ.

Table 5-6. Amount of collected data per OD and day vs. required anomaly detection time

Monitoring Period (min)	Centralized Architecture	Distributed Architecture
3	720	96
5	360	96
10	144	96
15	96	96

Even though the different proposed methods for anomaly detection can be improved by changing the monitoring period, the best solution to avoid dealing with huge amount of monitored data is clearly to place traffic anomaly detection directly into the network node, as proposed in section 5.3.

5.5.3 VNT Reconfiguration

To evaluate the performance of the ODEON algorithm, the event driven simulator triggers a network reconfiguration in case of a sudden anomaly. Lightpath set-up path was set to 1 min. The gain in terms of traffic loss is analyzed for different values of maximum traffic expected per OD.

Table 5-7 Improvement using ODEON.

Max OD Traffic (Gb/s)	Loss (Gb) without ODEON	Loss (Gb) with ODEON	Loss reduction (%)			
			mean	1am	9am	5pm
40	15	10	33%	9%	22%	20%
60	20	13	33%	58%	13%	56%
80	28	18	38%	21%	18%	54%
100	45	28	37%	43%	17%	54%
120	49	21	56%	40%	36%	76%
140	48	27	44%	29%	18%	80%

Table 5-7 shows the results of traffic loss with and without ODEON assuming that an anomaly can occur at any hour, as well as the loss reduction at three different day times. In view of the results, we can conclude that ODEON reduces traffic losses in more than 30% in average, reaching up to 80% of reduction for specific hours and loads.

5.6 Concluding Remarks

Two different anomaly detection methods have been studied in this chapter: *traffic-based* and *score-based*. The *traffic-based* consists in detecting anomalies after receiving a number of consecutive atypical monitoring data values with respect to the $\mu \pm 3\sigma$ confidence interval, while the *score-based* method is a probabilistic classifier that assigns two labels for the response: *normal* and *anomaly*. Besides, an algorithm based on a multi-response model has been presented to predict whether a sequence of consecutive traffic data belongs to the normal class or, on the contrary, there is sufficient evidence to declare it as anomalous. Next, a traffic anomaly detection architecture has been proposed based on the following components:

- Traffic samples from every OD pair are collected at a given rate and stored in the collected data repository.
- Collected data is used by a traffic anomalies detection module, to compare its value against that of a predicted traffic model.
- An estimator module that pre-processes collected data to fit multi-response predictive traffic models for the average.

Predictive models include both, mean and deviation as response variables. All these components were first assumed to be placed centralized in the network controller.

Different function placement architectures and monitoring strategies were explored for the data analytics to perform OD traffic anomaly detection aiming at reducing anomaly detection times. It was shown that 15-minute monitoring cannot provide the short anomaly detection times required to react against unexpected traffic changes. Consequently, four different approaches mixing detection methods and monitoring strategies have been proposed using the centralized architecture; the shortest anomaly detection times were achieved when monitoring every 1 min. However, this represents a large amount of data storage in the centralized controller. In view of the above, an alternative distributed architecture was proposed where the proposed data analytics method for OD anomaly detection were moved to the network nodes thus, relaxing data collection from the centralized controller to the traditional 15-min. period, that can be used for traffic modelling and estimation purposes.

After detecting an OD traffic anomaly, a network reconfiguration can be triggered to avoid traffic losses as a result of capacity exhaustion. In this regard, the ODEON algorithm that targets at reconfiguring the VNT after a traffic anomaly is detected was proposed. Simulation results showed remarkable packet loss reduction when VNT reconfiguration was carried out.

The proposed distributed architecture presents some open aspects that can be summarized as follows:

- The network nodes need to be upgraded to introduce the anomaly detection functionality. This entails that the architecture of the network nodes needs to be extended to support monitoring and data analytics software, as well as monitoring programmability (i.e., monitoring parameter configuration). In addition, the architecture of the network controllers needs extensions to support modeling traffic from monitoring data, data analytics to make decisions (e.g., to initiate a network reconfiguration), monitoring programming, etc.
- Related to monitoring programmability, the distributed architecture needs more monitoring parameters to be specified (i.e., per-OD traffic models and score thresholds), which requires extending the interface between the controller and network nodes.
- In the migration process from nodes with and without extended capabilities, the data plane could include both node types, which would entail increasing the complexity of managing the data plane. In such cases, a hierarchical approach [Sa16] can be implemented, where monitoring data from non-extended nodes can be collated in some intermediate element implementing the extended capabilities.

The next chapter continues addressing traffic anomalies issue, in particular, traffic anomalies that arise from a common cause and affect several OD pairs simultaneously.

Chapter 6

Multiple Traffic Anomalies Detection

The previous chapter targeted single traffic anomalies detection and subsequent reconfiguration; however, an external event might cause multiple related traffic anomalies. In the case of triggering the ODEON problem to reconfigure the network just after one traffic anomaly is detected, some Key Performance Indicators (KPI) as well as traffic losses would be unnecessarily high. KPIs include the number of network reconfigurations, and the total reconfiguration time. In light of that, we propose the Anomaly and Network Reconfiguration (ALCOR) method to anticipate whether other ODs are anomalous after detecting one anomalous OD pair. Exhaustive simulation on a realistic network scenario in the case of multiple anomalies, show that ALCOR can significantly improve KPIs.

6.1 Introduction

6.1.1 Motivation and Objectives

Multiple anomalies can be caused under special circumstances, e.g., when a disaster affects the network [Na16]. In such scenarios, several ODs leaving from a common origin node ($o \rightarrow (M \subset V)$) or arriving to a common destination node ($(N \subset V) \rightarrow d$) can be produced, where M represents the set of destinations from o and N is the set of origins to d and V is the set of packet nodes in the VNT. Here, reconfiguring the VNT after an individual OD traffic anomaly is detected can result in both, an intolerable number of VNT reconfiguration, as well as and traffic losses and in a far from optimal network configuration in terms of resource utilization.

In this chapter, we study multiple OD anomalies. Specifically, the Anomaly and Network Reconfiguration (ALCOR) method is proposed to improve the following key performance indicators (KPI) in the event of multiple OD traffic anomalies:

- the number of network reconfigurations (*nReconfig*),
- the total reconfiguration time (*tReconfig*),
- the traffic losses (*lossTraffic*).

The method developed in this chapter, consists in anticipating whether other ODs are anomalous after detecting one anomalous OD pair before triggering a reconfiguration. The discussion is supported by the results from exhaustive simulation over a realistic scenario.

6.1.2 Notation

Table 6-1 presents the key notation used in this chapter.

Table 6-1. Relevant notation used in this chapter.

δ_r	time that reconfiguration process lasts
ε_A	anomalous ODs threshold
ε_S	suspicious ODs threshold
o	origin node
d	destination node
M	set of destinations from o
N	set of origins to d
nReconfig	number of network reconfigurations
tReconfig	total reconfiguration time
lossTraffic	traffic losses
$\hat{y}(t)$	normalized value of $y(t)$ with respect to the average model
$\hat{Y}$	vector with normalized traffic data samples

6.2 Dealing With Multiple OD Traffic Anomalies

6.2.1 Multiple Anomalies and Network Reconfiguration

Let us assume now that an event causes several related OD traffic anomalies. For illustrative purposes, Fig. 6-1 shows an example where a traffic anomaly has been detected in OD *R1→R3* (i.e., that threshold ε_A has been violated at time t_0); the

score evolution with time for several monitoring intervals is shown in Fig. 6-1(a). Note that a purely *per-OD* reconfiguration would immediately trigger a network reconfiguration after the *R1→R3* OD traffic anomaly is detected (Fig. 6-1(b)). Let us assume that the monitoring period δ is fixed to 1 minute and the reconfiguration process takes δ_r, e.g., 2 minutes. Then, network reconfiguration process for OD pair *R1→R3* will end at $t_0+\delta_r$. Later, at $t_0+3\delta$, an OD traffic anomaly in pair *R1→R5* is confirmed and thus, the controller triggers a new reconfiguration for such OD that finishes at $t_0+3\delta+\delta_r$. While the network reconfiguration for OD pair *R1→R5* is still in progress, another anomaly for OD pair *R1→R2* is detected at $t_0+4\delta$, but the reconfiguration for that OD anomaly needs to be delayed until that for OD pair *R1→R5* finishes. Therefore, reconfiguration for OD pair *R1→R5* starts at $t_0+5\delta$ and finishes at $t_0+5\delta+\delta_r$.

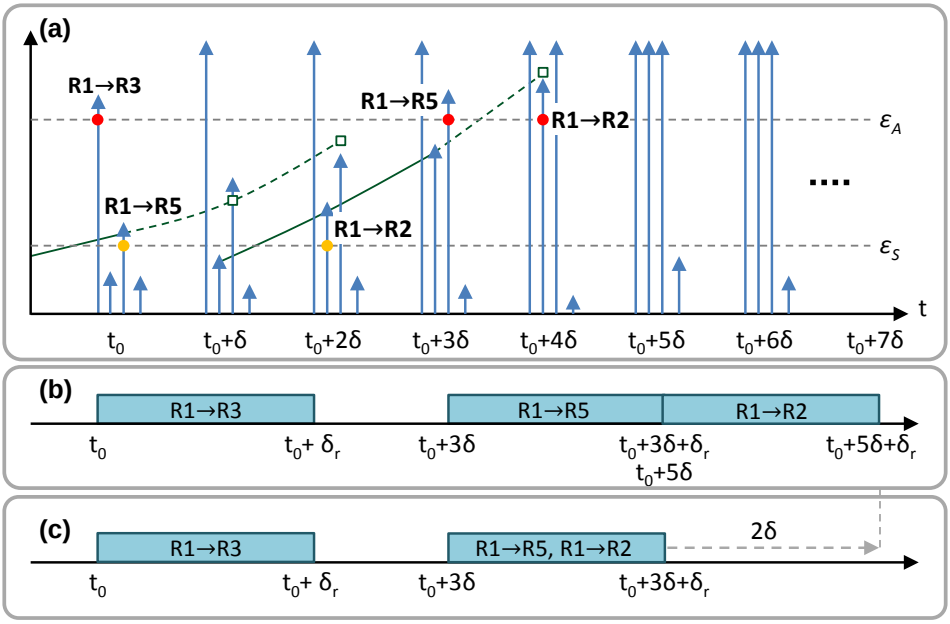


Fig. 6-1. Example of multiple anomalies. (a) Score vs. time. (b) Single and (c) multiple OD reconfiguration.

To reduce the number of reconfigurations and minimize the time when the last reconfiguration finishes, thus reducing traffic losses, a procedure to detect multiple OD anomalies would be useful Fig. 6-1(c). To that end, we propose the ALCOR method to analyze other ODs and to compare their scores against a different threshold ε_S for suspicious ODs. ALCOR aims at improving the considered KPIs by reducing the number of network reconfigurations, as well as the total reconfiguration time in the case of multiple anomalies. This is achieved by avoiding the case when anomalies are detected while the reconfiguration triggered by other anomalies is still in process. Thus, ALCOR anticipates the reconfiguration of some suspicious ODs with sufficient evidence to shortly become anomalous. Since those anomalies are related, we propose to focus that analysis on outgoing OD pairs *R1→N*, as well as on incoming ODs *M→R3*.

In the example in Fig. 6-1, a threshold violation ε_S is detected for OD pair *R1→R5* after the detection of an anomaly in OD pair *R1→R3* at time t_0. After predicting the evolution of the *R1→R5* score by means of extrapolation based on past score values (lines with square markers in Fig. 6-1(a), the occurrence of an anomaly during next 2δ in such suspicious OD pair is discarded and hence, reconfiguration of OD pair *R1→R3* is triggered at t_0. Later on, OD pair *R1→R5* anomaly arises and OD pair *R1→R2* is identified as suspicious. In this case, there is sufficient evidence to declare the latter as anomalous in the next time interval and consequently, it is jointly reconfigured with *OD* pair *R1→R5* at time $t_0+3\delta$. As a result, preparing the network for the new traffic conditions takes 2δ time units less than the total time required by the per-OD reconfiguration.

6.2.2 Evolved MDA Architecture

In Chapter 5, two architectures for single OD-based traffic anomaly detection were analyzed: centralized (Fig. 5-4) and distributed (Fig. 5-5) architecture, leading to the conclusion that distributed architecture achieved best anomaly detection times while transferring the least amount of data to the central repository. For that reason, Fig. 6-2 presents an evolved distributed architecture focused on multiple traffic anomalies, where OD anomaly detection data analytics method was brought to the network nodes (Fig. 6-3) aiming at providing fine-grained monitoring data every δ period to lower detection times. Upon the detection of an anomalous OD pair at the node, the ALCOR module in the network controller is in charge of deciding the ODs for which VNT reconfiguration needs to be triggered.

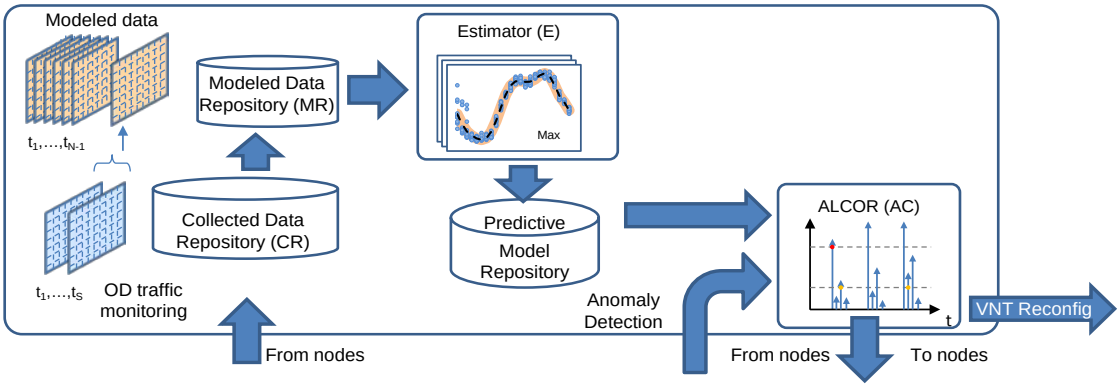


Fig. 6-2. Architecture for OD traffic anomaly detection.

Specifically, the AD module in the network nodes, detects anomalies by using as input, the traffic models that have been estimated by the Estimator in the MDA module. To be able to perform such traffic estimation, monitoring data is aggregated and conveyed periodically to the MDA system. A number of parameters need to be configured in the network nodes, including the thresholds, the traffic models, the OD pairs to be monitored, etc.

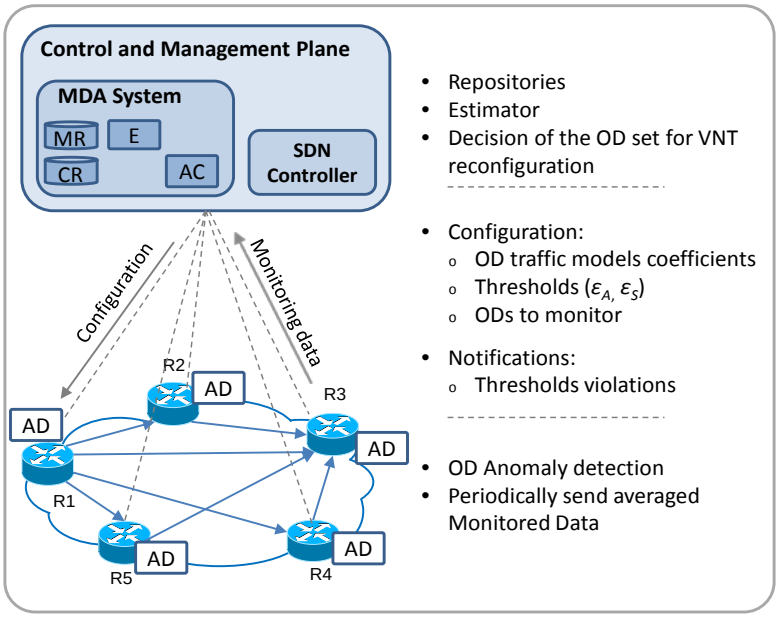


Fig. 6-3. Distributed monitoring architecture

6.3 The Anomaly and Network Reconfiguration (ALCOR) Method

This section focuses on the ALCOR method that identifies multiple related anomalies; the score value defined in eq. (5-3) is used to decide whether to trigger a network reconfiguration with only confirmed anomalous OD pairs or anticipate further anomaly evidences in suspicious OD pairs. Assuming the distributed architecture Fig. 6-3, ALCOR configures every node in the network specifying, among other parameters, the anomaly detection thresholds (ε_A and ε_S). Initially, nodes only monitor outgoing OD traffic and periodically send average values towards the controller, which estimates the coefficients to model OD traffic (Fig. 6-4(a)). The coefficients for every OD pair are forwarded to the origin and destination nodes and are used together with the ε_A threshold to detect traffic anomalies.

Whenever an OD traffic anomaly is detected by a node (i.e., OD pair R1→R3 in Fig. 6-4), a notification is sent to the controller and, as a result, ALCOR decides to activate the notification for threshold ε_S in origin node for all outgoing OD pairs $o \rightarrow N$, as well as to activate traffic monitoring for the incoming ODs $M \rightarrow d$ and ε_S threshold notification in the destination node (i.e., R1→{R2, R3, R4, R5} and {R1, R2, R4, R5}→R3 in Fig. 6-4(b)).

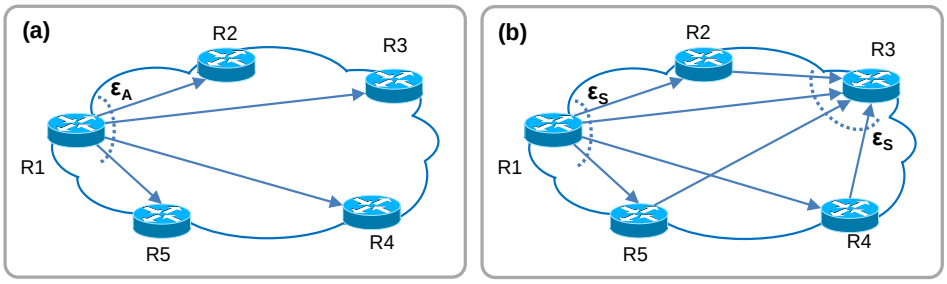


Fig. 6-4. (a) OD pair R1→R3 anomaly detection. (b) Anomaly threshold notification reconfigured in R1 and R3.

ALCOR decision problem can be addressed by solving the algorithm presented in Table 6-2. The algorithm receives an *od* in the set of OD pairs in the network, for which an anomaly threshold *thr* (either ε_S or ε_A) has been exceed at time t and returns a set of tuples with the OD pairs to be reconfigured together with the threshold violated and the time of such event. Note that in normal conditions, network nodes are configured to send notifications only after threshold ε_A is exceeded and therefore, no notification is sent when the score for any OD pair exceeds ε_S. Only when one network node notifies about an OD pair threshold violation some network nodes will be configured to send notifications for ε_S violations in some of the monitored OD pairs. In that regard, line 1 in Table 6-2 checks whether ALCOR can start making decisions.

Table 6-2. ALCOR Algorithm.

```
INPUT: od, thr, t
OUTPUT: Sol
 1:  if decisionOngoing=false AND thr= εS then return ∅
 2:  if decisionOngoing=false then
 3:    Sol ←∅; MSi ←∅; MSo ←∅
 4:    decisionOngoing = true
 5:  if thr= εA then
 6:    <MA, MSi, MSo>←reconfigureODMonitoring(MA, MSi, MSo, od)
 7:    Sol ←Sol U {<od,thr,t>}
 8:    configureTimer (time, ALCOR(∅, 0, 0))
 9:    return ∅
10:  if od≠ ∅ then
11:    Sol ←Sol U {<od,thr,t>}
12:    removeTimer()
13:  Sol←decideReconfiguration(Sol,t) then
14:  removeMonitoring(MSi)
15:  configureMonitoring(MSO, εA)
16:  MA ←MA U MSo
17:  decisionOngoing = false
18:  return Sol
```

In case that no decision process is started yet, auxiliary sets are initialized and variable *decisionOngoing* is set (lines 2-4). Next, the MDA system configures

monitoring parameters to detect ε_S violations for every other OD pairs leaving node o and entering node d (line 5). Note that initially only outgoing OD traffic was monitored to detect ε_A violations. Sets of monitored ODs are updated, containing those ODs being monitored for threshold ε_A (M_A), outgoing ODs for threshold ε_S (M_{So}) and incoming ODs for threshold ε_S (M_{Si}). The partial solution *Sol* is updated with the detected anomaly and a timer is started waiting for new evidences (lines 7-8). When the controller receives an ε_S violation, *Sol* is updated and timers disabled (lines 10-12).

The function *decideReconfiguration*(·) detailed in Table 6-3 is used to select which ODs in *Sol* need to be actually reconfigured. Recall that *Sol* contains both anomalous and suspicious ODs; the former subset will be always reconfigured (lines 3-4 in Table 6-3), whereas ODs in the latter group are analyzed one by one and selected (or not) for reconfiguration (lines 6-12). Specifically, the normalized monitored traffic $\hat{Y}$ is used to estimate the expected normalized traffic for the next δ_r time. Although normalized traffic presents a stationary behavior around zero mean under normal conditions, the presence of an anomaly dramatically alters this behavior causing sharp increasing trend during anomaly lifetime.

Table 6-3. decideReconfiguration algorithm.

```
INPUT: Sol, t
OUTPUT: Sol*
 1:  Sol* ← ∅
 2:  for each i in Sol do
 3:    if i.thr = εA then
 4:      Sol* ← Sol* U {i}
 5:    else
 6:      Ŷ(t) ← getNormalizedTraffic(i.od)
 7:      Ŷ'(t) ← BoxCoxTransform(Ŷ(t))
 8:      for k=1.. δr do
 9:        ŷ'(t+k) ← linearExtrapolation(Ŷ'(t+k-1))
10:        ŷ(t+k)  ← BoxCoxTransform-1(ŷ'(t+k))
11:        Ŷ(t+k) ← update(Ŷ(t+k-1), ŷ(t+k))
12:        s(t+k) ← computeScore(Ŷ(t+k))
13:        if s(t+k)≥εA then
14:          Sol* ← Sol * U {i}
15:          break
16:  return Sol*
```

The procedure to predict future normalized traffic is based on linear extrapolation from observed $\hat{Y}$ data series assuming that observations in $\hat{Y}$ are linearly correlated with time, i.e., $\hat{y}(t)$ values in $\hat{Y}$ can be expressed a linear function of t. To guarantee that linearity, we obtain $\hat{Y}'$ as the transformed $\hat{Y}$ data series after applying a Box-Cox transformation [Ch02] (line 7). In brief, that procedure finds the transformation parameter λ that returns $\hat{Y}'$ with the highest linear correlation with respect to time. Once λ is obtained, every $\hat{y}'$ in $\hat{Y}'$ is computed as follows:

$$\hat{Y}'(t) = \begin{cases} \dfrac{\hat{y}(t-i)^{\lambda} - 1}{\lambda}, \forall i \in 0..m-1 & \lambda \neq 0 \\ \log(\hat{y}(t-i)), \forall i \in 0..m-1 & \lambda = 0 \end{cases} \quad (6\text{-}1)$$

Fig. 6-5 shows examples of data series $\hat{Y}$ under normal traffic and under an anomaly, where plots show values in $\hat{Y}$ (markers), as well as the trend line of those values with respect to time (dashed line). Under normal traffic (Fig. 6-5(a)), normalized traffic values present stationary behavior around 0, as well as no linear correlation. However, under unexpected traffic $\hat{Y}$ shows a remarkable increasing trend with time, which might be non-linear (Fig. 6-5(b)). We propose the Box-Cox transformation in equation (6-1) to prepare data for linear regression fitting (Fig. 6-5(c)). Therefore, linear extrapolation can likely predict expected future normalized traffic in the event of an anomaly.

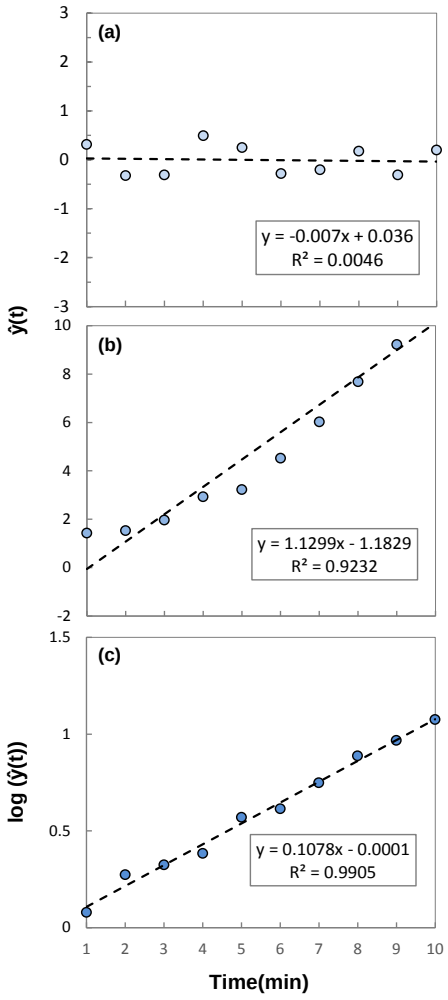


Fig. 6-5. Example of data series $\hat{Y}$ under normal traffic (a) and under an anomaly (b). Box-Cox transformation is applied in (c).

Normalized traffic for the future δ_r time is then forecast (line 8) and the obtained value is used to update $\hat{Y}$ (lines 9 and 10); updated $\hat{Y}$ contains monitored normalized traffic in the first m-x positions and forecast normalized traffic in the last x positions. This data is used to compute the predicted score (eq. (5-3)) and compared against ε_A (lines 11 and 12).

6.4 Illustrative Results

In this section, we apply ALCOR when multiple anomalies arise together and study the obtained performance.

6.4.1 Scenario

For evaluation purposes we extended the event-driven simulator in OMNET++ used in Chapter 5 generate multiple anomalies; the anomaly detector and the ALCOR modules were developed in R and kept as a separated standalone modules. Traffic anomalies were generated following the same procedure defined in the previous chapter (Fig. 5-6a), but in this scenario:

- inter arrival time between anomalies is set between 1-5 minutes,
- involves a maximum of 6 nodes

In order to tune ε_S threshold value, we analyzed its impact on the defined KPIs (*nReconfig*, *tReconfig* and *lossTraffic*). We selected ε_S equal to10 that is high enough for the sample to be out of the interval $\mu \pm 3\sigma$ and it is low enough to allow that many OD traffic pairs to be considered by the ALCOR method.

The value of ε_A needs to be also tuned (recall that it was fixed to 1000 in Chapter 5). To this end, we run experiments to study how the influence of that threshold on the selected KPIs: *i*) the number of reconfigurations that need to be performed when multiple anomalies arrive; *ii*) the total time required to detect all the anomalies and to reconfigure the network; and *iii*) normalized volume traffic that is lost due to network congestion before reconfiguration finishes (we assume that all traffic exceeding 3σ will be lost). Fig. 6-6 shows the results; we can conclude, in view of the figure, that $\varepsilon_A = 1000$ provides the best overall results.

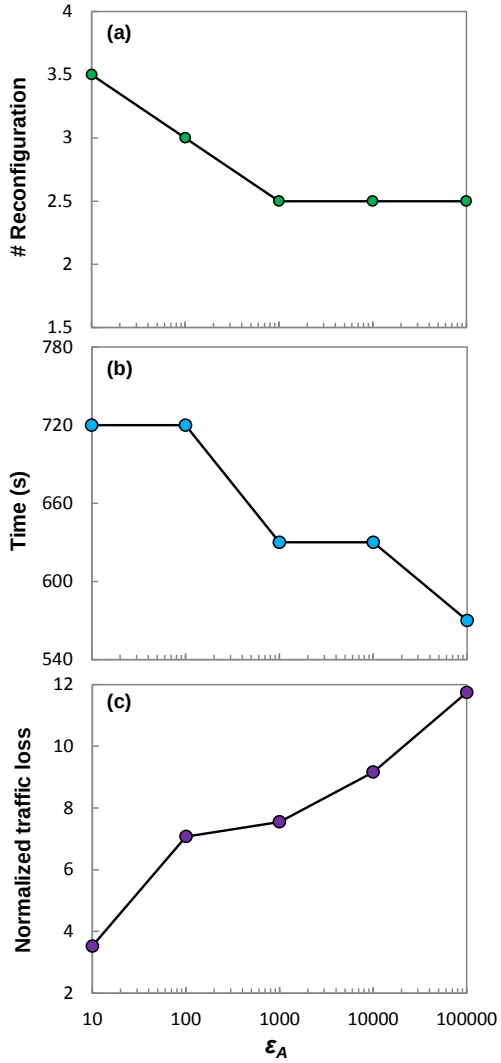


Fig. 6-6. KPIs vs ε_A.

6.4.2 ALCOR Performance

Let us first evaluate the accuracy of the decisions that ALCOR makes. Recall that ALCOR applies traffic linear extrapolation and the result is used as input to compute the expected score and to make reconfiguration decisions based on that forecast data (see *decideReconfiguration*(·) function in Table 6-3). Regarding the *score-based* method, we considered reconfiguration time δ_r equal to 2 min. Let us define that a decision in time t is incorrect if the *decideReconfiguration*(·) function predicts the scores in the interval $[t+1, t+\delta_r]$ for a given OD pair to be below ε_A (i.e., reconfiguration decision is not made at time t), while later, at some time t in the interval, some score actually exceeds ε_A. In our simulations, the *decideReconfiguration*(·) function was triggered 300 times, resulting in incorrect

decisions in only 5.65% of the times. Therefore, the goodness-of-fit of the traffic linear extrapolation model for score prediction purposes is validated.

Let us now consider multiple anomalies affecting different OD pairs. As concluded in the previous subsection, we assume the distributed monitoring architecture. Let us now study the performance of the proposed ALCOR method and compare that against a purely per-OD reconfiguration strategy. In addition, to evaluate the performance of the predictive capacity of ALCOR, we compare its performance against running the ALCOR algorithm under a perfect information assumption (ALCOR-PI), where the decision of reconfiguration is made with perfect knowledge of the future, i.e., it makes perfect decisions.

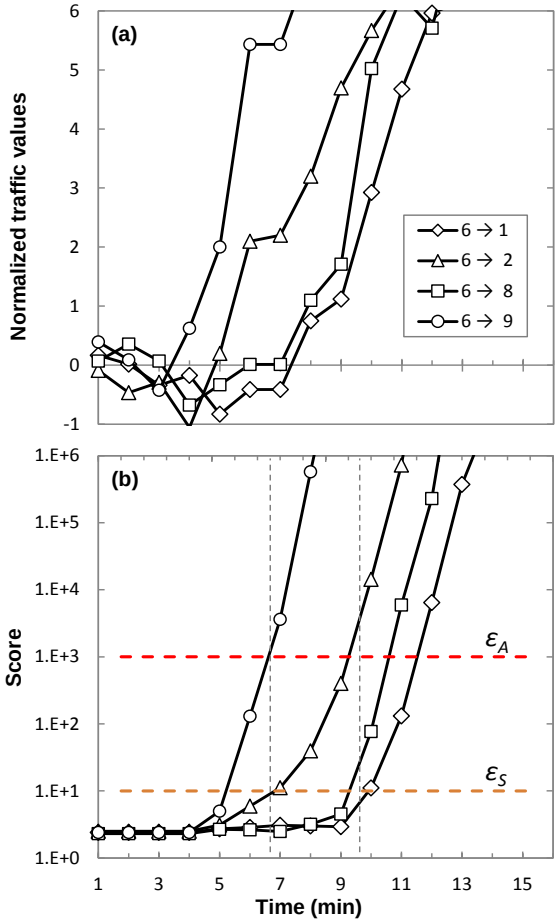


Fig. 6-7. Normalized traffic (a) and score (b) vs. time for anomaly scenario 1.

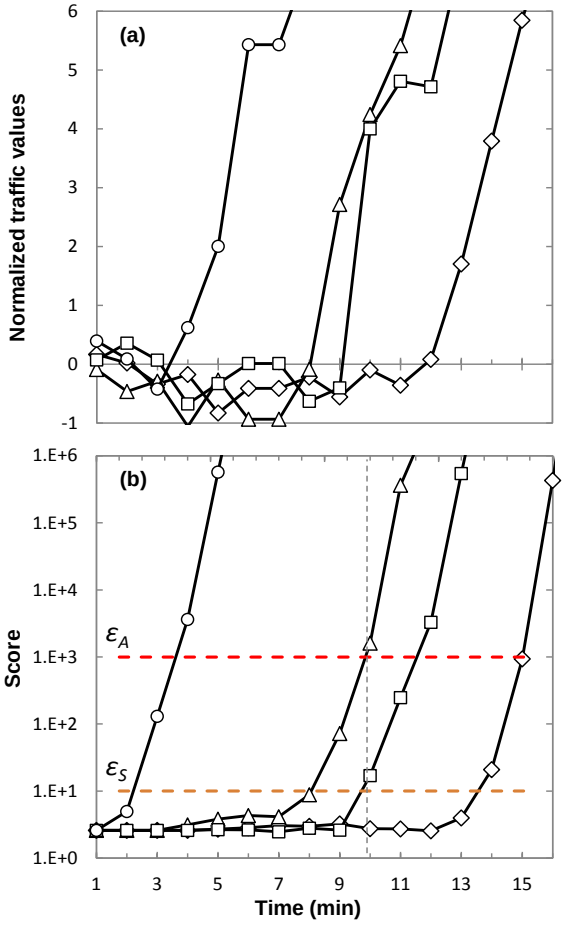


Fig. 6-8. Normalized traffic (a) and score (b) vs. time for anomaly scenario 2.

In order to evaluate the performance of the proposed ALCOR algorithm, we focus on the KPIs previously defined. Graphs in Fig. 6-7 and Fig. 6-8 present the results for two different scenarios, when four traffic anomalies arrive close in time one to the other (scenario 1) or more spaced (scenario 2). Fig. 6-7(a) and Fig. 6-8(a) plot the evolution of normalized traffic values as a function of the time; under normal conditions they should be centered on zero (the mean value) and within some

interval in terms of σ. When a traffic anomaly occurs in an OD, the normalized traffic changes sharply. Notwithstanding that sharp variation, it is difficult to set thresholds for normalized traffic values since the probability of observing values out of normal boundaries (e.g., 3σ) is not negligible and therefore, the score presented in eq (5-3) that considers previous traffic values is used. Fig. 6-7(b) and Fig. 6-8(b) plot the computed scores against time for the four affected ODs under the considered scenarios. Note the difference between normalized traffic and score for OD pairs 6→9 and 6→2 in scenario 1 (Fig. 6-7(a)); although the normalized traffic value is around 2σ at minute 5 and minute 8 for OD pairs 6→9 and 6→2, respectively, their score values are under ε_S for OD pair 6→9 and above ε_S for OD pair 6→2.

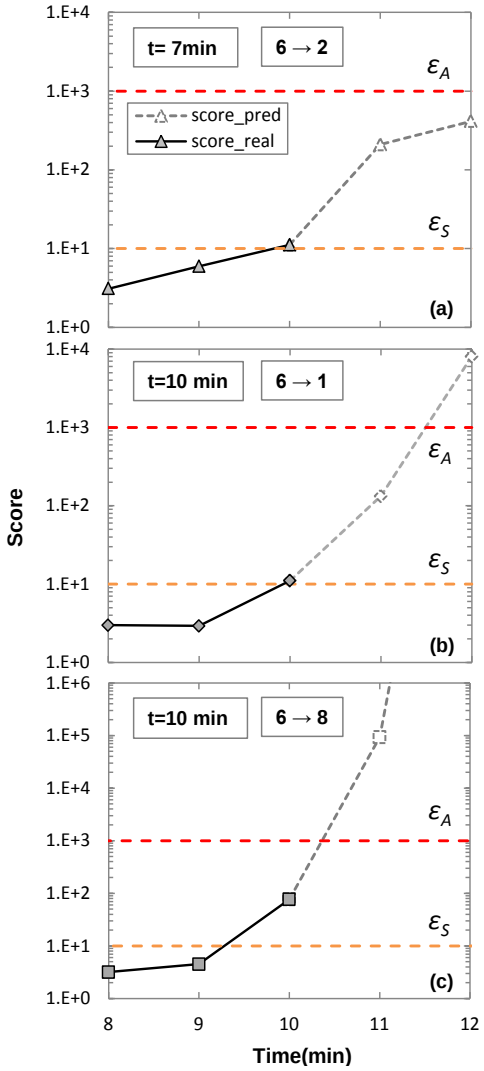


Fig. 6-9. Real and predicted score values against time for the three OD pairs in anomaly scenario 1.

Comparing both scenarios, it is clear that ALCOR would made different decisions in each case. For instance, in scenario 1, when the first anomaly in OD pair 6→9 is detected at minute 7, OD pair 6→2 has already exceeded ε_S threshold. In this scenario, ALCOR would detect that and run the *decideReconfiguration*(·) function. To clarify score extrapolation, Fig. 6-9 plots the score values computed when an anomaly has been detected (dark markers) and the predicted score values for the next two minutes (dotted markers) for those OD pairs with scores exceeding ε_S threshold. Then, ALCOR will find that OD pair 6→2 will not become anomalous in the next 2 minutes (see predicted scores for OD pair 6→2 in Fig. 6-9) and it will trigger a network reconfiguration only for OD pair 6→9. Later, at minute 10 OD pair 6→2 violates ε_A; at that time, score extrapolation anticipate that OD pairs 6→1 and 6→8 will become anomalous in the next 2 minutes (see Fig. 6-9), so ALCOR triggers a network reconfiguration for all three OD pairs. Interestingly, the *per*-OD reconfiguration would need 3 reconfigurations, 1 more than ALCOR and delays the network adaptation in 2 minutes.

Similarly, the first OD traffic anomaly is detected in pair 6→9 at minute 4 under scenario 2. Nonetheless, since no evidences of other anomalous ODs are found, ALCOR decides to trigger a reconfiguration for that OD pair. When a new OD traffic anomaly is detected in pair 6→2 at minute 10, OD pair 6→8 also violates ε_S threshold and shows evidences of becoming anomalous in the next 2 minutes, so ALCOR triggers a new reconfiguration for both ODs. Finally, a third reconfiguration is triggered for OD pair 6→1 at minute 15. Note that the *per*-OD strategy needs four reconfigurations and, although the network would be adapted to the new conditions at the same time as in the case of ALCOR, the *per*-OD strategy increases traffic losses in a 27%.

In view if the above, it is interesting to study the performance of ALCOR under several traffic anomaly inter-arrival times and let us consider the scenario where six ODs are affected and assuming an anomaly scaling factor x2.

Table 6-4. Gains from applying ALCOR (%).

Avg. Interarrival time (min)	nReconfig	tReconfig	lossTraffic
0.5	45.5	39.9	23.5
1.0	47.7	36.4	28.3
1.5	38.2	21.3	12.0
2.0	38.2	18.2	19.7
2.5	32.7	6.4	10.6
3.0	17.2	0	8.1

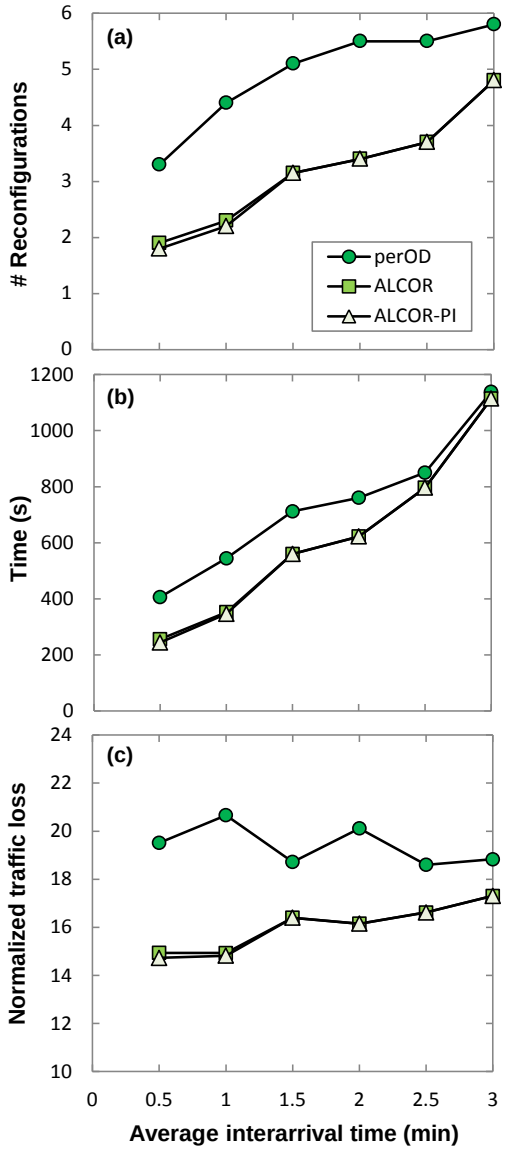


Fig. 6-10. Number of reconfigurations (a), total reconfiguration time (b), and normalized traffic loss (c), against the anomalies inter-arrival time.

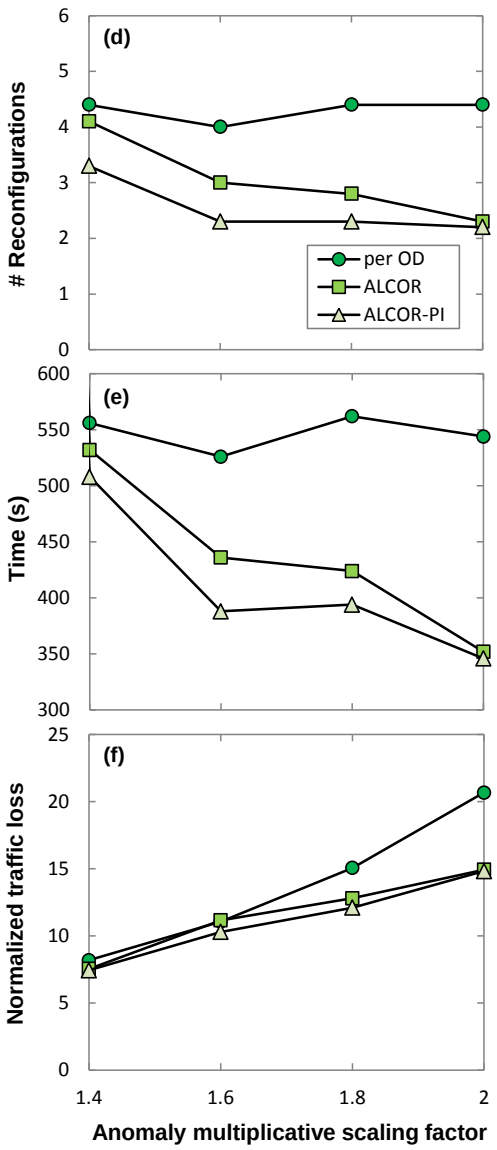


Fig. 6-11. Number of reconfigurations (a), total reconfiguration time (b), and normalized traffic loss (c), against the anomalies scaling factor.

Fig. 6-10 shows that applying ALCOR algorithm results in an improved in the defined KPIs, i.e., reduction in *nReconfig* (Fig. 6-10(a)), *tReconfig* (Fig. 6-10(b)), and *lossTraffic* (Fig. 6-10(c)), compared to the per-OD strategy even when anomaly inter-arrival time is as high as 3 min (larger than that of the considered reconfiguration time). Table 6-4 summarizes the gains from applying ALCOR.

Interestingly, ALCOR performs virtually like ALCOR-PI; however, this could be as a result of the high value of the considered anomaly scaling factor, so let us analyze the influence of the traffic anomaly intensity on the performance of ALCOR (Fig.

6-10). We studied the range for anomaly scaling factors from x1.4 up to x2, considering that lower scaling factors can hardly considered as traffic anomalies since they produce traffic values within the limits of normal traffic. One can observe how the intensity affects both the *nReconfig* (Fig. 6-11(a)), the *tReconfig* (Fig. 6-11(b)), and *lossTraffic* (Fig. 6-11(c)), showing ALCOR a better performance than that of the per-OD approach in all the cases and getting closer to the ALCOR-PI method as soon as the scaling factor increases.

6.5 Concluding Remarks

In the event of multiple related anomalous OD pairs, its detection can be spread along the time, which would entail unnecessary number of network reconfigurations and long total reconfiguration times. In view of that, the ALCOR method is proposed to identify multiple related OD traffic anomalies targeting at reducing the considered traffic losses coming from unexpected traffic increments, as well as the number of network reconfigurations performed to consequently adapt the network capacity. Since ALCOR needs to access monitoring data from several network nodes, it is placed in the centralized controller. Simulation results show remarkable saving in the KPIs (e.g., number of reconfigurations, the total reconfiguration time and the traffic losses) compared to a per-OD reconfiguration strategy.

Chapter 7

BER Degradation Detection and Failure Identification

Previous chapters focused on L2 traffic anomaly detection and VNT reconfiguration. Let us now center our attention on the optical layer where optical connections might support vlinks in packet-over-optical multilayer networks. Therefore, QoT degradation in the optical layer L0 impacts on the QoS of the services deployed on top of such networks. Monitoring the performance of the physical layer allows verifying the proper operation of optical connections, as well as detecting BER degradations and anticipating connection disruption. In addition, failure identification facilitates localizing the cause of the failure by providing a short list of potential failed elements and enables self-decision making to keep committed service level.

In this chapter, we analyze several soft failure causes affecting the QoT of optical connections and propose two different algorithms: one focused on detecting significant BER changes in optical connections, named BER Anomaly Detection (BANDO), and the other focused on identifying the most probable failure pattern, named Failure Identification Algorithm (LUCIDA).

Assuming the distributed architecture proposed in previous chapters, BANDO would run inside the network nodes to accelerate degradation detection and send a notification to the LUCIDA algorithm that would run on the centralized MDA system. Experimental measurements were carried out on two different setups to obtain values for BER and received power; these allow to generate synthetic data used in subsequent simulations. Results show significant improvement anticipating maximum BER violation and small failure identification errors.

7.1 Introduction

7.1.1 Motivation and Objectives

This chapter is focused on soft failures occurring at the optical layer; recall Chapter 2 where soft failures were introduced. It is hard to discern the real cause of soft failures in view of Fig. 2-10 , since, filters misconfiguration and transmitter laser shift could lead to similar evidence. However, discovering and identifying a failure pattern can reduce remarkably the subsequent failure localization effort by providing a short list of potential failed elements (e.g., filters used by a certain connection). Moreover, failure identification enables self-decision making to keep committed service level, e.g., by triggering re-routing of the traffic used by a connection where a gradual BER degradation has been detected.

For this very reason, in this chapter, we focus on BER degradation detection and failure identification. Specifically, the contribution of this chapter is three-fold:

- Section 7.2 motivates the definition of two different algorithms: *i*) the BANDO algorithm focused on detecting significant BER changes in optical connections, and *ii*) the LUCIDA algorithm that identifies the most probable failure pattern.
- In Section 7.3 the proposed BANDO and LUCIDA algorithms are described in detail. BANDO algorithm is designed to follow the metered BER and to raise notifications in case of abrupt BER changes. LUCIDA is a probabilistic algorithm that analyzes time-series from monitoring and notifications and returns the most probable failure class together with its probability.
- Experimental measures for BER and P_{Rx} obtained from two different setups are reported in Section 7.4. Based on experimental measurements, realistic scenarios are generated, and exhaustive simulations are run, where obtained results show the performance of the proposed algorithms.

7.1.2 Notation

It is worth noting that in the rest of the chapter, unless explicitly stated otherwise, we just use failure localization for the sake of brevity. Besides, Table 7-1 presents the key notation used in this chapter.

Table 7-1. Relevant notation used in this chapter.

Various	
n	capacity of vector inside the node
M	monitoring data: tuple of length n containing metered {<t,ber, P_{Rx} > data for every connection received at a given rate
M.Ds, M.Dns	*stationary D* contains oldest samples and the *non-stationary D* the most recent samples
M.N	monitoring notifications: tuple of length n {<t,$type$,$data$>}
ber	last BER received value
P_{Rx}	last value of received power
$\mu(\cdot),\sigma(\cdot)$	mean and standard deviation of the last n BER measures
δ	parameter to compare BER threshold with
$\Delta(i, j)$	returns the relative difference of window i with respect to j.
Q, H	set of failure classes and relevant features
$P_{RX}high$	P_{Rx} above the reference level
BERTrend	BER positive trend
BERPeriod	BER periodicity
α	$\alpha \in [0,1)$ is the minimum allowed probability in $F(x)$ (the cumulated probability when the feature takes the value x)
Boundaries and Thresholds	
Bound	outer boundary to anticipate BER threshold violation
k	values for inner and outer boundaries
lBound	lower inner boundary limit
uBound	upper inner boundary limit
BER max	the maximum pre-FEC BER that the equipment can correct
BER threshold	represents the maximum tolerable BER for a connection
tmax	time when maximum BER would be reached
Notifications	
bExc, bCh	notification for boundary excess and boundary re-estimation
thExc, thDec	notification for upper threshold exceeded and fallen below

7.2 Soft Failure Detection and Failure Identification

As introduced before, a gradual degradation of the optical signal might cause service losses to client demands. For illustrative purposes, Fig. 7-1 plots the pre-FEC BER evolution with time when a gradual BER degradation starts impacting an optical connection; we assume that BER is monitored by the receivers of the connection. Many commercial equipment, such as the ones used in our experiments in section 7.4.1, tolerate some amount of errors until automatically tear-down the connection when some BER threshold is exceeded. Notwithstanding a restoration procedure could be started to recover the affected traffic after the disruption is detected, it would be desirable to anticipate such event and re-route those demands according to SLAs. The proposed BANDO algorithm detects changes in the monitored BER measured in the receptor of an optical connection, focusing on anticipating such detection as much as possible and leaving enough time to plan a re-routing procedure e.g., during off-peak hours.

As for the failure identification, we propose an algorithm named as LUCIDA that analyzes monitoring time-series and, based on the expected patterns of the considered failure causes, identifies the most probable cause of failure affecting a given set of optical connections.

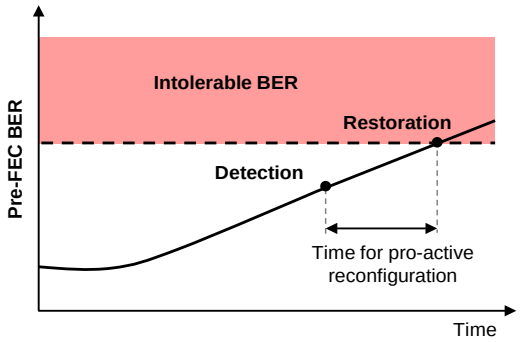


Fig. 7-1. Monitoring data stream

Information retrieved by commonly used power monitors can be combined with monitoring information accessible through emerging transponders based on coherent detection [Na15]. In particular, such transponders offer the possibility to monitor several parameters associated to connections or to the traversed links: e.g., pre-FEC BER or linear dispersion.

7.2.1 Considered Soft Failures

For illustrative purposes, Fig. 7-2 plots the evolution with time of pre-FEC BER and P_{Rx} monitoring data metered at the receiver side of a connection affected by each of the failures above-described. In the case of *Signal Overlap* (SO) (Fig.

7-2(a)), the allocation of a neighboring optical connection results in a sudden increment in both, BER and P_{Rx}, of the previously established connection. In the case of the new connection, high pre-FEC BER and within limits P_{Rx} values can be measured just after its set-up. As for *Filter Tightening* (FT) (Fig. 7-2(b)), it happens when a too much narrow filter configuration distorts the signal; such effect may become even more relevant when the signal drifts (e.g., due to a laser drift) toward the rising edge of the filter. Similarly as it happened for newly established connection in the previous case, high pre-FEC BER and P_{Rx} values within limits can be measured in the receptor.

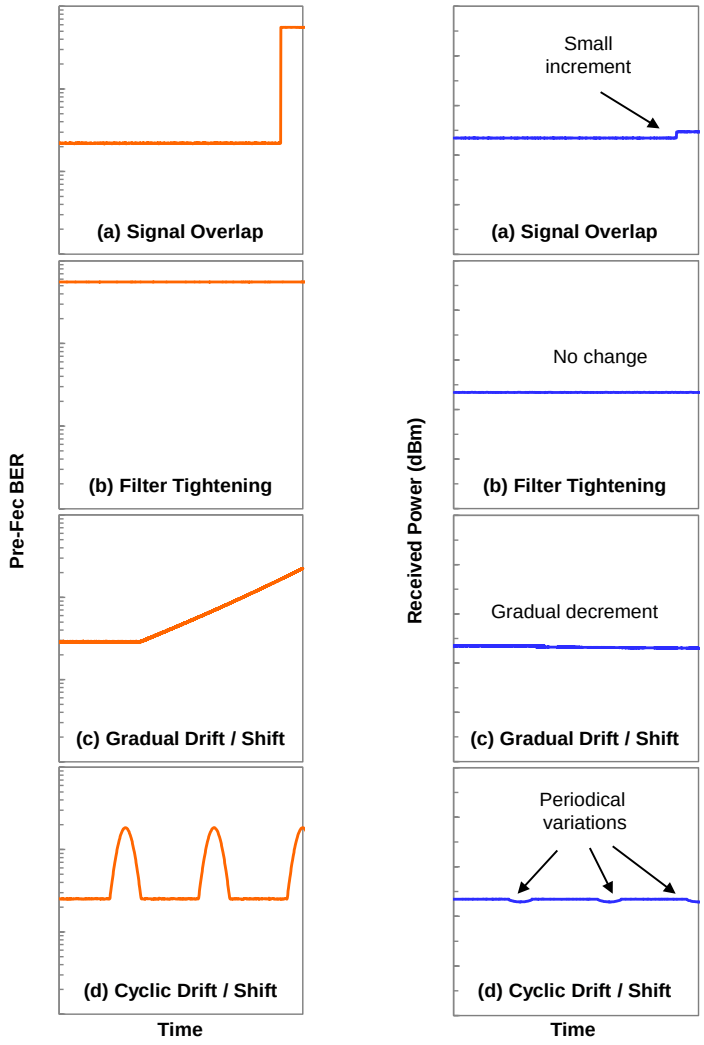


Fig. 7-2. Example of pre-FEC BER and P_{Rx} monitoring time series for the considered BER degradation failures: (a) SO, (b) FT, (c) LD/FS, and (d) cLD/cFS.

In the case of *Filter Shift* (FS) or *Laser Drift* (LD) (Fig. 7-2(c)), pre-FEC BER shows a gradual deterioration with time, while measured P_{Rx} reduction is almost imperceptible. Finally, in the case of *Cyclic Filter Shift* (cFS) or *Cyclic Laser Drift* (cLD) (Fig. 7-2(d)), high pre-FEC BER and slight P_{Rx} reduction periods when part

of the signal is out filters' bandwidth are followed by normal values when the signal is inside them. Note that any combination of the previous failures might happen, e.g., a *Gradual Cyclic Drift* (gCLD) would produce increasingly higher pre-FEC BER periods. These cyclic failures are especially difficult to identify due to its periodic nature.

It is hard to discern the real cause of the above failures since transmitter laser degradation, and filters misconfiguration could lead to similar evidence. In chapter Chapter 7 and Chapter 9, we would concentrate in the prompt detection of pre-FEC BER degradation and in the identification of the failure pattern as presented in Fig. 7-2.

7.2.2 Proposed Architecture

Let us propose the placement of the algorithms considered in this chapter. In order to detect BER degradation and identify the cause failure, we propose two different algorithms that work in a coordinate manner. For BER degradation detection, we propose the BANDO algorithm that can be placed inside network nodes, close to the observation points, to reduce the amount of monitoring data to be conveyed to the control/management plane.

Because of its target, LUCIDA needs to be placed on the MDA system, where monitoring data from different nodes, as well as operational data regarding the optical connections are available.

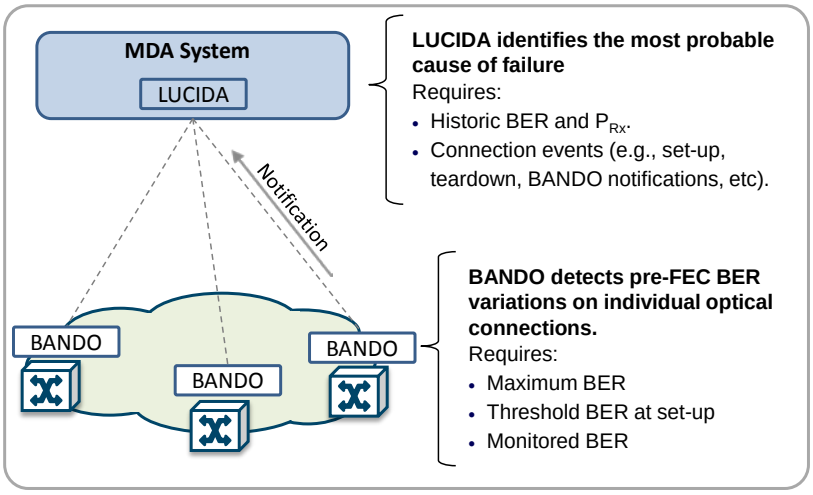


Fig. 7-3. Proposed architecture and algorithm features.

Fig. 7-3 presents the suggested architecture and algorithm placement. The BANDO algorithm runs inside the optical nodes and has access to fine-granular monitoring data to accelerate BER degradation detection. Once BER variation is detected, a notification is sent towards the MDA system for further analysis, which triggers LUCIDA for failure identification. Main features of the proposed algorithms are

also summarized in Fig. 7-3. Depending on the particular case, different reconfiguration algorithms can be triggered after the failure has been identified.

7.3 Algorithms for BER Degradation Detection and Failure Identification

In this section, we define BANDO and LUCIDA algorithms in detail.

7.3.1 BANDO Algorithm

We assume that metered pre-FEC BER and P_{Rx} data for every connection is received at a given rate (e.g., every minute) and stored in a vector *M* of fixed capacity *n* in the node. BANDO analyzes pre-FEC BER data to detect gradual changes with time that might derive into BER degradation and intolerable BER values, as well as sudden anomalous BER values.

Fig. 7-4 illustrates three cases of BER evolution with time, where the dark continuous line represents monitored BER. Besides, two different limits are presented:

- *BER max* is the maximum pre-FEC BER that equipment can correct
- a BER threshold for the current connection computed as a function of the estimated BER (e.g., 5*estimated BER) and represents the maximum tolerable BER for such connection.

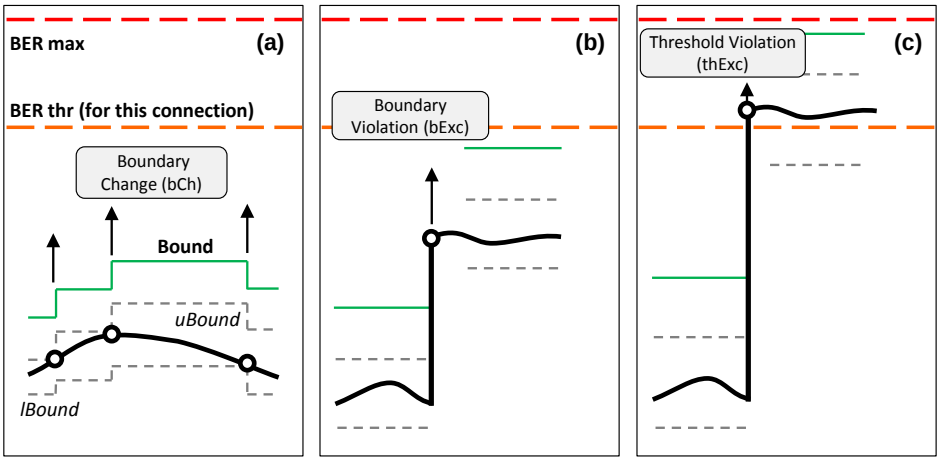


Fig. 7-4. BER and boundaries evolution with time

To follow BER evolution with time, an *outer boundary* is used to anticipate BER threshold violation and to detect sudden BER variations. In addition, two *inner boundaries*, named as a lower boundary (*lBound*) and upper boundary (*uBound*),

are used to trigger boundary re-estimation when measured BER reaches, exceeds or falls below one of them. Inner and outer boundaries are estimated as *bound* = $\mu(M.ber)$ +/- $k \cdot \sigma(M.ber)$, where $\mu(M.ber)$ and $\sigma(M.ber)$ are the mean and the standard deviation computed on the last n BER measures and k is a multiplicative factor different per each boundary.

Every time an event occurs, a notification is sent to the MDA system and analyzed by LUCIDA; defined events include:

- the boundary is re-estimated (bCh)
- the boundary is exceeded (bExc)
- BER exceeds the threshold (thExc)
- BER falls below the threshold (thDec)

Fig. 7-4(a) presents an example of monitored BER evolution with time causing boundary changes. As soon as monitored BER crosses one of the inner bounds, a boundary re-estimation is triggered, and a notification is sent toward the MDA system. Note that such boundary changes do not necessarily entail excessive BER, so the notification has an INFO severity level. Fig. 7-4(b) and Fig. 7-4(c) present two examples of sudden BER variation, where the bound and the BER threshold is exceeded, respectively. In such cases, boundaries are reset, and notifications are sent to the MDA system with WARNING and MAJOR severity levels, respectively. Note that, in case pre-FEC BER exceeds maximum BER, a notification will be sent with a CRITICAL severity level.

Table 7-2. BANDO Algorithm.

```
INPUT: M, ber, Bounds, state
OUTPUT: Event, state, Bounds
 1:  Event ← Ø
 2:  Update(M with ber,P_Rx)
 3:  if ber ≥ Bounds.th then
 4:    if state ≠ thExc then
 5:      Event = thExc
 6:    return <Event, thExc, ->
 7:  if state = thExc then
 8:    return <thDec, bExc, ->
 9:  if ber ≥ Bounds.outerBoundary then
10:    if state ≠ bExc then
11:      Event = bExc
12:    return <Event, bExc, ->
13:  if ber is in [Bounds.lBound, Bounds.uBound] then
14:    return < Ø, Normal, ->
15:  Bounds ← recomputeBounds(Bounds, M, ber)
16:  return <bCh, state, Bounds>
```

BANDO algorithm presented in Table 7-2 has been designed in order to obtain the current state and what *Event* led to it. The *state* is used to store whether BER

status is normal or has exceeded either the boundary or the threshold. Besides, the *event* stores notifications and actions.

When a new sample (*ber*, P_{Rx}) arises, a new event is created (line 1) and the tuple *M* is updated (line 2). Then, the value of BER of the new sample is compared against the different *Bounds* to classify which type of *Event* and new *state* (lines 3-14). Finally, in line 15, the boundaries are recomputed and line 16 returns the output of the algorithm <*Event*, *state*, *Bounds*>.

7.3.2 LUCIDA Algorithm

Regarding failure identification, we propose LUCIDA as a probabilistic algorithm that returns the most likely failure among a set *Q* of failure classes. Firstly, LUCIDA computes the probability of a set *H* of relevant features that can be observed on collected monitoring time-series. In view of the failures described in Section 2.2, three relevant features that can be identified and quantified in time-series are:

- P_{Rx} *above the reference level* (P_{RX}*high*)
- *BER positive trend* (*BERTrend*)
- *BER periodicity* (*BERPeriod*).

Next, LUCIDA maps feature probabilities to failure probabilities by means of predefined combination functions.

Upon the reception of a BANDO notification with a relevant BER change, i.e., either *bExc* or *thExc*, the algorithm in Table 7-3 is triggered.

Table 7-3. LUCIDA Algorithm

INPUT: *notif*, α, β_{qh}, δ
OUTPUT: <*class, prob, timeToMaxBER*>

```
 1: connId ← notif.conn_id
 2: lastBer ← notif.getLast().ber
 3: threshold ← notif.threshold
 4: storeNotif(connId, notif)
 5: if (lastBer / threshold) < δ then return <'no',-,->
 6: M.D={<t,ber,P_Rx>} ← getData(connId)
 7: M.N={<t,type,data>} ← getNotif(connId)
 8: P_H ← computeFeatureProbs(M, α) (Table 7-4)
 9: P_Q ← computeFailureProbs(P_H, β_qh)
10: failureClass ← arg max_(q ∈ Q)(P_Q)
11: tmax ← -
12: if P_H(BERTrend) > 0 then
13:   tmax ← computeTMax (M)
14: return <failureClass, P_Q(failureClass), tmax>
```

After retrieving useful data from the received notification, the notification is stored in a database for further analysis (lines 1-4 in Table 7-3). Then, the ratio between the last monitored BER value and the connection BER threshold is computed and compared to parameter δ. In case the ratio does not exceed δ, we assume that no failure is evinced (line 5); otherwise, failure detection is positive, and the identification procedure is started (lines 6-14).

Failure identification is based on processing historical BER and P_{Rx} time-series obtained from a monitoring DB, as well as historical notifications stored in the notification DB. The first step consists in retrieving P_{Rx} and notification time-series that are stored in the local structure M and computing feature probabilities from data (lines 6-8). Once all feature probabilities have been computed, failure probabilities are evaluated (line 9). For each of the failures q, a score is computed by means of the product of feature probabilities (eq. (7-1)), where β_{qh} coefficients are defined in the interval [0,1].

$$S(q) = \prod_{h \in H} \left[\beta_{qh} \cdot P_H(h) + \left(1 - \beta_{qh}\right) \cdot \left(1 - P_H(h)\right) \right] \tag{7-1}$$

For example, if β_{qh}=1 the partial score of feature h equals $P_H(h)$; if $P_H(h)$=0, the partial score will be 0, thus discarding the evidence of failure q. Finally, to obtain a failure probability in the range [0,1], the score of every failure is normalized eq. (7-2).

$$P_Q(q) = S(q) \Big/ \sum_{q \in Q} S(q) \tag{7-2}$$

Next, the failure class with the maximum probability is retrieved (line 10). Additionally, if the probability of feature BERTrend is non-zero, the time when maximum BER would be reached, *tmax,* is estimated by means of linear extrapolation computed from monitoring data (lines 11-13).

Table 7-4 details the algorithm to compute feature probabilities. Input time-series are firstly split into two segments: *i*) the *stationary* segment (D_S) containing the oldest samples which average and standard deviation remain near to a constant value, and *ii*) the *non-stationary* segment (D_{NS}) that contains the most recent samples where meaningful changes of mean and/or standard deviation with respect to the stationary segment are observed (line 1 in Table 7-4). The rationale behind this division is based on the assumption that monitored signals behave stationary in time under normal conditions and that stationary behavior is severely altered in the event of a failure.

Table 7-4. ComputeFeatureProbs Algorithm

INPUT: M, α
OUTPUT: P_H

```
1:  <D_s, D_ns> ← splitSegments(M.D)
2:  F ← N(μ(D_s.P_Rx), σ(D_s.P_Rx))
3:  x ← M.N.P_Rx.getLast()
4:  P_H[P_RXhigh] = p(x) ← computeTruncatedProb(F, x, α)
5:  P_H[BERTrend]←0; P_H[BERPeriod] ← 0
6:  D_notif ← extractDataSeries(M.N)
7:  for D in {D_ns.BER, D_notif.BER} do
8:     max_model ← linearRegression(D,'max')
9:     x ← max_model.slope.mean
10:    F ← N(0, max_model.slope.std)
11:    p(x) ← computeTruncatedProb(F,x,α)
12:    P_H[BERTrend] ← max(P_H[BERTrend], p(x))
13:    min_model ← linearRegression(D,'min')
14:    D' ← normalize(D, min_model, max_model)
15:    SP={<period, density>} ← spectrogram(D')
16:    SPhigh← {sp∈SP | sp.density≥ mean(SP.density)}
17:    x ← 1-(|SPhigh|/|SP|)/0.5
18:    F ← N(0, (|SPhigh|)^-1/2)
19:    p(x) ← computeTruncatedProb(F,x,α)
20:    P_H[BERPeriod] ← max(P_H[BERPeriod], p(x))
21: return P_H
```

To compute feature probabilities, we obtain the probability distribution function F that returns high probabilities when the evidence of the desired feature is significant. To give emphasis to significantly high feature values (x), we use the truncated probability distribution defined in (eq. (7-3)), where $F(x)$ is the cumulated probability when the feature takes the value x, $\alpha \in [0,1)$ is the minimum allowed probability in F, and $F^{-1}(\alpha)$ is the inverse of the distribution function and returns the value with a cumulative probability equal to α.

$$p(x) = \begin{cases} 0, & x < F^{-1}(\alpha) \\ \dfrac{F(x) - \alpha}{1 - \alpha}, & x \geq F^{-1}(\alpha) \end{cases} \quad (7\text{-}3)$$

The probability of feature $P_{RX}high$ is computed by characterizing the probability distribution of P_{Rx} in the stationary segment, i.e., the P_{Rx} reference level (lines 2-4). Without loss of generality, we assume a Gaussian distribution function defined by the mean and standard deviation of the samples in that stationary segment ($\mathcal{N}(\mu,\sigma)$).

In the case of features related to BER, the non-stationary time-series segment D_{ns} is used. Since D_{ns} time-series could be noisy, we consider another time-series D_{notif}, created from the notifications M.N time-series that could reflect more clearly the desired features of trend and periodicity; D_{notif} data is completed with intermediate

data points computed by linear interpolation (line 6). In the algorithm, we compute BER-related feature probabilities in both time-series and return the highest probability for each feature (lines 7-21).

For the *BERTrend* feature, the linear model that represents the evolution of the maximum BER with time is found; time-series are split into several chunks, and the model is obtained by applying linear regression to the pairs <time, maximum value>. Note that this model collects trend independently of whether the time-series has a meaningful period or not (line 8). The mean and the standard deviation of the slope of the model allow evaluating whether that mean slope is significantly higher than 0 (lines 9-10). Finally, the feature probability is computed (eq. 5.3) and P_H is updated (lines 11-12).

As for the *BERPeriod* feature, we compute a linear model for the evolution of the minimum BER that it is used, together with that for the maximum, to normalize the selected time-series D as specified in (eq. (7-4)) (lines 13-14).

$$D'(t) = \frac{D(t)\text{-}min\_model(t)}{max\_model(t)\text{-}min\_model(t)} \tag{7-4}$$

Next, the spectrogram of D' is computed to obtain the density value for every possible period interval (line 15) [Pr91]. To detect periodicity, we look for periods with densities clearly higher than the majority of the densities; hence, we find the set of periods with a density over the mean and its proportion over the total of periods is compared to the expected proportion in case of no periodicity, i.e., 0.5 (lines 16-17). Since x tends to be 0 when no meaningful period is observed, we use a Gaussian distribution centered in 0 and with a standard deviation inversely proportional to the number of periods over the mean (line 18). Feature probability is eventually computed and P_H updated (lines 19-20).

As a final remark, it is worth noting that the accuracy to detect and identify failures is subject to various factors, including the configuration of BANDO and LUCIDA parameters. The next section presents illustrative results to find the configuration leading to faster and more accurate detection and identification of failures.

7.4 **Results**

In this section, we first present the experimental setup needed to evaluate the performance of the proposed algorithms for early pre-FEC BER degradation detection and failure identification and then, illustrative simulation results are presented.

7.4.1 Experimental Measurements for BER and P_{RX}.

In this subsection, we experimentally reproduce the soft failures presented in Section 2.2 aiming at measuring BER and P_{Rx} data that will be used to generate synthetic data for the simulations in the next subsection. A comprehensive set of measurements was carried out in a testbed deployed in CNIT premises employing two types of 100Gb/s transmission systems and different configurations of traversed filters, channel spacing, and optical spans.

In the first setup used for the measurements, the considered 100Gb/s transmission system is based on Polarization-Multiplexed Quadrature Phase-Shift Keying (PM-QPSK) and coherent detection derived from the lab implementation utilized in [Cu13]. Two modulated lasers (signal 1 and signal 2) are multiplexed by means of a WSS configured to reserve a 37.5 GHz frequency slot for each channel. Measurements are reported for signal 1.

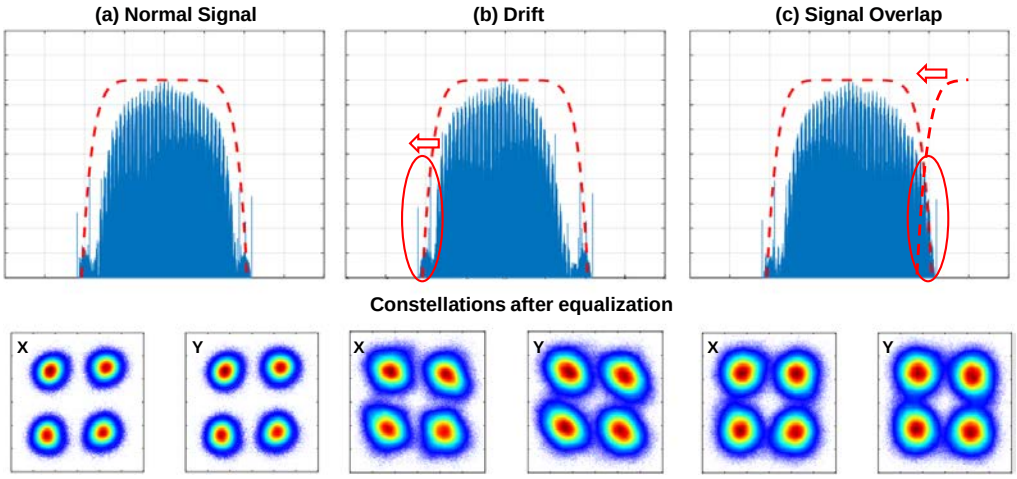


Fig. 7-5. Experimental results for the (a) normal conditions and (b), (c) considered failures.

In a first experiment, drift effects are applied by inducing frequency drift to signal 1. In a second experiment, signal overlap is introduced by applying laser drift to signal 2. In this second case, the channel spacing among the signals decreases, inducing an increase of interference. The spectrum related to signal 1 is reported in Fig. 7-5 for both experiments. Fig. 7-5(a) shows signal 1 spectrum under normal conditions. Fig. 7-5(b) reports on the first experiment, showing the slight shift in frequency due to the laser drift and Fig. 7-5(c) reports on the second experiment, showing that part of signal 2 falls within the bandwidth of signal 1.

Additional experiments have been performed on a second setup exploiting, as signal 1, a commercial 100Gb/s transmission system based on PM-QPSK and coherent detection. In this second setup, four 80km-spans are also introduced to assess the system performance under different conditions of OSNR.

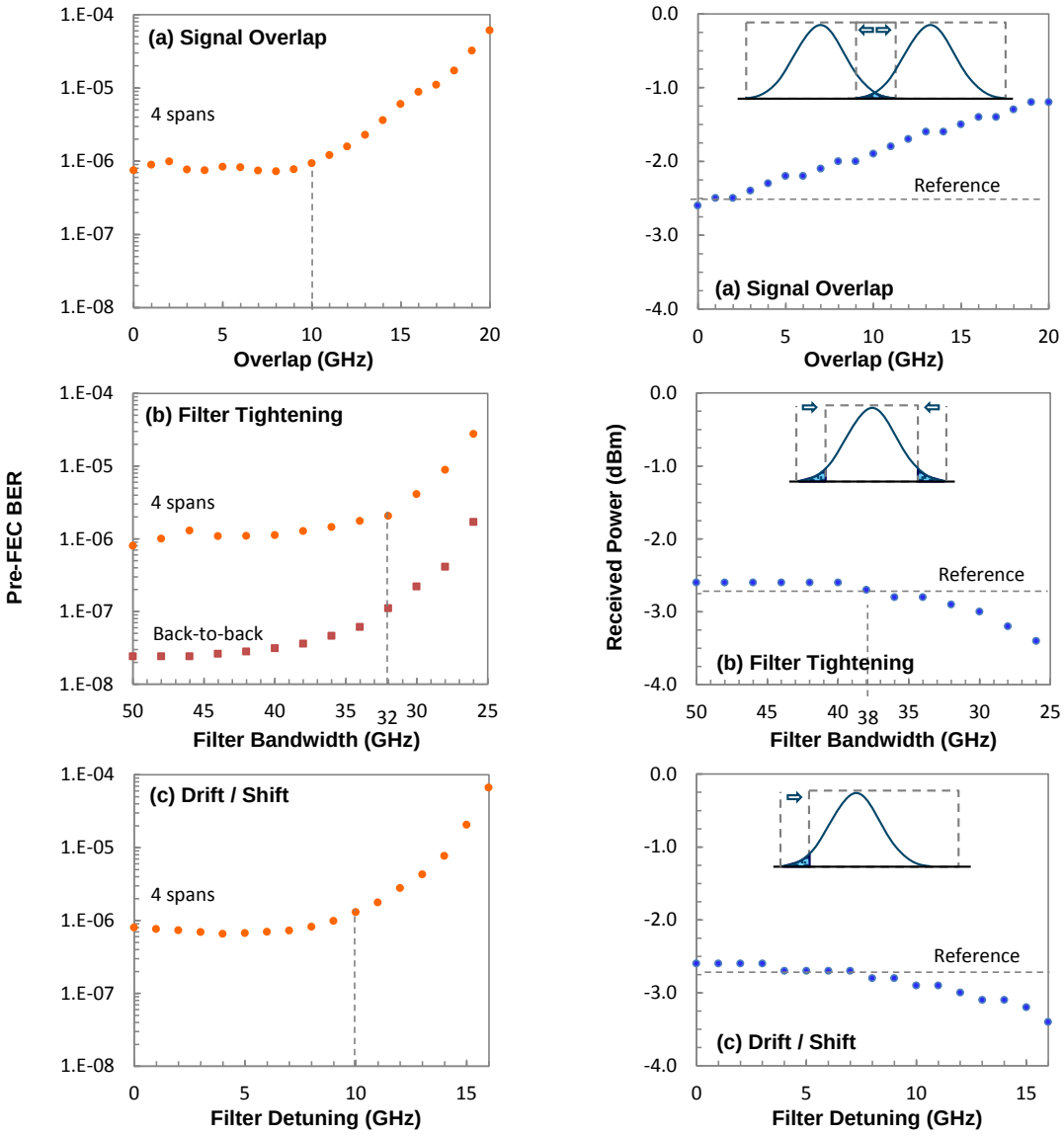


Fig. 7-6. Experimental BER and P_{Rx} for: (a) signal overlap, (b) filter tightening, and (c) drift/shift.

Fig. 7-6 reports pre-FEC BER and P_{Rx} provided by the commercial 100Gb/s system. Plots in Fig. 7-6(a) show the measured values in the case of signal overlap. In particular, the 100Gb/s signal used in the first setup is now utilized to induce overlap (on?) the commercial 100Gb/s signal (x-axis in Fig. 7-6(a) reports such overlap). Results show that the pre-FEC BER starts to increase when channel overlapping goes above 10GHz, while received power starts increasing for small channel overlapping values since part of signal 2 enters in signal 1 bandwidth.

In the case of filter tightening (Fig. 7-6(b)) x-axis reports the actual bandwidth configured on the traversed Bandwidth Variable Wavelength Selective Switch (BV-WSS). The central frequency of the signal has been aligned with the center of the

filter, i.e., both sides of the signal are equally affected when filter tightening is applied. For comparison, Fig. 7-6(b) also reports the measurements for the first experimental setup (back-to-back configuration). Results show that up to 32GHz can be configured without significant penalties, whilst further reduction of the actual frequency slot drives signal degradations. Note that post-FEC performance is error free in all the reported plots. The minimum supported filter configuration is 26GHz (lower values would tear down the connection since post-FEC error free condition can be no longer guaranteed). Regarding received power, results show a clear deviation from the reference value, starting from a frequency slot of 38GHz.

A similar behavior was observed in the case of drift (Fig. 7-6(c)), where *x*-axis reports filter detuning. Pre-FEC BER increases when filtering effects become more relevant because of filter detuning. Results show that the pre-FEC BER starts increasing when filter detuning goes above 10GHz. Obviously, received power decreases when part of the power is cut by the filter; a clear deviation from the reference value is shown for the received power when filter detuning goes above 10GHz. Another case of drift is that of the laser. 48-hours monitoring was performed with the bandwidth set to 30GHz and, because laser drift of the commercial card, pre-FEC BER was observed as if the bandwidth was 28GHz.

As a conclusion, although the behavior of the pre-FEC BER looks similar for all three failure cases, that of the received power is different. Indeed, the proposed LUCIDA algorithm is based on the identification of such different behaviors to discern between failures.

7.4.2 Simulation Scenario and Parameter Tuning

According to the experimental measurements in the previous section, we generated synthetic monitoring time-series at a rate of one sample per minute by means of a generator implemented in R. Each monitoring sample includes a synthetic measure of pre-FEC BER and P_{Rx}. The generator allows reproducing realistic monitoring activity of a set of optical connections with different characteristics, such as route, spectrum allocation, and slot width. Based on such characteristics and those of the underlying optical network topology, signal behavior in the absence of failures is generated. Besides, a per-connection BER threshold is computed based on an estimated BER value computed as a function of the OSNR of the links in its route [Sa11].

The generator allows reproducing any of the soft failures introduced in Section 2.2. According to the selected failure, one or more connections become affected at a given time, when some of their relevant physical properties are altered, e.g., filter bandwidth is narrowed; in the case of gradual changes, the magnitude of the alteration increases linearly with time following a predefined rate. Varying optical connection properties, failure class, failure magnitude, and gradual variation rate, we generated more than 100 distinct configurations. For each configuration, five

60-day instances (each generating 86,400 monitoring samples per optical connection) were randomly generated. Some of these configurations produced instances where BER never exceeded connection's BER threshold (we call this as the *lowBER* set), whereas the rest contain at least one monitoring sample exceeding the connection's BER threshold (we call this as the *highBER* set).

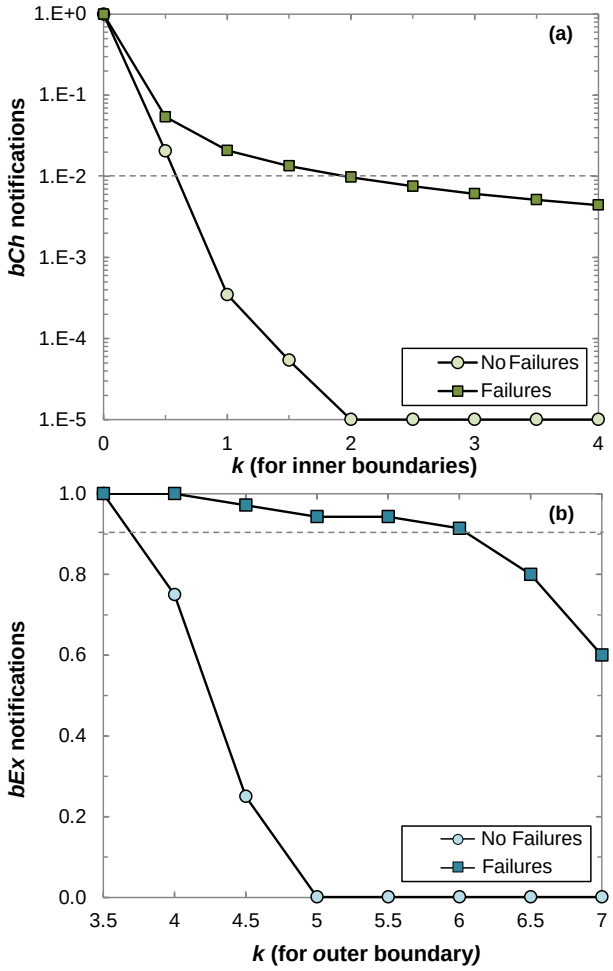


Fig. 7-7. Tuning of BANDO parameters.

Both BANDO and LUCIDA algorithms were implemented in R and integrated into a simulator following the architecture presented in Fig. 7-3. Aiming at finding the best configuration for BANDO parameters (to avoid an excessive number of notifications being sent to the controller while keeping it informed of meaningful BER changes), we set n=15 and perform several tests with a wide range of k values for inner and outer boundaries; results are reported in Fig. 7-7 were values are normalized to those for the minimum k.

Starting with inner boundaries, Fig. 7-7(a) shows a number of *bCh* notifications for different values of k for connections affected by a failure and for those normal. Hence, configuring k equal to 3 allows keeping boundaries constant when normal

BER behavior is monitored. In the event of connections with failure, less than 1% of all monitoring samples generate a *bCh* notification, which is enough to keep track of BER evolution with time as it will be shown in the following results.

Regarding the outer boundary, Fig. 7-7(b) shows the amount of *bExc* notifications as a function of *k*. Fixing *k* equal to 6 eliminates those notifications caused by atypical BER measures that do not entail failures, as well as keeps more than 90% of those notifications raised in the event of a failure. It is worth noting that *bExc* notifications are much less frequent than *bCh* ones and consequently, its impact on total notification overhead is negligible.

7.4.3 Degradation Detection and Failure Identification

Once BANDO has been properly configured, simulations including failure detection and identification were run. We configured LUCIDA parameters α=0.7 and β_{qh} = 1 if failure *q* must present evidence of feature *h* (β_{qh} = 0, otherwise).

In the simulations, LUCIDA was triggered in two distinct modes: only upon the reception of a *thExc* notification (*Major* mode) and upon the reception of any notification (*Info* mode). It is worth noting that only the *Info* mode allows detecting failures in the lowBER set, which confirms the need of BANDO and LUCIDA collaboration.

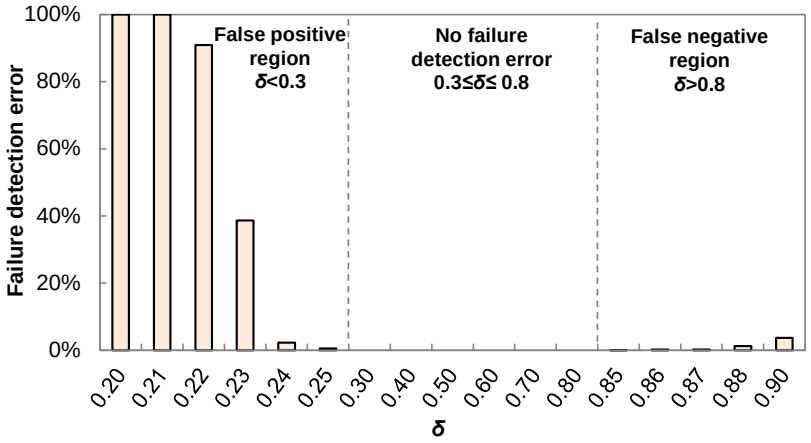


Fig. 7-8. Failure detection errors

For the *lowBER* instance set and the *Info* triggering mode, Fig. 7-8 analyzes δ parameter tuning, where the percentage of decision errors is plot as a function of its value. Since BER threshold is set as 5 x estimated BER, we assume δ=0.2 as starting point. When δ<0.3, some normal optical connections cross the failure detection condition and are classified as one of the failure classes thus, producing a false positive detection. On the other end, δ>0.8 produces that some actual failures never reach the detection limit and hence, they are wrongly classified as normal

(false negatives). In the middle, failure detection has no error and hence, we assume δ=0.5 for the ongoing results.

Let us now focus on the identification of the detected failures. Table 7-5 details the identification error upon the reception of the first triggering notification. Note that no identification error is observed for signal overlap and filter tightening failures, which is a good result since these failures generate very few notifications and need to be identified as soon as they are detected.

Table 7-5. Failure Identification Errors (First Notification)

Failure	**highBER**		**lowBER**	
	Major	***Info***	***Major***	***Info***
Signal Overlap	0%	0%	-	0%
Filter Tightening	0%	0%	-	0%
Gradual Drift / Shift	33%	37%	-	30%
Cyclic Drift / Shift	70%	48%	-	54%

As anticipated above, the *Info* mode allows LUCIDA to detect all signal overlap and filter tightening failures, even when they do not produce BER samples over the threshold, which enables detecting soft failures hidden below a too high threshold.

In the case of gradual and cyclic drift failures, the first identification is not correct in most of the cases since they are related to BER trend and periodicity features, and time is needed to ensure their presence or absence. However, both failures produce many and various notifications compared to signal overlap and filter tightening ones, and therefore, the opportunity of identifying the failure extends beyond time.

In view of the above, we study the time needed for a right failure classification of gradual and cyclic drift failures. Plots in Fig. 7-9(a) for the *Info* mode and Fig. 7-9(b) for the *Major* mode represent the evolution of the computed failure probability of a cyclic drift failure as a function of the number of periods since the first bCh event. Note that markers represent only those notifications that actually triggered failure identification phase, that is when the ratio between the last monitored BER and the threshold exceeds δ. In both modes, the most probable failure identified when a triggering notification is received before the first periodical peak is gradual drift since during the raising front LUCIDA detected a meaningful trend.

In contrast, the probability of the cyclic drift failure class is negligible since no periodicity was found. However, when a complete period is observable, BER periodicity feature starts being significant and cyclic drift becomes the most probable failure class from that point on. The difference between both modes is the time for a right failure classification; because under the *Info* mode LUCIDA

receives several notifications as a result of different events detected by BANDO, it allows a clearer identification of the non-stationary time-series segment, and therefore, it is able to produce right classifications after one single period, i.e., less than half of time compared to the *Major* mode. Although illustrated in Fig. 7-9 for just an instance, this gain keeps constant for all other cyclic drift instances.

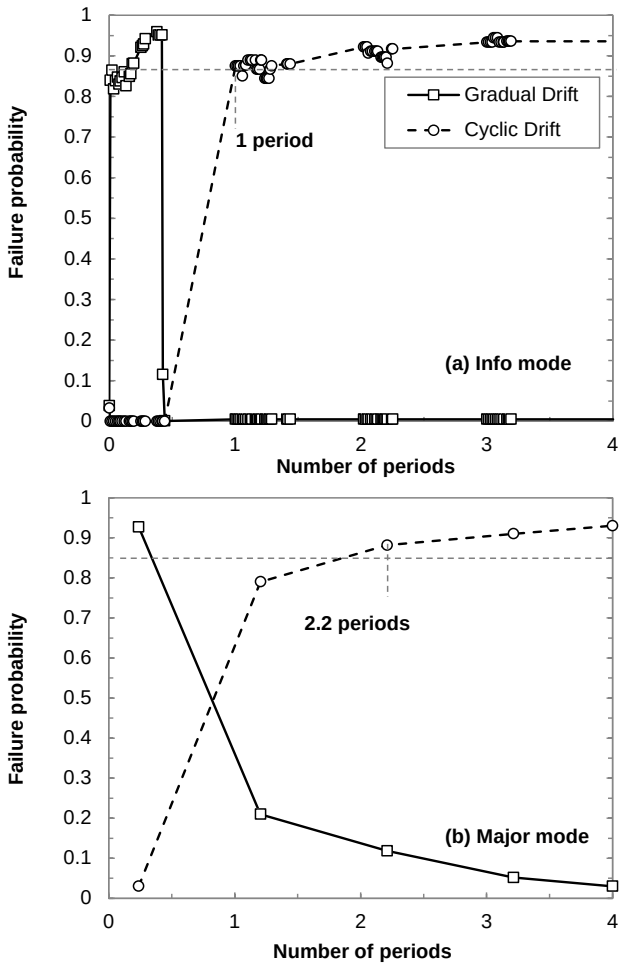


Fig. 7-9. Cyclic Drift Identification.

Finally, Fig. 7-10 illustrates the accuracy of the estimation for the time when max BER ($1 \cdot 10^{-6}$) will be reached in case of a gradual drift failure. Prediction based on linear extrapolation is shown at three different time instants. Although the failure is perfectly identified as gradual drift upon the reception of a *bExc* at day 30 (Fig. 7-10(a)), due to the lack of evidence of the actual future BER trend evolution, no max BER violation in the following 30 days is predicted. Later, upon the reception of a *thExc* at day 36 (Fig. 7-10(b)), max BER violation is predicted to happen in the near future. It is not until day 42, i.e., five days before the connection is disrupted, that prediction becomes steady to a constant value, which happens in Fig. 7-10(c); hence, this method provides enough anticipation for an optimal reaction against

the failure. Comparable results were obtained for the rest of gradual drift instances.

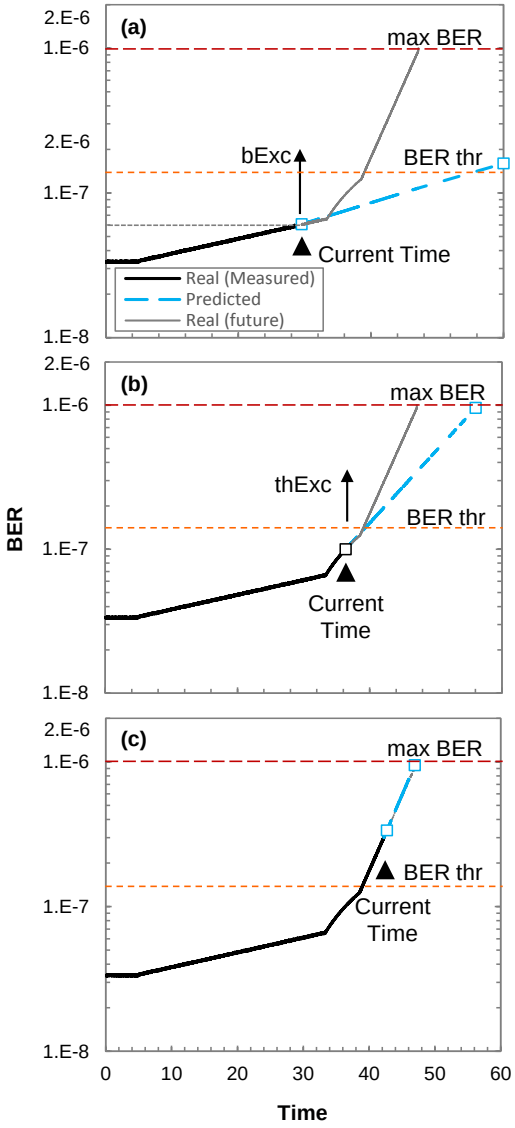


Fig. 7-10. Max BER anticipation.

7.5 Conclusions

SLA violations entail money losses for the network operators and hence, minimizing such violations is of paramount importance to them. This chapter focused on anticipating BER degradation detection at the optical layer, which typically supports many of the offered services. In addition to a prompt BER degradation detection, the chapter targeted at failure identification to help to localize the cause of the failure.

In this regard, two cooperating algorithms have been proposed: *i*) the BANDO algorithm which works inside the optical nodes to take advantage of a fine monitoring granularity, and *ii*) the LUCIDA algorithm, working in the centralized MDA system. BANDO detects changes in the BER of optical connections and sends notifications to LUCIDA.

To evaluate the performance of the algorithms, different BER degradation failures were considered, including gradual and periodical degradation. Aiming at studying realistic scenarios, experimental measures were carried out on two different setups involving commercial equipment. The results of the experiments were used to generate synthetic data used to simulate the considered BER degradation failures.

Simulation results show that maximum BER violation was anticipated several days before the connection was disrupted, which allows planning a network reconfiguration to be performed on low activity hours. Interestingly, the cooperation of BANDO and LUCIDA algorithms demonstrated its advantage for failure identification compared to a centralized algorithm receiving notifications only after BER threshold violations.

The next chapter goes further by focusing on the location of the failure after BER degradation is detected. We take advantage of BER degradation detection and failure identification/localization to trigger network re-configuration, so as to avoid SLA breaking.

Chapter 8

Network Reconfiguration after Soft Failure Detection

This chapter focuses on triggering a re-routing process after failure detection and identification has been achieved, targeting at reducing SLA violation. First an effective machine learning-based algorithm is proposed, to localize and identify the most probable cause of failure impacting a given service. Once the failure has been detected, identified and in some cases localized, reconfiguration can be performed for one particular case. Specifically, the SCULPTOR algorithm is defined for demand re-routing, triggered by BANDO notifications. Results show that the proposed identification and re-routing algorithms noticeably reduce bandwidth and the number of demands affected.

8.1 Introduction and Notation

Service layer connections are usually set up on top of VNTs, where vlinks are supported by lightpaths in the optical layer. Thus, errors in lightpaths translate into errors in those connections that might cause packet losses and retransmissions and lead to unacceptable QoS. For this very reason, a gradual degradation in the optical layer could impact a large number of client demands. To keep committed QoS, monitoring the physical layer is key to verify the fulfilment of SLA and, in case of faults or degradations, to localize the failed elements [Ta15] [Da16] and to take actions for preserving the services. However, the degradation might affect differently each client demand; specifically, those demands related to a SLA need especial attention since a SLA violation represents money losses for the network operator [Zh16].

In this chapter, we continue studying the effects on QoT monitoring parameters of several failures on the optical layer, specifically those of *filter tightening* and *signal overlap interference*; collected QoT monitoring parameters include received power

(P_{Rx}) and pre-FEC BER. After failure detection, re-routing is performed in certain conditions. Specifically, the contribution of this chapter is three-fold:

- Section 8.2 proposes an algorithm that analyzes monitoring time-series and, based on the expected patterns obtained in our experiments for the considered failure causes, localizes and identifies the most probable cause of failure at the optical layer affecting a given service.
- Section 8.3 presents a re-routing algorithm named SCULPTOR that re-routes the affected demands upon the reception of BER degradation notifications for the sake of SLA fulfillment.
- The discussion is supported by the results in Section 8.4.

Some key notation used in this chapter is presented in Table 8-1.

Table 8-1. Relevant notation used in this chapter.

s_d	source node for the demand
t_d	target node for the demand
b_d	required bandwidth for the demand
d_p	current path serving the demand
q_d	committed QoS in case the demand is related to a SLA contract
e	virtual link

8.2 Multilayer Failure Localization

This section focuses on analyzing the degradation caused by signal overlap interference and filter tightening. For illustrative purposes, recall that Fig. 7-2 presented monitoring data series for Pre-FEC BER and P_{Rx} magnitudes, for the possible causes of failure affecting a given optical connection (e.g., SO, FT, FS, LD, cFS, cLD).

As presented in Chapter 7, in case of LD, P_{Rx} decreases because of the filtered power and BER is degraded; in fact periods with degraded BER are followed by others with normal BER, which makes difficult to localize the failure cause. In fact, BER degradation is not always caused by P_{Rx} decrease, as shown in Fig. 7-2, the signal overlap example, where the allocation of a neighboring lightpath results in a sudden increment observed in the target lightpath. Hence, failure localization entails deep analysis of monitoring data from several lightpaths.

With the above in mind, we propose a probabilistic failure localization algorithm based on BN [Bi06]. A BN is a directed acyclic graph where nodes represent features and edges the conditional dependency between a pair of features (see

Chapter 2). Each node is associated to a probability function that takes values from the parent nodes and returns the probability of the feature represented by the node.

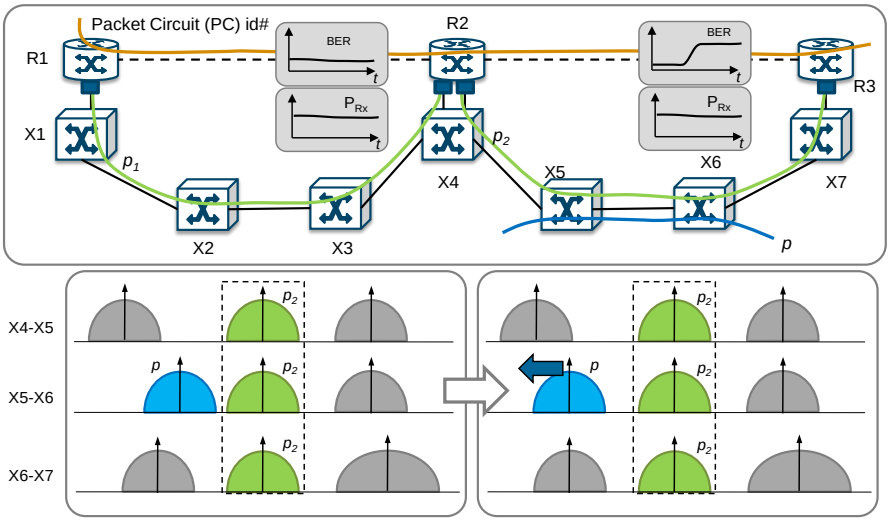


Fig. 8-1. Example of failure localization caused by signal overlap interference.

Fig. 8-1 shows an example of the proposed failure localization algorithm. Let us imagine that a service using a connection between *R1* and *R3* has detected and notified service degradation to the service provider. The monitoring data of the two lightpaths supporting the service connection are analyzed. As observed, *p1* monitoring data series show an almost constant trend for both power and BER, whereas *p2* BER suffered a steeped increase at some point in the past. From this available data, the failure localization algorithm returns no failure with probability 95% for *p1* and identifies interference with 70% and filter tightening with 25% for *p2*.

According to the probabilities above, the scope of network reconfiguration is firstly focused on *p2* and those lightpaths sharing an optical link in the route of *p2*. A deeper analysis identifies that *p2* BER increment is correlated with lightpath *p* set-up. Therefore, by slightly shifting *p* in the spectrum away from *p2*, its BER should be improved and the detected service degradation eventually reduced. However, a monitoring period after reconfiguration is needed to verify that BER degradation has been completely solved. In the case that *p2* BER has not reduced to normal values after *p2* shifting, the second action in the list is taken, which might consists in making filters wider to overcome the probable filter tightening failure. With this second reconfiguration step, the service degradation should be finally solved.

The BN needs to be trained to locate different causes of failures and to return its probability. Before the training phase, several experimental tests for each of the possible causes of failure, as well as for the no failure case, need to be carried out to obtain monitoring data series similar to the ones in Fig. 7-2. Then, those data series are transformed into relevant *descriptive features* collecting their main

characteristics, such as minimum, maximum, average, trend stepped change presence and size, etc. Since BNs require categorical features (i.e., with a finite range of levels), continuous features can be easily discretized by applying a clustering algorithm to find the number and ranges of each of the levels. The type of failure is also added as response feature.

Table 8-2. Failure Localization Algorithm

```
INPUT: s, BN
OUT: A
1:  P←getLightpaths(s)
2:  A←∅
3:  for each p in P do
4:    H←getMonitorDataSeries(p)
5:    F←BN.computeFeatures(H)
6:    F'←BN.discretize(F)
7:    D←BN.predict(F')
8:    A.p←sortProblemList(D)
9:  return A
```

The proposed algorithm that integrates the BN is presented in Table 8-2, it receives as input the affected service connection *s* and the previously trained BN. After retrieving the set *P* of lightpaths supporting *s* from the operational database (line 1 in Table 8-2), every single lightpath is sequentially processed as follows: first, the available BER and P_{Rx} monitoring data series are retrieved from the monitoring repository in the form of variable-length time-series of continuous data (line 4). Then, continuous features are computed and transformed into categorical values (lines 5-6). The prediction *D* returns, for each of the failures, the probability that it actually occurs (line 7). Such probabilities allow sorting the list of failures for every single lightpath, which is returned (lines 8-9).

8.3 Meeting Committed QoS

In the previous section we focused on localizing failures on a multilayer network and identifying the most probable cause of failure after its detection.

In this section, we propose an algorithm to promptly detect distinct BER anomaly patterns with the objective of anticipating intolerable BER values. BANDO algorithm defined in Chapter 7 detects boundary and threshold crossing and generates a notification containing additionally information, e.g., a prediction for the expected time when the maximum BER threshold is going to be crossed.

After BANDO notifications (e.g., exceeding the outer boundary *Bound*) are received, we propose the SCULPTOR algorithm to be triggered for pro-actively re-routing those demands affected by QoS degradation. For the sake of clarity let us

consider an example of the proposed re-routing illustrated in Fig. 8-2, where three client demands are being served. Each demand is denoted by:

- a demand identifier d_j,
- the required bandwidth,
- whether the demand requires some QoS level.

In the example, demands have different bandwidth requirements (30Gb/s, 70Gb/s and 40Gb/s respectively) and they are all ending in the same router (*R3*) but with diverse origins. In this example, let us assume 100Gb/s vlinks.

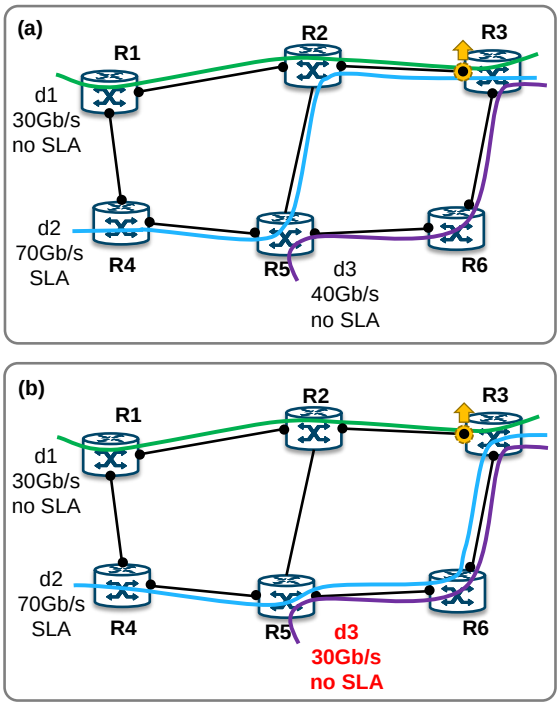


Fig. 8-2 (a) Initial demand routing before a service degradation. (b) After re-routing algorithm.

Fig. 8-2(a) shows the initial routing just when the BANDO algorithm has detected a BER degradation in the lightpath supporting vlink *R2-R3*, detected in *R3* endpoint. It is worth noting that since demand d_2 requires (high) QoS, when a BER degradation affects one of the links in its path, re-routing for this demand is mandatory to fulfill its SLA contract. On the contrary, both demands d_1 and d_3 do not require any particular QoS (best effort traffic), so when a BER degradation affects some link in their paths no re-routing is strictly required. When the degradation is detected, the SCULPTOR re-routing algorithm is triggered to find re-routing paths for those affected demands degradation and requiring QoS. SCULPTOR can also re-route no-QoS demands (affected or not by the BER degradation) with the objective of releasing resources that can be used to re-route affected demands with QoS requirements.

Fig. 8-2(b) shows the re-routing suggested by SCULPTOR. The initial path for SLA-related demand d_2 was affected by the BER degradation and it has been re-routed using an alternative path to avoid the BER degraded vlink. However, the available capacity in vlinks *R5-R6* and *R6-R3* would be exceeded unless the bandwidth of some of the demands being served through those vlinks are squeezed. The solution is to squeeze the bandwidth of not SLA-related demand d_2. Finally, demand d_1 does not require QoS and thus, it was not re-routed although its path contains the degraded *R2-R3* vlink.

The SCULPTOR reconfiguration problem can be stated as follows:

Given:

- A VNT represented by a graph G(N, E), where set N contains the MPLS nodes and the set E of vlinks.
- Set D of demands currently being served. Each demand d is characterized by the tuple: <s_d, t_d, b_d, d_p, q_d>, where s_d is the source node for the demand, t_d is the target node, b_d is the required bandwidth, d_p is the current path serving the demand, and q_d is the committed QoS in case the demand is related to a SLA contract.
- The monitored BER (in particular, that for which BANDO detected BER degradation) for every vlink $e \in E$.

Output: The demands to be re-routed and the new paths.

Objective: Minimize the amount of bandwidth entailing SLA violation, as well as the amount of unserved bandwidth (affected by bitrate squeezing).

The following parameters have been defined.

Demands and paths:

$P(d)$ Subset of pre-computed paths for demand d.

δ_{ep} Equal to 1 if path p uses link e.

q_p Equal to 1 if path p meets QoS requirements.

d_p Current path for demand d.

The decision variables are:

x_p Binary, equal to 1 if demand d uses path p; 0 otherwise.

y_{dp} Real, served bitrate for demand d through path p.

z_d Binary, equal to 1 if demand d is re-routed, 0 otherwise.

The SCULPTOR formulation is as follows:

$$\min \sum_{d \in D} \sum_{p \in P(d)} \left(q_d \cdot (1 - q_p) \cdot y_{dp} + (b_d - y_{dp}) + \alpha \cdot (1 - q_d) \cdot z_d \right) \quad (8\text{-}1)$$

subject to:

$$\sum_{p \in P(d)} x_p = 1 \quad \forall d \in D \tag{8-2}$$

$$x_p \geq d_p - (1 - q_p \cdot q_d) \quad \forall d \in D, p \in P(d) \tag{8-3}$$

$$x_p \leq d_p + q_p + (1 - q_d) \quad \forall d \in D, p \in P(d) \tag{8-4}$$

$$y_{dp} \leq x_p \cdot b_d \quad \forall d \in D, p \in P(d) \tag{8-5}$$

$$y_{dp} \geq q_d \cdot x_p \cdot b_d \quad \forall d \in D, p \in P(d) \tag{8-6}$$

$$\sum_{d \in D} \sum_{p \in P(d)} \delta_{pe} \cdot y_{dp} \leq b_e \quad \forall e \in E \tag{8-7}$$

$$\sum_{p \in P(d)} (1 - d_p) \cdot y_{dp} \leq b_d \cdot z_d \quad \forall d \in D \tag{8-8}$$

The objective function (8-1) minimizes the amount of bitrate that cannot be served (rejected, lost) with no QoS and minimizes bitrate affected by errors. Constraint (8-2) ensures a demand is routed by only one single lightpath (not multi-path routing is allowed). Constraint (8-3) ensures that if both the demand and the path have QoS requirements, the current path is not modified. For other cases, the path can be changed. Constraint (5-3) ensures that a demand with QoS requirements ($q_d=1$) should not be re-routed to a path with no QoS ($q_p=0$). Constraints (8-5) and (8-6) prevents squeezing the bitrate for demands with QoS requirements. Constraint (8-7) guarantees that the available bitrate b_e in each link is not exceeded. Finally, constraint (8-8) accounts for demands that are re-routed.

Regarding SCULPTOR constraints, it does not allow re-routing SLA-related demands if the current path already provides enough QoS and also to paths that do not ensure enough QoS.

To implement SCULPTOR, we first generate a set of possible paths for each demand and label them according to the fulfillment of the specific QoS requirements for that demand. As an illustrative example, let us assume that demand d2 in Fig. 8-2 requires QoS $\leq 10^{-7}$ and BER in R3 is 10^{-8} then the SLA agreement is currently met. However, in the case that BER in R3 would be 10^{-6}, the SLA would be violated and therefore the demand would be candidate to be re-routed through other path fulfilling the required QoS. Finally, note that the SCULPTOR problem always returns a feasible solution, i.e., the current one.

8.4 **Results**

In this section, we present two different cases where re-routing of the affected demands is mandatory to meet the committed QoS. We first present results regarding the performance of the BN targeting at failure localization and identification prior to re-routing. Likewise, notifications coming from BANDO's algorithm could also trigger re-routing. A study regarding the improvement achieved on the decrease of both, affected bandwidth and the affected demands, by using SCULPTOR algorithm is also presented.

Let us consider the experimental results from Chapter 7 where BER degradation failures were reproduced in order to generate synthetic data for the simulations.

According to the experimental values in Fig. 7-6, we generated synthetic monitoring time-series for the normal signal and the considered failure cases. A set of 5,000 randomly generated time-series were first used to train the BN and next, 500 additional ones used for testing. Table 8-3 reports the obtained goodness-of-fit computed as the probability that the BN predicts the actual failure cause as the first option. Note that only 0.8% error was observed in some tests where a normal signal was predicted instead of a filter tightening. In such cases, the second most probable cause of failure was filter tightening failure. This demonstrates the validity of the proposed procedure and BN to localize and identify failures in the optical layer.

Table 8-3. BN Goodness-of-Fit

		Real		
		Normal	**Filtering**	**Signal Overlap**
Pre-diction	**Normal**	99.2%	0.8%	0%
	Filtering	0%	100%	0%
	Signal Overlap	0%	0%	100%

According to the experiments in Chapter 7, we generated synthetic monitoring BER data time-series (at a rate of one measure each 15 minutes) reproducing BER degradation, as well as normal BER time-series. Normal BER was set to 10^{-8} and for the filtering failure, we considered an incremental BER lasting from 2 to 10 days to reach intolerable BER = 10^{-6}. The minimum time between detection and intolerable BER value was 40 hours, more than enough to plan the application of the SCULPTOR algorithm.

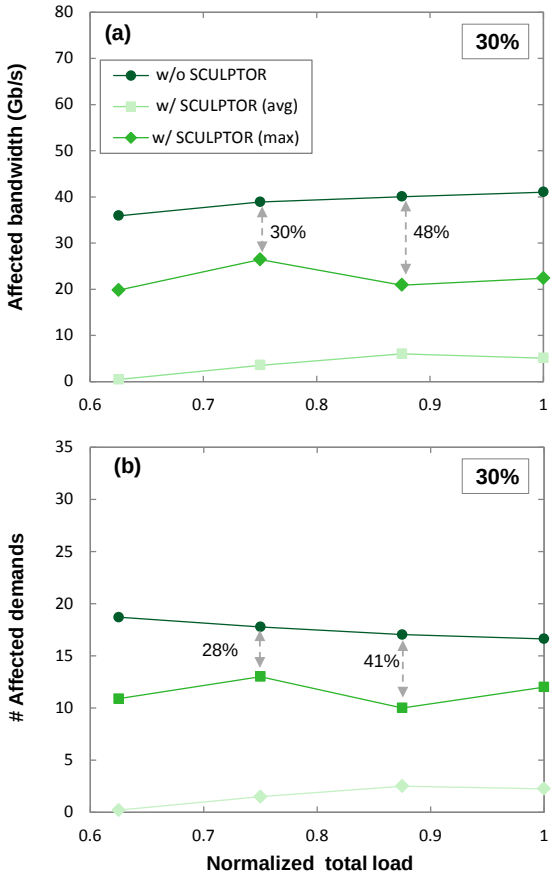


Fig. 8-3. Affected bandwidth and number of affected demands when 30% of the demands requiring QoS.

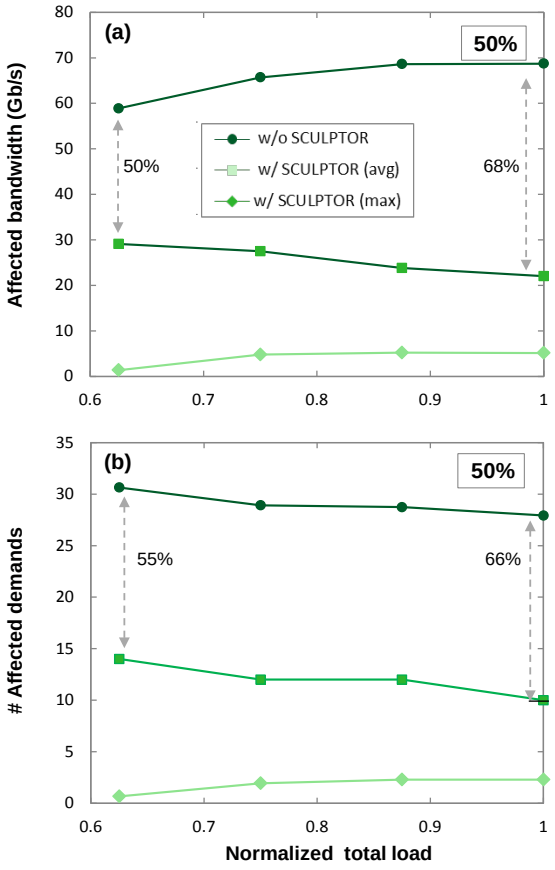


Fig. 8-4. Affected bandwidth and number of affected demands when 50% of the demands requiring QoS.

In order to evaluate SCULPTOR algorithm performance, we simulate a 30-node and 56-link MPLS network, where initially, 870 demands are being served. To evaluate the proposed algorithm, we compare the affected bandwidth and number of demands when SCULPTOR is applied or not. Fig. 8-3(a) and Fig. 8-3(b) present the gain obtained by using SCULPTOR in terms of affected bandwidth when 30% or 50% of the demands require QoS. Interestingly, the amount of affected bandwidth is reduced by at least 30% when SCULPTOR is applied. It is also worth noting that the affected bandwidth increases circa 60% when the amount of SLA-related demands increases from 30% to 50%, while the affected bandwidth remains mostly constant when SCULPTOR is applied.

The same study can be applied to the number of affected demands (Fig. 8-4(a) and Fig. 8-4(b)) obtaining similar conclusions as for affected bandwidth.

8.5 Conclusions

BER degradation in the optical layer could impact a large number of client demands, which can translate into money losses for network operators when violating SLAs. When a service detects excessive errors, a Bayesian Network – based algorithm is used to localize and identify the most probable cause of the errors at the optical layer. Results showed the effectiveness of the algorithm.

Targeting at anticipating intolerable BER, the BANDO algorithm was used for detecting, among others, gradual increasing BER degradation; its detection helps meeting the committed QoS by re-routing the affected SLA-related demands. To that end, the SCULPTOR algorithm was proposed.

Results showed that SCULPTOR noticeably reduces the number of affected bandwidth and demands.

In this chapter, two algorithms were proposed. First, a failure identification algorithm is run to classify between: *normal*, *filter* or *signal overlap*. Only for the *signal overlap* case, localization can be performed at the optical layer. However, location in the case of multilayer networks is easier, and allows to re-route demands as a function of the committed QoS. The next chapter focuses failure localization at the optical layer.

Chapter 9

Soft Failure Localization

Methods to detect BER degradation and failure identification methods were proposed in the previous chapters. Although soft failure detection and identification might allow to detect SLA violations while anticipating possible hard failure events; it is not enough to help operator and maintenance processes, where failure localization is strictly required. So far, failure localization techniques have been proposed and deployed mainly for hard failures, while significant work is still required to provide effective and automated solutions for soft failures, both during commissioning testing and in-operation phases. In this chapter, we focus on soft failure localization by proposing two techniques for active monitoring during commissioning testing and for passive in-operation monitoring. The techniques rely on specifically designed low-cost optical testing channel (OTC) modules and on the widespread deployment of cost-effective OSA.

The retrieved optical parameters are elaborated by machine learning-based algorithms running in the agent's node and in the centralized MDA system. In particular, the Testing optIcal Switching at connection SetUp timE (TISSUE) algorithm is proposed to localize soft failures during commissioning testing, whereas, the FailurE causE Localization for optIcal NetworkinG (FEELING) algorithm is proposed to localize failures during lightpath operation. Extensive simulation results are presented, showing the effectiveness of the algorithms to correctly localize soft failures.

9.1 Motivation and Objectives

Chapter 7 aimed at detecting in advance excessive BER in lightpaths in order to anticipate connection disruption. Once a BER degradation is detected, it is of paramount importance to localize the failure, so as to accelerate network maintenance and to reconfigure the network.

In this chapter, we focus on soft failure localization during commissioning testing, as well as once lightpaths are in operation. In the rest of the chapter, unless explicitly stated otherwise, we just use failure localization for the sake of brevity.

In this PhD thesis we propose to apply a similar concept to OSC (Chapter 4), for commissioning testing and failure localization purposes; we name it as optical testing channel (OTC). The main difference is that, in OTC, the low-speed low-index OOK modulation is applied to a continuous-wave laser rather than to a high-speed coherent signal. The modulation parameters in OTC are the same as in OSC to guarantee accurate BER estimation, while requiring simple and low-cost hardware for the operator.

We propose using OTC systems for active monitoring during commissioning testing, as well as the use of OSAs for passive monitoring. The techniques presented in this chapter highly depend on the modulation format of the lightpaths, so we restrict ourselves to focus specifically on QPSK-modulated signals since it is the most common modulation format used in medium and long reach telecom operator networks. Specifically, the contribution of this chapter is three-fold:

- Section 9.2 presents our proposals for BER estimation and failure localization. Besides, a node architecture equipped with OSAs and OTC modules is proposed and modules running in the agent's node and in the MDA system are presented.
- Section 9.3 focuses on designing the OTC system to be used during commissioning testing. The Testing optIcal Switching at connection SetUp timE (TISSUE) algorithm that received estimated BER and localizes failures is presented.
- Section 9.4 targets at localizing failures affecting a lightpath using OSAs. Optical spectrum features (see Chapter 2) are exploited by machine learning-based algorithms to detect degradations and identify failure classes. The FailurE causE Localization for optIcal NetworkinG (FEELING) algorithm running in the MDA system uses these modules to localize, classify and estimate the magnitude of the failure.

The discussion is supported by the results from simulation presented in Section 9.5.

9.2 Before and In-Operation Failure Localization

Two different scenarios for failure localization are considered:

- During lightpaths' commissioning testing to ensure the proper lightpath performance before they are delivered to the customer and enter into

operation. Note that since excessive BER might lead to SLA violations, BER needs to be checked at the reception side and, in the case of excessive BER, the source of the errors should be localized as accurately as possible. At this point, we assume that transponders are at the customer side and that the lightpath is already established in the network between ingress and egress nodes at the switching level, so active monitoring can be applied by injecting a *test signal*.

- Once a lightpath is in operation, BER can be measured, and BER degradations can be detected in advance before reaching excessive BER levels (see previous chapter). Once detected, the cause of failure needs to be localized, this time by using passive monitoring techniques, to facilitate lightpath re-routing.

For these scenarios, we propose the use of two monitoring systems to be installed in the optical network nodes: a redesigned OSC and OSAs (see Chapter 2). Here, the concept of OSC is redefined and renamed as OTC, where the OTC_{Tx} module is equipped with a tunable laser and a Pseudo-Random Bit Sequence (PRBS) generator to create a test signal. Then, the OTC_{Rx} receives the test signal and estimates the BER (see the details of this new OTC in Section 9.3).

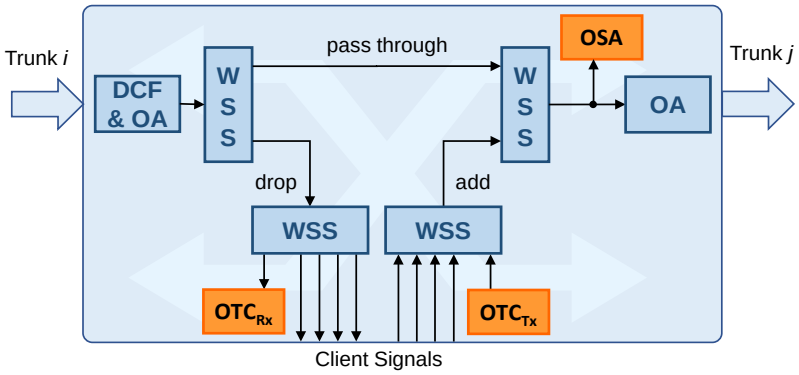


Fig. 9-1. Simplified optical node architecture with OTC and OSA monitoring.

Fig. 9-1 presents a very simplified diagram of the architecture of an optical node, where only one incoming and one outgoing links, as well as the local signals being added and dropped are represented. The node consists of WSSs, OAs, Dispersion Compensation Fibers (DCFs) and channel equalizers; OTC and OSA monitoring systems are highlighted. OTC modules are connected to local WSSs in the architecture in Fig. 9-1. In addition, only one single OTC_{Tx} and one single OTC_{Rx} modules per node need to be equipped, which although limits the number of concurrent test that can be carried out, also limits the number of consumed local WSS ports; this has a significant impact on the cost of the ROADMs [Kh15]. On the other hand, OSAs are placed in every outgoing link, so the number of OSAs per node equals the nodal degree. In this case, we have limited the number of OSAs due to its cost, and although failure localization can still be carried out, the

granularity of the localization would be at the node level. To achieve a finer failure location granularity, more OSAs should be placed, consequently increasing the node cost.

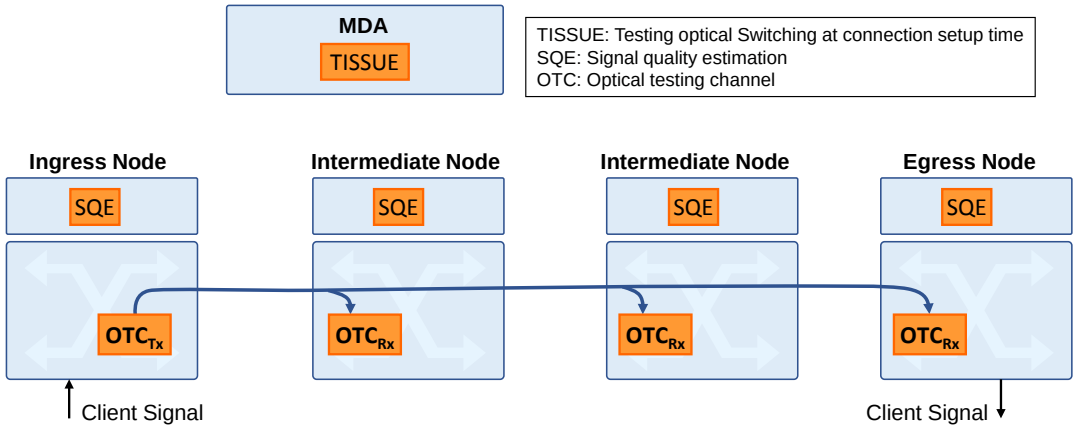


Fig. 9-2. OTC active monitoring for commissioning testing and failure localization.

Fig. 9-2 shows an example of the use of the proposed OTC monitoring system for before-operation tests and failure localization. One OTC_{Tx} is used in the ingress node to generate the test signal, and one OTC_{Rx} per intermediate and egress node is used to estimate the BER. Note that, since the lightpath has not been delivered to the customer yet, the client signal is not connected to the lightpath neither in the ingress nor the egress node at this stage. A module named as Signal Quality Estimation (SQE) running in the node's agent is in charge of receiving the measured BER in the local OTC and correlate to what the client signal would observe. The TISSUE algorithm, running in the MDA system, is in charge of allocating the OTC modules in the network nodes, setting-up the local connections from them to the lightpath in the end nodes, receiving BER estimations and deciding whether the tests pass or not, and estimating the elements that participate in the excessive BER.

Fig. 9-3 depicts the use of OSAs to localize soft failures once the lightpath is in operation. OSAs acquire the whole C-band spectrum, and then, data for the portion of the spectrum allocated to the lightpath under study is extracted. OSAs passive monitoring is carried out in the ingress and every intermediate node (but not in the egress one). The feature extraction module defined in Section 2.2, running in the node's agent is in charge of analyzing the spectrum.

The FEELING algorithm, runs in the MDA system and is in charge of commanding the modules in the nodes and of receiving a diagnosis, as well as the relevant signal points from them to localize the failure and estimate its magnitude. It is worth mentioning that FEELING must be able to distinguish between actual failures and normal effects that could lead to similar evidence, specifically filter tightening effects due to filter cascading of a normal signal. FEELING takes advantage of the

classifiers module that generates a diagnosis of one signal focusing specifically on filtering problems. In addition, these failure magnitude estimation modules (ME) have been designed to quantify specific failure effects:

- Laser Drift Estimator (LDE).
- Filter Shift Estimator (FSE).
- Filter Tightening Estimator (FTE).

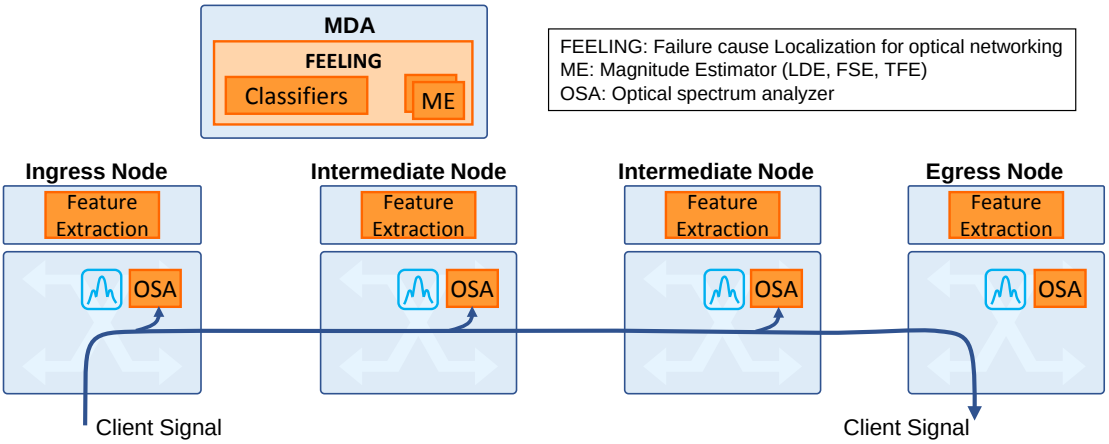


Fig. 9-3. OSA passive monitoring for in-operation failure localization.

The next section is focused on the design of the proposed active and passive monitoring systems.

9.3 Use Case I: Commissioning Tests and Failure Localization

Fig. 9-4 presents the first use case addressed in this chapter, which shows the OTC system design, where a continuous-wave laser is OOK modulated. A PRBS pattern generator drives the modulator (modulation speed is below 1 GHz with low modulation index). At different intermediate nodes, the OTC channel is dropped and received with a simple low-bandwidth photoreceiver connected to a BER tester.

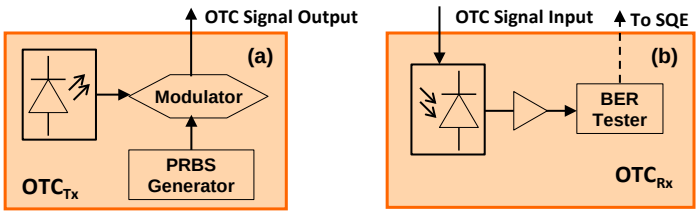


Fig. 9-4. OTC system design: a) OTC transmitter and b) OTC receiver

Note that, it is necessary for the operator to linearly adjust the modulation speed of the OTC channel according to the baud-rate of the lightpath requested by the client. In this chapter, we assume 25 GBd DP-QPSK client signals and OTC is 250Mb/s OOK. A BER conversion model (e.g., table or function) is needed to translate the OTC measured BER value into a client QPSK signal estimated BER.

As introduced above, the TISSUE algorithm running in the MDA system is in charge of collecting the QPSK signal estimated BER from each of the intermediate nodes. Running in the network nodes, the SQE module is in charge of acquiring the OTC BER and use the BER conversion model to obtain the estimated BER.

Table 9-1 TISSUE Algorithm

```
INPUT lightpath
OUTPUT FailureList
 1: <otc_Tx, OTC_Rx> ← allocateResources (lightpath)
 2: setupConnections (lightpath, {otc_Tx} ∪ OTC_Rx)
 3: for each r ∈ OTC_Rx do:
 4:   BER.estim[r] ← getEstimatedBER(r)
 5:   BER.theo[r] ← computeTheoBER(r.node, lightpath)
 6: failures ←∅
 7: for i = 1..|OTC_Rx|-1 do:
 8:   estimSlope ← compSlope(BER.estim[i],BER.estim[i+1])
 9:   theoSlope ← compSlope(BER.theo[i], BER.theo[i+1])
10:   if estimSlope / theoSlope > α then
11:     failures ←failures ∪ {<i, i+1>}
12: deAllocateResources (lightpath, otc_Tx, OTC_Rx)
13: return failures
```

Initially, the TISSUE algorithm (Table 9-1) allocates the OTC modules in the network nodes along the route of the lightpath and sets up the needed connections between the OTC modules and the lightpath, so the OTC_{TX} module injects the test signal in the ingress node and all the OTC_{RX} modules get the test signal to measure BER (lines 1-2 in Table 9-1). Next, the QPSK BER estimated values are collected from the SQE modules, and theoretical BER values are computed based on OSNR values [Sa11] (lines 3-5).

Finally, the difference between the slopes of both estimated and theoretical BER in each span are computed to determine the existence of a failure; if the slopes difference is above a maximum value, a failure has been detected in such span (lines 6-11). The OTC modules are released (line 12) and the list of spans in failure is eventually returned (line 13).

9.4 Use Case II: In-Operation Failure Localization

Let us now focus on the use of OSAs to localize failures once the lightpath is in operation. For the sake of clarity, Table 9-2 defines subset of features (see Chapter 2) that are considered in the classification module presented in this section, where ∂ stands for the derivative.

Table 9-2 Selected spectrum features

Module	bandwidth (*bw*)			symmetry (*sym*)			Freq. shift (Δfc)
	∂	-3dB	-6dB	∂-3dB	∂-6dB	slot-3dB	∂
Classifier	X	X	X	X	X	-	-
LDE	-	-	-	-	-	-	X
FSE	-	-	-	-	-	X	X
FTE	-	X	-	-	-	-	-

As described in Fig. 9-3, the classification module placed inside the MDA system generates a diagnosis of one signal focusing specifically on filtering problems; it classifies signals into three classes: *Normal*, *FilterShift*, *FilterTightening*.

In order to differentiate between the previous classes, we propose the algorithm presented in Table 9-3 that trains a SVM that will be used for classification. The algorithm receives a dataset that is firstly balanced by replicating samples for the less frequent classes, and it is then randomly split into training and testing (lines 1-2 in Table 9-3). After few initializations (lines 3-4), an iterative procedure is executed to fit a SVM, where the loop is iterated on both, the cost of misclassifying (*misClassCost*) and the degree of the polynomial kernel (*kernelDegree*), which are parameters to control the complexity and size of the SVM. For every *kernelDegree* (e.g., lineal, 2nd polynomial degree, etc.) a decision SVM is fitted from the training dataset and the error, defined as wrong classified samples over the total number of samples, is computed for both training and testing datasets (lines 5-9). In case of reducing the minimum error obtained so far, the difference between error from training and from testing is stored (lines 10-16). The SVM fitted with the input dataset is eventually returned (line 17).

The decision-making units of the classifiers module use the SVM previously fitted. The classifier module presented in Table 9-4, consists of a hierarchy of two binary classifiers based on SVM. For both classifiers, the algorithm in Table 9-3 is called with a particular *dataset* to obtain a particular SVM model. Initially, a failure class (line 1-2) is obtained using the corresponding SVM model (*SVMModel_NF*) to distinguish "*Normal*" from "*Filter problems*". For the case where the latter is

returned, lines 7-10 perform the same prediction using in this case the *SVMModel_FTFS* model, in order to classify between "*Filter Shift*" or "*Filter Tightening*".

Table 9-3 Classification Module: SVM Training Algorithm

```
INPUT dataset, kernelDegree, misClassCost
OUTPUT SVMModel
 1: dataset←balanceClassesByReplication(dataset)
 2: <training,testing>←randomSplit(dataset)
 3: minDiff←∞
 4: minError←∞
 5: for i in kernelDegree do
 6:   for c in misClassCost do
 7:     svm ← fitSVM(training, i)
 8:     errorTesting ←predict(SVM, testing)
 9:     errorTraining ←predict(SVM, training)
10:     if errorTesting<minError then
11:       minError←errorTesting
12:       minDiff←|errorTraining-errorTesting|
13:     else if errorTesting==minError then
14:       diff←|errorTraining-errorTesting|
15:       if diff≤minDiff then
16:         minDiff←diff
17: return SVMModel(dataset)
```

Table 9-4 Classifier Module

```
INPUT SVMModel_NF, SVMModel_FTFS, CapturedOptSpec, SelFeat
OUTPUT class
 1: decision← Ø
 2: decision←predict(SVMModel_NF, CapturedOptSpec, SelFeat)
 3: if decision ==0 then
 4:   class = "N"
 5: else if decision ==1 then
 6:   class = "F"
 7:   decision←predict(SVMModel_FTFS, CapturedOptSpec, SelFeat)
 8:      if decision ==1 then
 9:        class = "Filter Shift"
10:      else if decision ==0 then
11:        class = "Filter Tightening"
12: return class
```

Finally, the FEELING algorithm that uses the above-defined modules is detailed in Table 9-5; recall that FEELING is called upon the detection of excessive BER at the reception side of an optical signal. The algorithm first calls feature extraction module, in the ingress and last intermediate nodes to perform signal verification and obtain a diagnosis (lines 1-5 in Table 9-5). It is worth noting that *diagIngress* is a tuple <*class*, *X*, *features*>, where *X* is the captured optical spectrum. In the case that the diagnosis of both nodes is normal, FEELING ends with no failure detected (lines 6-7). Otherwise, in the event of laser drift diagnosis at the ingress, the LDE

module is run to measure failure magnitude (lines 8-10); for LDE modeling, we considered linear regression.

Table 9-5 FEELING Algorithm

```
INPUT lightpath
OUTPUT {<node, class, magnitude>}
 1: ingress ← lightpath.getNodeFromRoute (1)
 2: lastInterm ← lightpath.getNodeFromRoute (-2)
 3: FM ← getFilterMasks(lightpath)
 4: diagIngress ← getFailureDiagnosis (ingress, FM (1))
 5: diagLast ← getFailureDiagnosis (lastInterm, FM (-2))
 6: if diagIngress.class = diagLast.class AND
          diagIngress.class = Normal then
 7:    return {<1, Normal, ->}
 8: if diagIngress.class =LaserDrift then
 9:   magn← LDE(diagIngress.X)
10:   return <1, LaserDrift, magn>
11: XNodeChange ← diagIngress.X
12: diagChange = <class, magn> ←Classifier (diagIngress.X)
13: if diagChange.class<> normal then
14:   FailureSet←<1, diagChange>
15: else FailureSet ← Ø
16: for i=2..lightpath.RouteLength()-1 do
17:   node_i ← lightpath.getNodeFromRoute (i)
18:   Xi← getSignalPoints (node_i)
19:   diagNode_i ← Classifier (Xi)
20:   if diagNode_i.class <> diagNodeChange.class OR
        diagNode_i.magn - diagNodeChange.magn > α then
21:     XNodeChange ← Xi
22:     FailureSet ← FailureSet U {<i, diagNode>}
23: return FailureSet
```

In the case of a different diagnosis, FEELING starts a procedure to detect filter related problems at intermediate nodes using the classifiers module to compare diagnosis and magnitudes between nodes in the route of the lightpath. This process starts with the diagnosis at the ingress node that it is used as the initial reference node (lines 11-14). Then, the diagnosis of every intermediate node is compared against the one of the reference changing node and failure set is updated if either a new filter failure is detected or the magnitude increased above a certain threshold (lines 15-22). After processing all intermediate nodes, the list of failures detected is eventually returned (line 23).

When a filter related failure is detected, either a FSE or a FTE is called to estimate the magnitude of the failure as a function of the selected spectrum features (see Table 9-2). Linear regression for the magnitude estimators was used since both, magnitudes and features take real values. In order to find the proper set of features, we apply a stepwise approach that aims at finding the model with the

optimum balance between accuracy and number of coefficients (i.e., features) in terms of the AIC.

9.5 Results

This section reports the obtained results from simulating scenarios for the previous sections; Section 9.3 regarding commissioning testing and Section 9.4 related to in-operation failure localization.

9.5.1 Optical Testing Channel

Regarding commissioning testing, we first study the correlation between OTC measured BER and QPSK signal estimated BER. We performed simulations with a 250Mb/s OOK channel transmission with the OTC scheme described in Section 9.3, and a 25GBd DPQPSK signal to measure the BER after every span; Fig. 9-5(a) plots the obtained BER relation. It can be shown that there is an almost linear relationship between the BER of the OTC channel at 250Mb/s and the BER of a QPSK channel. Note that the above BER relation is specific for the particular case where the signal does not traverse any filter.

Because the signal will be affected by intermediate filters, different OTC vs. QPSK BER correlation curves need to be used as a function of the number of filters that the signal has traversed. Then, family of piece-wise linear models can be used to convert the measured OTC BER to the estimated QPSK signal BER as a function of the number of filters. Such models are stored in every SQE module and used every time the TISSUE algorithm requests BER estimation. Finally, after several tests, we set TISSUES's parameter α to 2.

At lightpath commissioning testing, the TISSUE algorithm requests SQE modules along the route of the lightpath to obtain BER estimations and compares them against theoretically computed values. Fig. 9-5(b) plots an example of theoretical and estimated BER for the last seven 100km spans of the simulated 10-span scenario (the first three spans are not shown since their BER is lower than 10^{-7}). As observed, values are very close (about half decade difference in BER values), proving that the OTC scheme is an effective testing technique for operators to check the quality of a new lightpath, as well as to localize spans with excessive BER.

Finally, to evaluate the TISSUE algorithm, we added 2dB of noise after span #5. Fig. 9-5(c) plots the estimated QPSK BER and the theoretical BER for the last seven spans. TISSUE localizes the failure after noticing the large estimated BER slope compared to the theoretical one.

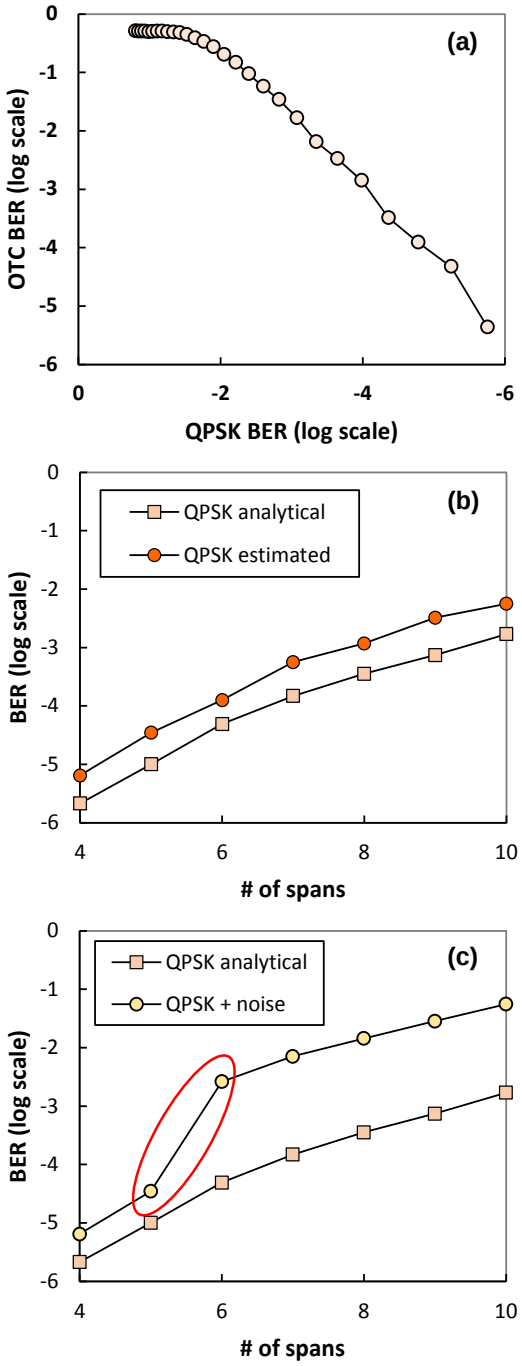


Fig. 9-5. (a) OTC vs. QPSK BER correlation. (b) Estimated QPSK BER vs. theoretical QPSK BER. (c) Degraded BER and failure localization.

9.5.2 Optical Spectrum Analyzer

Regarding in-operation failure localization, we simulated a 30 GBd DP-QPSK signal that passes through 10 single mode fiber spans. After each span, an optical amplifier compensates for the accumulated attenuation of the fiber. Each node is

modeled as a single optical filter emulating optical switching functionality performed by several WSSs; filters bandwidth is set to 37.5 GHz, leaving 7.5 GHz as a guard band for the lightpath

Emulating the optical node architecture in Fig. 9-1, coarse-granular OSAs are placed after every filter to analyze the optical spectrum. OSAs have been configured with a granularity of 625 MHz.

Simulations have been carried out to produce a database of samples belonging to different failure classes (including normal operation):

- in case of *LaserDrift* failures, a frequency shift is applied to the laser emission frequency; the frequency of the local oscillator at the *Rx* side is configured accordingly;
- for *FilterShift* failures, a frequency shift is applied to the central frequency of filters;
- *FilterTightening* failures are emulated by modifying the bandwidth of the filter.

Regarding failure magnitudes, although we simulated a wide range of them, we considered as actual failures those with a magnitude higher than a certain threshold, while samples below the threshold were re-labeled as normal. Specifically, thresholds were set to 1.5GHz for *LaserDrift*, 3 GHz for *FilterShift*, and 32 GHz for *FilterTightening*. Recall that *FilterTightening* magnitude increases when filter bandwidth decreases.

It is worth mentioning that the *FilterTightening* failure needs to be distinguished from filter cascading, as described in section 2.2. In view of that, the training of classifiers and magnitude prediction modules has been carried out with a shorter testbed (only two spans and filters between *Tx* and *Rx*) to avoid mispredicting filter cascading as filter failure. 28 distinct configurations of failure and magnitude have been simulated, generating up to 500 different samples for training and testing. Each sample consists of 56 different features obtained at several power levels.

The first classifier in the classifiers module, which is in charge of identifying between normal and filter failures, provides no classification error. This result is the key for the failure localization process since we can conclude that the classifiers module provides perfect localization of a failed filter in the absence of filter cascading effects. The second classifier, used upon the localization of a failure to distinguish between *FilterShift* and *FilterTightening*, returns a classification error around 18%. Although this error is not negligible, it is worth noting that its negative impact is small since filter failure identification is not as crucial as filter failure localization. Finally, magnitude predictors were fitted with the above-mentioned features to provide highly accurate linear models with average errors below 5%.

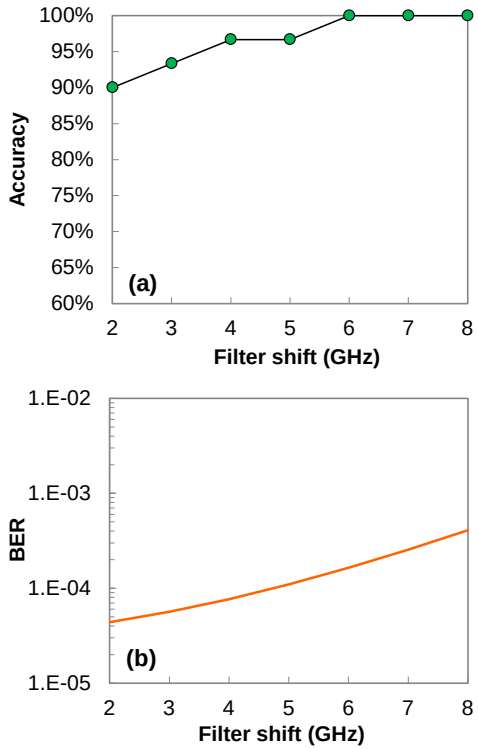


Fig. 9-6. FEELING performance for *FilterShift* in failure localization

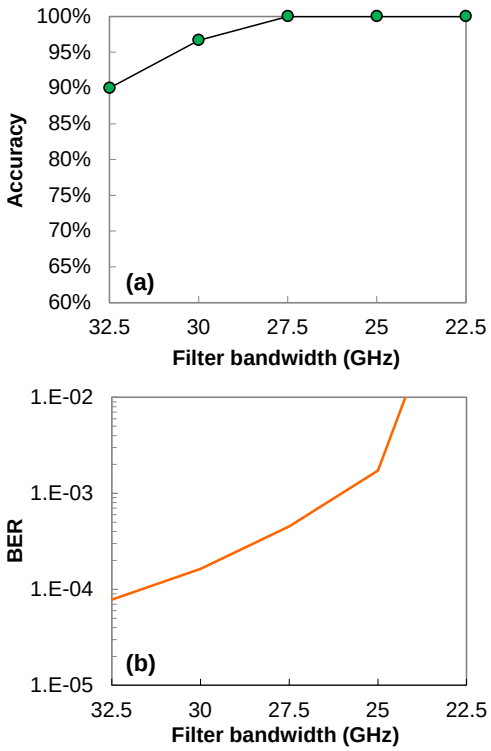


Fig. 9-7. FEELING performance for *FilterTightening* in failure localization

Once classifiers and predictors have been successfully trained and validated, let us evaluate the performance of FEELING. As for previous results, we carried out simulations for all failures and several magnitudes, considering only one failure per simulation.

For the case of *LaserDrift*, FEELING is able to localize the failure with 100% of accuracy, which is as a consequence of the goodness of the selected spectrum features, and the fact that in our simulations, the transmitter was collocated with the ingress node, and thus the signal arrives without any filter cascading effects. For the case of filter related failures, Fig. 9-6(a) and Fig. 9-7(a) illustrate localization accuracy for *FilterShift* and *FilterTightening*, respectively when considered filter mask correction. Accuracy in terms of the proportion of correct localizations is provided as a function of the magnitude of the failure, the conclusion is that, as soon as the failure magnitude increases, localization accuracy also increases. For *FilterShift* higher than 5GHz (Fig. 9-6 (a)) and for *FilterTightening* smaller than 28 GHz overall accuracy reaches 100%.

Finally, it is important to recall that FEELING is triggered upon the BANDO algorithm in Chapter 7 detects excessive BER in the reception of a lightpath. The calibration of BANDO includes BER thresholds that are setup in order to perform prompt and even anticipated BER degradation. Assuming a BER increase due to a gradual degradation of a filter, Fig. 9-6(b) and Fig. 9-7 (b) are provided to illustrate

the relation between BER change detection thresholds and failure localization. These figures depict the simulated BER as a function of failure magnitude. For illustrative purposes, let us imagine that, due to two different configurations, BANDO detects excessive BER at $8{\cdot}10^{-5}$ and $5{\cdot}10^{-4}$. Without entering into details, the former could correspond to a BER threshold violation anticipation while the latter could represent an actual threshold violation. A BER equal to $8{\cdot}10^{-5}$ could correspond to a degraded filter shifted around 4 GHz or narrowed until 32GHz, a failure that is localized with an accuracy around 90%. On the other hand, BER equal to $5{\cdot}10^{-4}$ is obtained for failures whose magnitude is large enough to perfectly localize them. Hence, modules for BER degradation and failure identification and localization must be configured with a global perspective to achieve optimal overall performance.

9.6 Concluding Remarks

Proper operation of the network components is a key factor to provide the expected QoS to the end-users and to avoid violating SLAs. Therefore, predicting upcoming failures that can disrupt the network operation, by continuous monitoring of the active lightpaths is of great importance. In this chapter, we proposed two monitoring systems to intelligently identify and localize soft failures during commissioning testing and lightpath operation.

In the case of commissioning testing, a low cost and complexity OTC system was proposed and validated as a promising technique for estimating the BER of 100Gb/s DP-QPSK modulated lightpaths. Simulations showed that the estimated BER can be used for testing and failure localization.

For the case of lightpath operation, a machine-learning based identification and localization algorithm (called FEELING) was proposed, taking advantage of continuous monitoring of the optical spectrum using cost-effective OSAs installed in the optical nodes. FEELING predicts whether a component failed and estimates the magnitude of the failure. As in the previous chapter, we focused on three classes of failures: *LaserDrift*, *FilterShift*, *FilterTightening*. In order to evaluate the accuracy of FEELING, we performed an extensive set of simulations, and the results showed that FEELING identifies/localizes *LaserDrift* with 100% of accuracy. In the case of filter related failures, FEELING can identify/localize the failure with the accuracy above 90%.

Chapter 10

Validation of the Distributed Monitoring and Data Analytics Architecture

This final chapter focuses on demonstrating the proposed distributed monitoring and data analytics architecture by validating the algorithms devised and evaluated in the previous chapters. Recall that the architecture consists on extended nodes that collate monitoring data and that are capable of performing data analytics and local decision making; in addition, a centralized MDA system is able to perform network-wide data analytics on monitoring data collected from all the nodes in the network. Specifically, the architecture is demonstrated by implementing three use cases that entail network cognition, namely: *i*) BER degradation detection and proactive lightpath restoration, *ii*) OD traffic anomaly detection based on predictive traffic model estimation and VNT reconfiguration, and *iii*) BER degradation detection and service layer connection re-routing.

10.1 Introduction

In this chapter we demonstrate our monitoring and data analytics architecture to support cognition in multilayer optical networks. Particularly, the contribution of this chapter is as follows:

- The proposed architecture is first motivated in Section 10.2, where a monitoring hierarchy to allow monitoring across different nodes of the same or different layers is proposed. In addition, Section 10.2, details this architecture that brings distributed and centralized data analytics to the network. The architecture includes a central MDA system performing wide-scope data analytics and extended nodes capable of performing local data analytics and decision making.

- To demonstrate the proposed architecture, three OAA loop use cases are defined in Section 10.3, as well as their workflows on the proposed architecture. Besides, data analytics algorithms to support Knowledge Discovery from Data (KDD) in the extended nodes and in the MDA system are proposed. The defined use cases are: *i*) BER degradation detection and proactive lightpath restoration, *ii*) OD traffic anomaly detection based on predictive traffic model estimation and VNT reconfiguration, and *iii*) BER degradation detection and service layer connection re-routing.

For the first use case, the BANDO algorithm from Chapter 7 is re-implemented and adapted for the considered extended node and the proposed architecture. Recall that BANDO algorithm detects BER changes in the optical connections by comparing BER measurement with boundaries (see Section 7.3). Besides, in certain cases, BANDO is able to extract a pattern of BER behavior in order to make predictions. Additionally, BANDO algorithm sends notifications towards the MDA including the severity level for BER changes.

The second use case includes the anomaly detector algorithm proposed in Chapter 5. That algorithm is adapted to be implemented inside the extended node. For anomaly detection purposes, the estimator (running in the MDA system) fits specific models for every OD pair which, in turn, are sent towards the extended node to be stored. Monitoring data collected from the nodes is directly compared against traffic model estimation, detecting therefore any anomaly that might occur. The ODEON optimization problem proposed in Chapter 5 is involved in case of VNT reconfiguration is needed.

Finally, the third use case considers a multilayer network affected by a BER degradation. For this particular case we recall SCULPTOR algorithm from Chapter 8 for demand re-routing. It is worth noting that SCULPTOR algorithm is only triggered by certain BANDO notifications.

10.2 Distributed and Centralized Data Analytics

10.2.1 Motivation and General Concept

In this section, we motivate the proposed architecture, whose main concept is illustrated in Fig. 10-1. The data analytics function is distributed in the architecture, i.e., optical and packet nodes are extended with monitoring and data analytics capabilities, and the control and management plane in charge of the network is extended with data analytics capabilities (MDA system).

In line with [RFC7011], we call *observation points* (*OP*) to location in the infrastructure where measurements can be performed. In connection-oriented networks, typical OPs include:

- Optical connections (lightpaths), where BER and P_{Tx}/P_{Rx}, among other parameters, can be measured at the optical transponders.
- L2-LSPs, where packets and bit rate can be measured in any packet node along the route of the LSP.

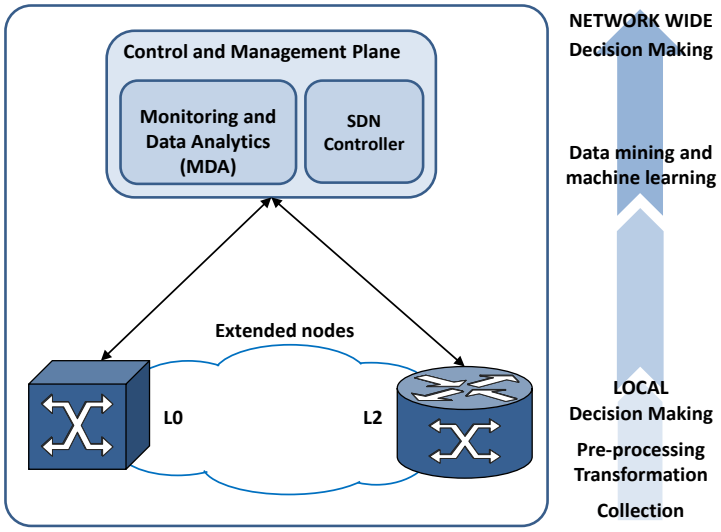


Fig. 10-1. Conceptual architecture.

Extended nodes' data analytics capabilities enable local KDD, so raw monitoring data collected from the physical devices is pre-processed, transformed, and modeled to fit the requirements of local KDD algorithms. Based on the obtained knowledge, the node can make local decisions and trigger notifications to the control and management plane.

Although extended nodes enable monitoring and distributed data analytics, the scarcity of computational and storage resources along with their strictly local perspective becomes insufficient when large datasets need to be processed and global network knowledge be discovered. Consequently, data analytics capabilities are also needed at the control and management plane. The centralized MDA system is thus responsible for:

- collating, processing, and storing monitoring data
- producing predictive models
- processing notifications received from the extended nodes
- managing the configuration of extended nodes
- issuing reconfiguration recommendations to the SDN controller

With this global vision, the MDA system is able to perform a network-wide KDD. Since large amounts of data need to be collected, stored, and analyzed, the MDA system needs to be designed with big data capabilities.

As an example, these capabilities would allow the MDA system to find predictive models for the traffic of L2-LSPs from monitoring data and send the predictive models to the extended nodes. The extended nodes can then compare the metered traffic against the predictive model and notify the control and management plane in case a traffic anomaly is detected.

Aiming at facilitating decision-making, we extend the definitions in [RFC7011] and propose defining two different monitory elements: *i*) OPs are locations in the network where monitoring data records are generated (as in [RFC7011]), *ii*) *observation groups* (OGs) collect data recovered from a set of observation points, which monitoring data needs to be aggregated.

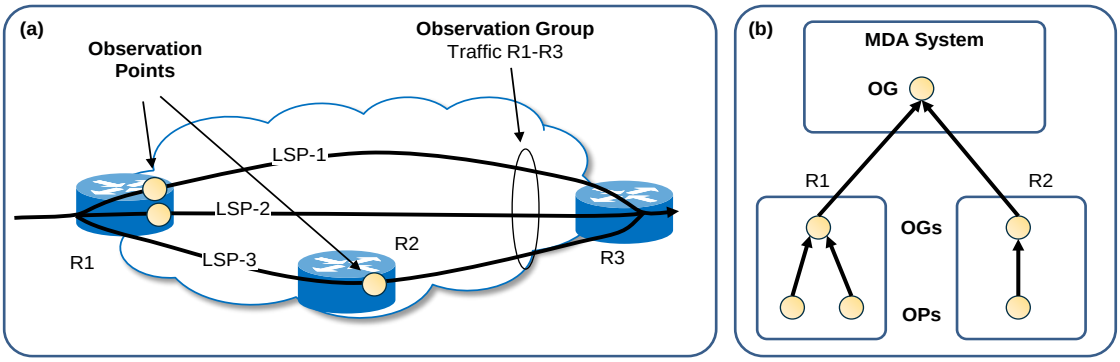


Fig. 10-2. (a) Monitoring observation points and groups, and (b) hierarchy.

For illustrative purposes, Fig. 10-2(a) presents an example where three separate L2-LSPs that convey the traffic of OD pair R1→R3. To measure the OD traffic, that of each individual L2-LSP needs to be metered and aggregated. L2-LSPs can be monitored by activating one single observation point in any packet node along their route, which is advantageous since the number of OPs that can be configured in every packet node is limited. Notwithstanding OPs have been activated for the three LSPs in our example in Fig. 10-2(a), not all of them are in the same packet node, so an OG that aggregates monitoring data records received from every L2-LSP supporting the OD traffic is configured.

This hierarchy is set up in the control and management plane and the MDA system sends it to the corresponding nodes, so data can be aggregated, if possible, directly in the extended node or, otherwise, in the MDA system. In the example (see Fig. 10-2(b), node R1 collects traffic samples for LSP-1 and LSP-2 and exports them to the MDA system as a single, aggregated traffic sample for OD R1→R3. The remaining OP data for LSP-3 in node R2 will be aggregated to that from node R1 once data arrive in the MDA system, to produce the final sample for the OD traffic.

In the next subsection, we present the proposed monitoring and data analytics architecture to support OAA loops.

10.2.2 Proposed Architecture

Fig. 10-3 presents an evolution for the proposed distributed architecture in Chapter 6. One of the main differences relies on the proposal of an extended node where several processes are now carried out *in situ*. Besides, now extended nodes receive monitoring messages from physical devices containing data records from multiple observation points.

First, monitoring samples are stored in a temporal repository allocated for every observation group; the repository consists of a queue where the latest N aggregated samples are stored. Time parameter G (granularity) specifies the period used to aggregate samples received from the observation points (e.g., 1 min), whilst time parameter T specifies the monitoring period (e.g., 15 min) used to aggregate and export observation group samples to the MDA system. Second, after monitoring data is collected and processed, modeled data is sent towards the MDA system for further analysis. Besides, due to the fact that the extended node has access to fine-grained monitoring data, among its new capabilities, it is worth highlighting the ability to perform local data analytics. For example, the anomaly detection algorithm can analyze fine-grained OD traffic monitoring data to detect anomalies and the BANDO algorithm can analyze BER monitoring data to detect BER degradations.

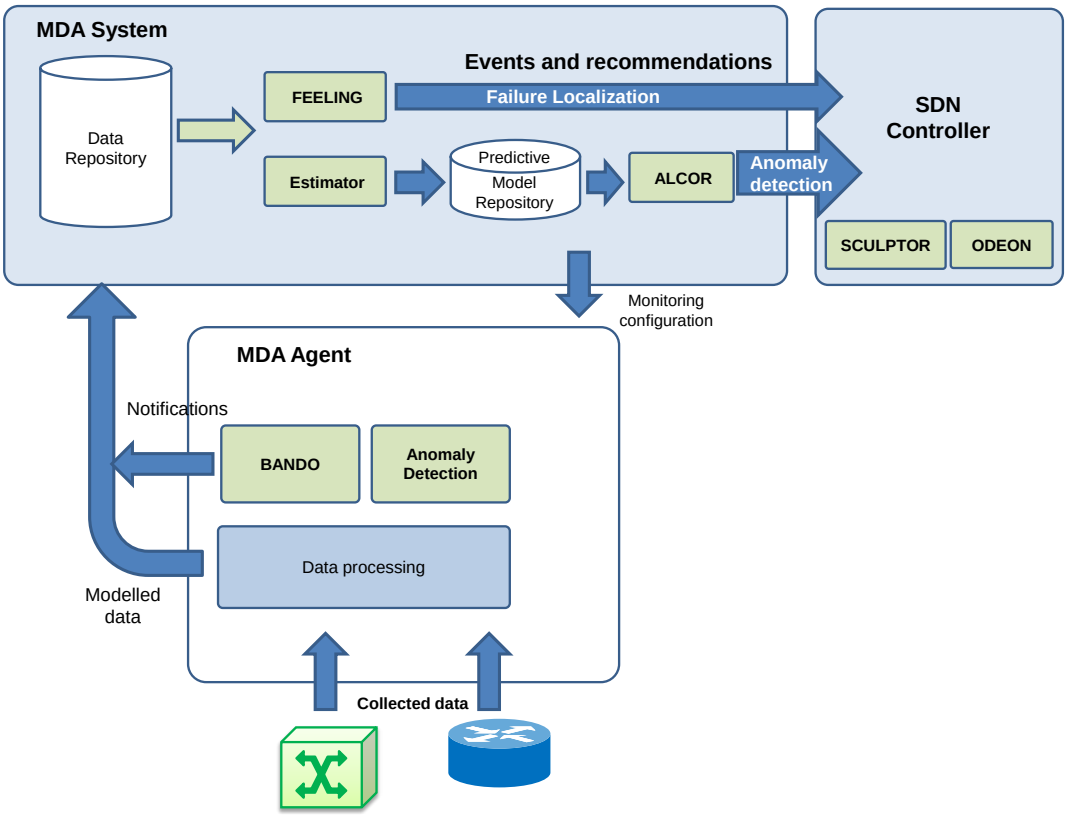


Fig. 10-3. Evolution of the distributed architecture for anomaly and failure detection.

The MDA system receives both, modeled data from the data processing in the extended node for model estimation and BANDO's notifications concerning BER measurements in the optical layer. In the MDA system, the OD traffic modeled data is received as input for the Estimator module which is in charge of estimating the specific traffic models for every OD pair. Then the new traffic models are stored in the Predictive Model Repository and used by the ALCOR algorithm. According to the notifications received from the anomaly detection algorithm, a change in the packet node configuration can be applied e.g., changing thresholds ε_S and ε_A (see Chapter 5 and Chapter 6). In case an anomaly has been detected a message is sent towards the SDN controller for recovery VNT reconfiguration; to that end, the ODEON algorithm can be used (see Chapter 5). Besides, BANDO's notifications are received in the MDA system as an input for FEELING algorithm (see Chapter 8); for failure localization during lightpath operation. For the sake of SLA fulfillment, the SCULTPOR algorithm in the SDN controller can be used to re-route the affected demands upon the reception of BER degradation notifications

One of the most interesting aspects of the extended node is the programmability of the local KDD. The extended node is able to receive configurations from the MDA system such as e.g., processes configuration and aggregation parameters. In addition, the control interface also includes a notification system toward the controller. This bidirectional exchange of data between the extended node and the MDA system opens the possibility to combine distributed and centralized data analytics workflows.

In the next section, we present the workflows for three demonstrative OAA use cases, as well as the adapted version of data analytics algorithms to support KDD in the extended nodes and in the MDA system.

10.3 OAA Control Loop Uses Cases

In this section, we demonstrate the architecture defined in the previous section by defining three different uses cases, namely: *i*) BER degradation detection and proactive lightpath restoration, *ii*) OD traffic anomaly detection based on predictive traffic model estimation and VNT reconfiguration, and *iii*) BER degradation detection and service layer connection re-routing. As previously mentioned, algorithms that have already been proposed in previous chapters are revisited so as to integrate them in the experimental validation.

10.3.1 L2 Traffic Estimation, Dissemination and Anomaly Detection

The second use case is related to traffic anomaly detection by comparing monitoring data with a traffic model. However, model estimation entails monitoring and processing time-series collated from OPs configured in different

nodes and spanning for a long time, e.g., several weeks; therefore, model estimation needs to be performed in the MDA system. In contrast, a traffic anomaly needs to be detected as fast as possible by comparing aggregated traffic samples against the predictive traffic model. Aiming at achieving minimum anomaly detection times, traffic anomaly detection process can be accomplished as a local KDD process running in the extended nodes, so traffic models produced in the centralized domain data analytics need to be disseminated to the corresponding extended nodes.

With this in mind, let us present the workflow needed for L2 traffic estimation and dissemination Fig. 10-4. First, packet nodes send monitoring messages encoding data records with measured traffic data to the corresponding extended node (message 1 in Fig. 10-4). When the extended node receives monitoring data, it processes and stores samples locally. Periodically (according to parameter T), extended nodes export aggregated samples to the MDA system (message 2). The MDA system, specifically the Estimator module, will eventually produce a new predictive OD traffic model that and will be disseminated to the corresponding extended nodes (message 3). The purpose of this process is to enable anomaly detection by comparing received monitoring with estimated traffic values at the extended node.

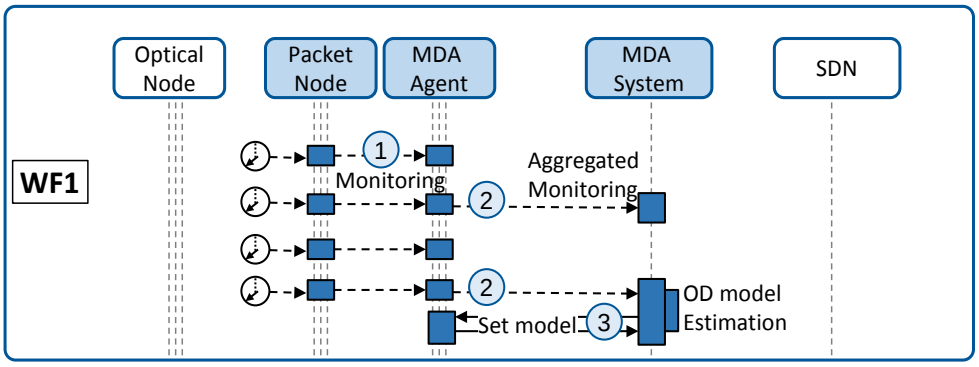


Fig. 10-4. Model estimation and dissemination.

Let us consider some examples of VNT reconfiguration using traffic prediction. In the example in Fig. 10-5(a) predictive traffic models have been estimated. In this example, if OD traffic anomaly is detected, the capacity of R2 - R3 vlink will be exceeded. Therefore, VNT reconfiguration needs to be performed, e.g., by creating a new vlink connecting R2 – R5. Fig. 10-5(b) shows that such new vlink has been set-up to cope with the unexpected traffic increment and LSP 02-05 has been re-routed. It is worth noting that vlinks in Fig. 10-5 are supported by lightpaths in the optical layer even if that is not shown in the scheme.

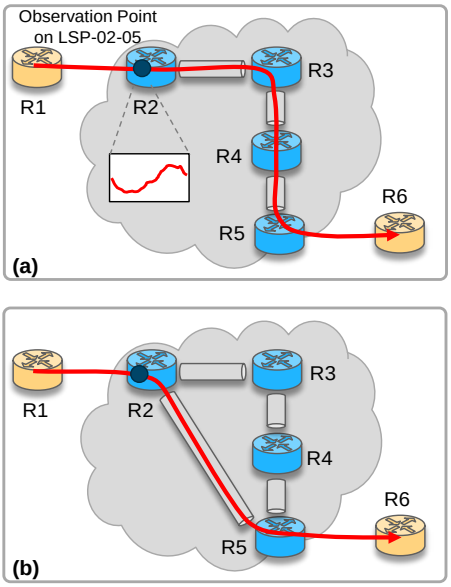


Fig. 10-5. Example of VNT reconfiguration using traffic prediction and after traffic anomaly detection.

After a predictive model has been estimated and made available in the extended nodes, an anomaly detector algorithm running in the extended node is executed every time a new traffic sample arrives in the temporal repository.

Usually a finer period than that used to export samples to the MDA system is used between the real and the extended node, e.g., 1 min.

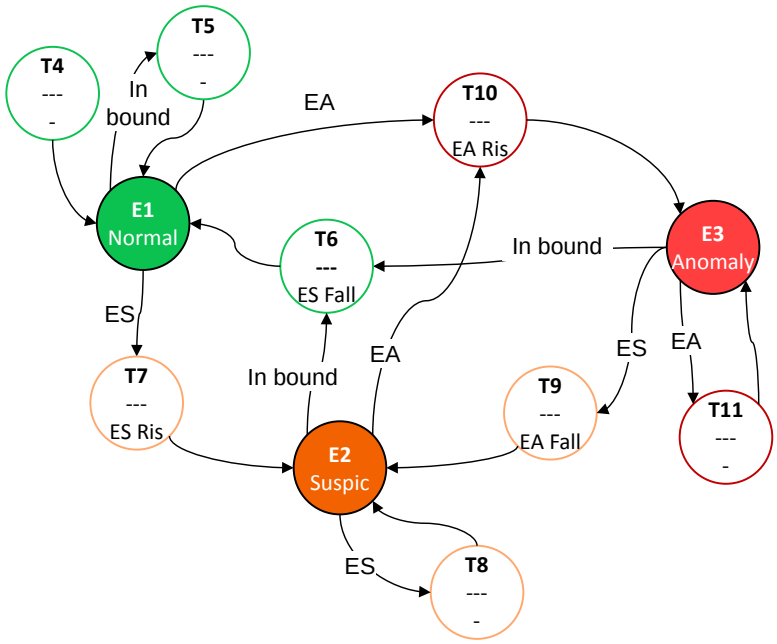


Fig. 10-6. Traffic anomaly detection finite state machine.

The traffic anomaly detection algorithm defined in Chapter 5 has been extended so it can be interpreted in an experimental environment. The new algorithm has been designed as an fsm with three main states and 8 transient states (Fig. 10-6); main states are used to store whether score status is normal or has exceeded a threshold, whereas transient states are used to produce notifications on threshold crossing. Every time a new sample arrives, two fsm transitions are performed, one to obtain the right output and action, and another to move to the new main state. The complete algorithm is detailed in Table 10-1.

Table 10-1 Experimental Traffic Anomaly Detection Algorithm.

```
INPUT obsGroupId
OUTPUT <sucess, error>
 1:  z=<y, time> ← getLastSample(obsGroupId)
 2:  if z=Ø then return <false, "No sample">
 3:  model ← getModel(obsGroupId)
 4:  if model=Ø then return <false, "No model">
 5:  <μ, σ> ← evaluateModel(model, t) (Chapter 4)
 6:  ŷ ← normalize(y, μ, σ) (eq. (5-1))
 7:  Ŷ ← getLastNormalizedSamples(obsGroupId)
 8:  Ŷ ← Ŷ U {ŷ}
 9:  if ŷ < 3 then return <false, ->
10:  s ← computeScore(Ŷ) (eq. (5-3))
11:  <transient, output> = doTransition(currentState, s)
12:  <currentState, _> = doTransition(transient, s)
13:  if output <> none then
14:     z[] ← getLastSamples(obsGroupId)
15:     sendNotif(obsGroupId, z[], output)
16:  return <true, ->
```

The algorithm receives as input the OG with new samples, and starts by getting the last sample from the temporal repository along with the predictive model (lines 1-4). Next, the sample bitrate is normalized with respect to the predicted average and standard deviation values (lines 5-6) and it is added to a series of previous normalized bitrate values (lines 7-8). The metered bitrate value is assumed normal if it is not higher than 3 times σ; otherwise a *score* is computed for the series as detailed in Chapter 5 (line 10) to decide the outcome of the algorithm. Two anomaly detection thresholds are configured: *suspicious* (defined by parameter ε_S) and anomaly (defined by parameter ε_A). If the score violates or falls down an anomaly detection threshold, the event is notified to the domain controller (lines 11-15).

Finally, the workflow for anomaly detection and VNT reconfiguration use case is presented in Fig. 10-7. The workflow starts when a sample with and anomalous amount of traffic, as compared with the predictive traffic model, is received in the extended node (message 1 in Fig. 10-7). Upon the detection of the traffic anomaly, the KDD process sends a notification to the MDA system detailing the metered traffic in the observation group. The notification is conveyed to the registered KDD processes; in this case to the one in charge of reconfiguring the network, e.g.,

ALCOR defined in Chapter 6. When ALCOR decides that the VNT needs to be reconfigured to cope with the anomalous traffic detected, it sends a notification to the SDN (message 3). The new VNT topology, as well as the new route for the anomalous LSP can be obtained by solving the ODEON problem proposed in Chapter 5. The SDN controller eventually implements the results for ODEON by setting up the new lightpath and then, reporting the LSP where the anomaly has been detected.

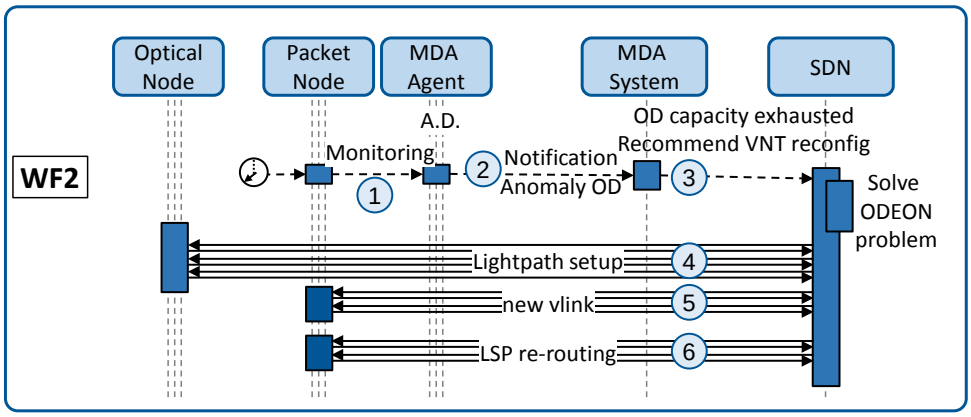


Fig. 10-7. Anomaly detection and VNT reconfiguration.

10.3.2 L0 BER Degradation with Lightpath and LSP Rerouting

For failure detection, the BANDO algorithm defined in Chapter 7, has been implemented as a finite state machine (fsm) with three main states and 11 transient states (Fig. 10-8); main states are used to store whether BER status is normal or has exceeded either the boundary or the threshold, whereas transient states are used to produce notifications and actions (i.e., boundaries re-estimation or reset). Every time a sample arrives, two fsm transitions are performed, one to obtain the output and action, and another to move to the new main state.

State E1 (*normal BER*) is reached when the last BER value falls below the boundary and the threshold. Transitions to transient state T1 follow BER within boundaries, while transitions from transient states T2 and T3 re-estimate the boundaries (see Fig. 7-4(a)). State E2 (*boundary exceeded*) is reached when the last BER value has exceeded the boundary, but it is still below the threshold (see Fig. 7-4(b)). Transitions from transient states T4 and T5 reset boundaries, so n-1 new samples are needed to arrive to re-compute new boundaries. Finally, state E3 (*threshold exce*eded) is reached when the last monitored BER is above the threshold (Fig. 7-4(b)). Transitions from transient states T7 and T8 reset boundaries, whereas from transient states T9 and T11 re-estimate them.

Recall that a prompt BER degradation detection helps meeting the committed QoS by re-routing the affected SLA-related demands; therefore the BANDO algorithm is used to monitor BER changes in the lightpaths.

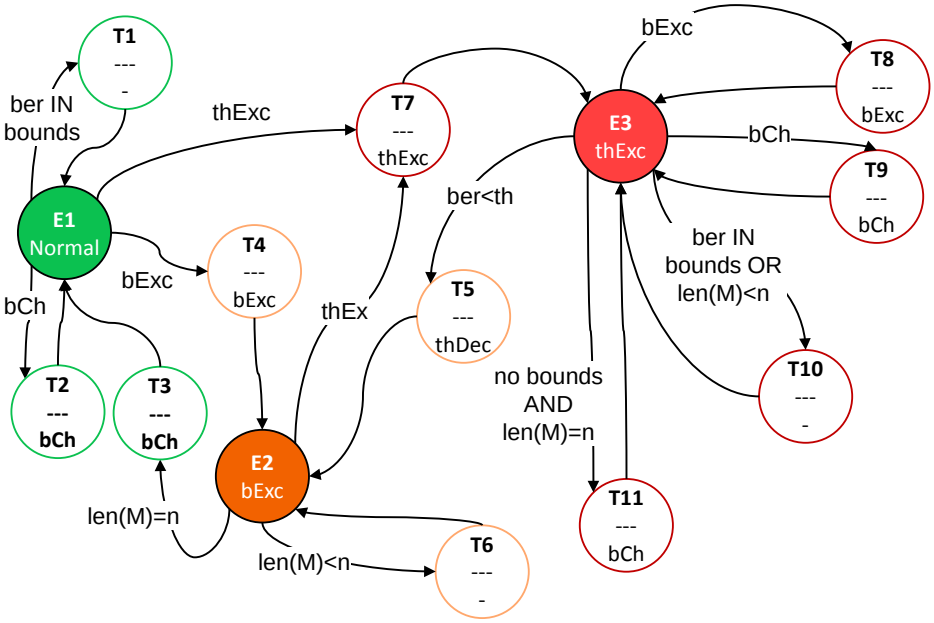


Fig. 10-8. BANDO finite state machine

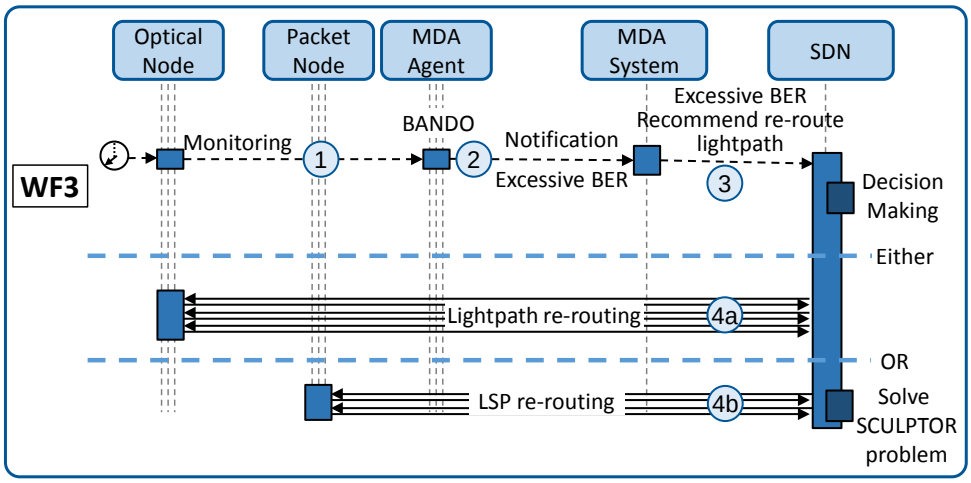


Fig. 10-9. Lightpath restoration after optical link failure prediction.

Fig. 10-9 presents the workflow considered for this use case. Initially, monitoring coming from the optical nodes is collected by the extended node in change of the optical node (message 1 in Fig. 10-9). Targeting at anticipating intolerable BER, the BANDO algorithm is used for detecting excessive BER in the received samples. Therefore, BANDO algorithm analyzes data, detecting, among others, gradual BER degradation. Depending on the case, a different notification (message 2) is be forwarded to the MDA which, in light of such detection, issues a recommendation to the SDN controller (message 3). Finally, managed by the SDN controller two different decisions can be taken depending on the circumstances. One decision could be to perform a lightpath re-routing directly over the optical nodes (message 4a). While, another option is to consider that the lightpath is supporting a vlink

and the reconfiguration is done solely at the L2 layer by running the SCULPTOR algorithm (message 4b) (see Chapter 8).

10.4 **Conclusion**

The dynamicity introduced by new connectivity services requires from control and managements architectures to help reducing resource overprovisioning. In that regard, this chapter proposed and demonstrated a hierarchical monitoring and data analytics architecture that consists of extended nodes that collate monitoring data, and are capable of performing data analytics on local data, together with a MDA system able to perform network-wide data analytics on monitoring data collected from all the nodes in the network.

The architecture has been demonstrated by defining three use cases in a multilayer network scenario: *i*) BER degradation detection and proactive lightpath restoration, *ii*) OD traffic anomaly detection based on predictive traffic model estimation and VNT reconfiguration, and *iii*) BER degradation detection and service layer connection re-routing.

Chapter 11

Visualization Tools

The previous chapters have focused on analyzing soft failures caused by filters and lasers that impact on the QoT of the lightpaths; in particular, algorithms for identifying and localizing the failed element have been developed. However, not only those failures need to be considered, as failures in other devices like optical amplifiers, can cause similar QoT degradations. In this chapter, we explore the use of data visualization techniques to guide human operators in failure localization tasks. Data visualization is of paramount importance for humans since it assists in the process to understand what is going on in the network. To that end, we propose the concept of *task-oriented visualization*, as the way to present charts that extract the fundamental information operators need; this entails not only collecting and storing, but also filtering and processing enormous amounts of monitoring data.

In this chapter, we consider the case where a subset of lightpaths' experience gradual BER degradation, i.e., BER evolves with time. We aim at using visualization techniques that guide a human operator to find out the most likely resource that is responsible for such degradation, as well as when such degradation started. Note that the challenge is to analyze an amount of data that might be huge (e.g., 1 sample every 15 min for the last, let us say, 4 weeks, for every lightpath in the network) and display charts in seconds, while revealing meaningful information for human operators. The proposed task-oriented visualization tackles such challenge by using charts that first help finding the subset of affected lightpaths (in this chapter we use *bubble chart* and *Sankey diagram*, just as examples) thus, reducing the number of lightpaths to be analyzed. In a second step, different charts are used to reduce the element in failure as much as possible (in this chapter a *column chart* or *spectrum color map* are proposed). Finally, the few suspicious elements can be analyzed in detail using more traditional *timeline* graphs.

11.1 Introduction

Data analytics architectures for optical networks like the presented in this PhD thesis, are facilitating the introduction of intelligence and cognition towards autonomous network operation. A challenging scenario where data analytics helps, as demonstrated in the previous chapters, is the localization of soft failures affecting optical systems. What makes soft failures difficult to detect is that the produced degradation can be initially very subtle and thus, very difficult to detect before lightpaths' degradation exceed some threshold. Recall that BANDO, running in the network nodes, and LUCIDA, running in the MDA controller, were presented in Chapter 7 to detect BER degradation and identify soft failures. Certainly, other algorithms could be devised to correlate several degraded lightpaths and localize the common set of links supporting the lightpaths. Nonetheless, in this chapter, we tackle the problem in a different way and present visualization techniques to guide human operators through myriads of data.

It is important to highlight, in the context of autonomous networking, that the role of human operators in the control and management of the network cannot be put aside, but the opposite; it should be reinforced by the availability and accessibility of rich and accurate monitoring data. Such large amount of monitoring data, however, needs to be adequately presented by means of advanced *operation-oriented* data visualization methods.

In fact, insightful visualization cannot simply consist in periodically plotting a set of charts trying to statistically summarize the current status of the network. Indeed, typical data visualization tools available in many management systems include *timeline charts* to represent single time-series, e.g., the evolution of BER monitored on a chosen lightpath. Commercial data visualization tools are including charts fueled by statistical and ML algorithms that allow extending dashboards with information extracted from manually selected monitoring metrics, e.g., to highlight whether a lightpaths' BER is likely to be anomalous [BluePlanet].

Specifically for the case of failure localization, many network management systems include some sort of data visualization to facilitate such task to human operators. Although these tools are really useful when the degradation is high, they fail to provide trend information, so the detection cannot be anticipated. For illustrative purposes, Fig. 11-1 shows an example, where two maps are presented for two different times t_1 and t_2; colors are used to give information about lightpaths' QoT thus, highlighting those with poor values. In particular, we are considering the case of a gradual degradation in a link, were increasing BER values have been collected for those lightpaths traversing such link; we are assuming that BER monitoring values are available for all the stablished lightpaths.

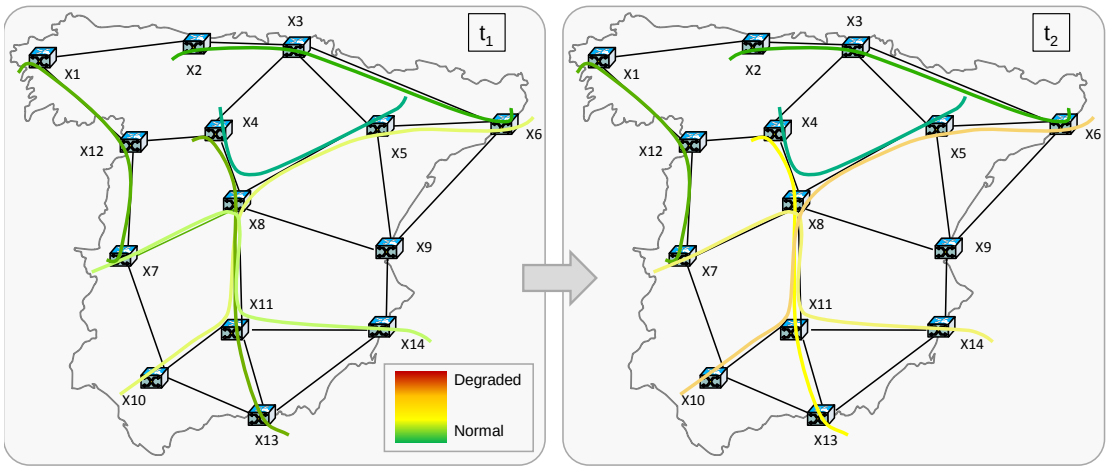


Fig. 11-1. Lightpath BER degradation evolution as a consequence of a soft failure in link X8-X11.

Some questions that cannot be answered looking at Fig. 11-1 at t_1 or t_2, are: *i*) is lightpaths' QoT normal? *ii*) Might be BER is a bit high but, should we be concerned? *iii*) Are lightpaths quickly degrading?

Instead the above network maps, *task-oriented* charts that are plotted according to a *visualization process* need to be specified for each desired use case, in the same way as different use cases require from different algorithms. More sophisticated charts need to be integrated to facilitate human operators to understand the relationship among monitoring metrics, find meaningful events, and correlate observable consequences caused by the same event.

11.2 Task-Oriented Visualization

Aiming to answer the above questions and to localize the element responsible for the degradation, we propose a *visualization process* that consists of a set of *chained* task-oriented charts.

When requested by the network operator, a huge amount of monitoring data belonging to all the established lightpaths collected during a period of time, need to be analyzed (left part of Fig. 11-2(a)). It is clear that one cannot simply use a timeline chart to plot the BER evolution for all the lightpaths, so other charts should be used. The visualization process that we propose consists in iteratively present the operator with charts that help him/her to reduce the number of suspicious elements causing a failure that affects a subset of lightpaths (right part of Fig. 11-2(a)).

Two examples of this chained process are presented in Fig. 11-2; the first one uses a bubble chart and the second one uses a Sankey diagram at the first step, where we aim at first to have an insight of which lightpaths show a suspicious behavior.

After this first step, one might want to know whether there is/are common resources responsible for such degradation, so other visualization charts are suggested for that end. Finally, once the number of suspicious elements is small, one-by-one analysis can be carried out. As it can be observed in the workflow in Fig. 11-2, we have analyzed a set of three chained task-oriented charts, however, other combinations of charts can be considered.

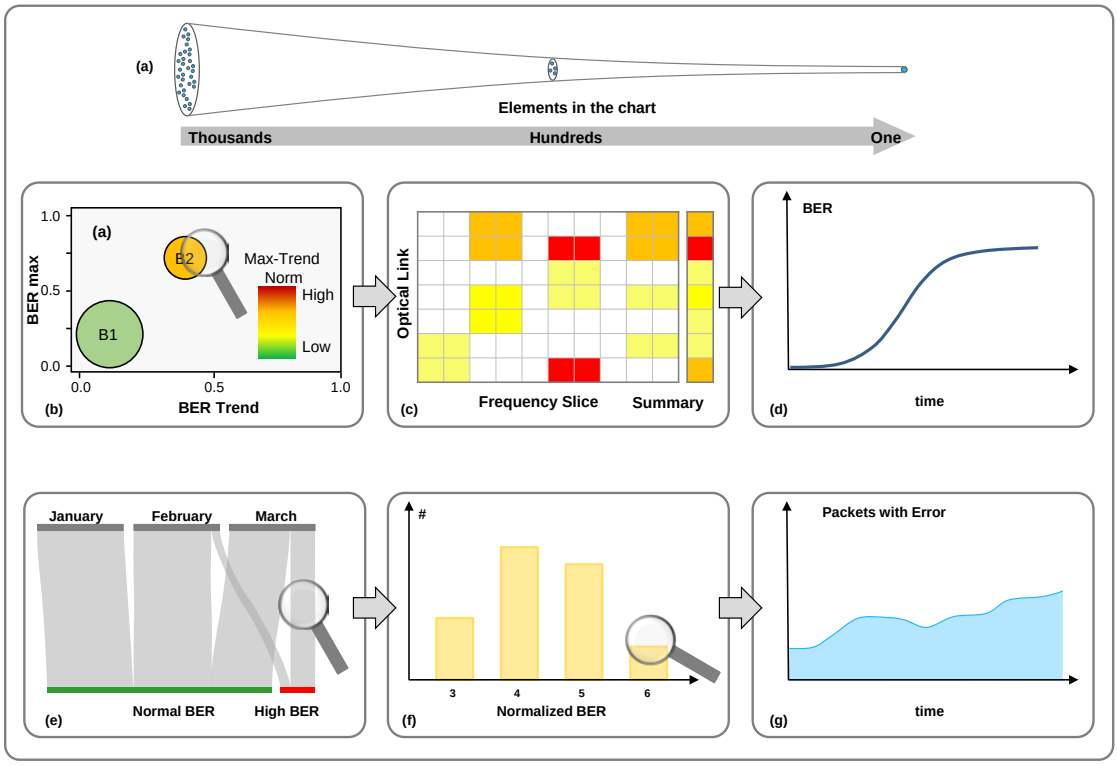


Fig. 11-2. Example of chained task-oriented applying visualization-assisted data filtering (a), bubble chart (b), network color spectrum map (c), historical BER (d), Sankey diagram (e), column bar plot (f), and historical packets with error (g).

11.2.1 Data Pre-Processing

Most of the visualization charts presented in this section require pre-processing monitoring data records to produce meaningful variables to be visualized. In the case of lightpaths' monitoring, data records contain, among others:

- time stamp (t);
- lightpath identifier (p)
- measured BER, BER_{tp}.

In addition, the lightpath operational database contains data about the lightpaths themselves, including their route and spectrum allocation, length, and estimated BER reference value, BER_Ref_p. Finally, let us assume that a global BER

threshold, *BER_Thr*, is configured as a limit of BER for all the lightpaths in the network. From such data, the pre-process phase transforms lightpaths' BER measurements producing a new variable, BER'_{tp}, representing the BER within the interval [BER_Ref_p, *BER_Thr*]; i.e.,

$$BER'_{tp} = \frac{\left(BER\_Thr - BER_{tp}\right)}{\left(BER\_Thr - BER\_Ref_p\right)}, \quad (11\text{-}1)$$

where BER_{tp} is previously forced to be confined in the defined interval. Next, two variables are computed by aggregating BER'_{tp} data from a selected time period (*T*):

- Maximum BER in *T*, BER_Max_{Tp}, computed as the quartile with probability of 95% in order to avoid spurious values; and
- BER trend in the period, BER_Trend_{Tp}, computed using the averaged first and last BER'_{tp} values.

A visualization database, *visualizationDB*, is created combining these two variables together with useful data about the lightpaths; such database will be the input of the visualization algorithms that eventually will produce the charts.

Owing to the fact that visualization is fostered by colors, we use a color palette specifically designed to guide operators in finding problems in the network. The proposed color palette is defined as a set of concatenated non-overlapping segments of gradient color and threshold values in the continuous interval [a, b] ({[$color_a$, $color_b$], [a, b]>}, a, b in [0, 1]), where the color of a given data value in the interval [0,1] results from finding the segment representing the data and then computing the color in the defined gradient.

11.2.2 Detecting Lightpaths Affected by a Gradual Soft Failure

Bubble Charts

Specially tailored *bubbles charts* can be used to provide the specific information that network operators need to detect soft failures before they can degrade the QoT of established lightpaths. Fig. 11-2(b) illustrates the bubble chart resulting from the evolution from t_1 to t_2 depicted in Fig. 11-1.

The proposed bubble chart uses BER measures for all lightpaths in the network for a given period of time (e.g., the last month) available in the big data repository in the MDA controller. The main features of the proposed bubble chart are summarized as follows:

- **Number of bubbles**, it can be fixed to achieve the best data representation.

- **Bubbles' position,** giving information of the relation between the BER value w.r.t. the BER change (trend) in the period; the metrics are relative to the expected BER for every lightpath, computed using analytical formulae.
- **Bubbles' size,** gives information about the number of lightpaths a bubble includes.
- **Color of each bubble,** computed considering two different metrics: BER and BER trend, representing the L2-norm of the bubble's position.

The resulting bubble chart shows extreme usefulness to detect lightpaths with an increasing BER degradation within the considered time period. As an illustrative example, two bubbles are represented in Fig. 11-2(b) aggregating lightpaths with low BER and trend (*B1*), and those lightpaths with high BER and appreciable trend (*B2*). In view of bubble *B2*, the operator can decide to further analyze the cause of failure of the paths contained in such bubble; to this end, he/she selects bubble *B2* and chooses to represent the lightpaths in a new task-oriented chart.

The bubble chart algorithm in Table 11-1 uses the *k-means* algorithm to find points in a 2D space (i.e., centroids), so that paths are grouped by assigning them to the nearest centroid. Each centroid is characterized by the coordinates BER_Trend_{Tp} (x-axis) and BER_Max_{Tp} (y-axis), and by the list of paths contained in the centroid (line 1 in Table 11-1). Next, for each centroid, its color is computed according to the L2-norm of the vector representing its position, i.e., $\|(BER\_Max_{Tp}, BER\_Trend_{Tp})\|_2$, within the color palette (lines 2-4). Finally, the algorithm returns the set of bubbles *B* (line 5). According to bubbles' color, one can infer the severity of the paths enclosed; note that this will pilot the operator towards these suspicious paths.

Table 11-1 Bubble Chart Algorithm

```
INPUT visualizationDB, colorPalette, numBubbles
OUTPUT Bubbles
1:  C={<xPos, yPos, paths>} ← k-means(visualizationDB, numBubbles)
2:  B ← Ø
3:  for c in C do
4:    B ← B U {<c.xPos, c.yPos, |c.paths|, getColor(colorPalette,
      ‖(c.xPos, c.yPos‖2)>}
5:  return B
```

Sankey Diagram

Another useful chart for this task is the Sankey diagram depicted in Fig. 11-2(e). This chart is a type of flow diagram that helps visually to emphasize the major or minor flow transfer within a system. The proposed diagram includes:

- **Upper labels**, consider different periods of time, e.g., monthly, weekly periodicity.
- **Lower labels**, consider levels for the normalized BER values.

- **Ribbon thickness,** flow running from the upper to the lower label; it considers the amount of lightpaths that have a BER evolution that satisfies the conditions of both labels.

Considering the above, the operator can also select the Sankey diagram as another option to easily find lightpaths with high BER without visualizing every individual lightpath's BER evolution. In this task-oriented chart, all or a subset of the lightpaths in the network are selected and classified according to a given statistical function (e.g., the *max* function) that is applied to BER measures for every defined period (e.g., every month). For illustrative purposes, only two levels (normal and high) of normalized BER (measured BER over estimated BER) are shown in Fig. 11-2(e).

The pre-process to generate the Sankey diagram consists in computing a single normalized BER value per each of the lightpaths and time periods (e.g., weeks); we used the 95% quantile to this end.

11.2.3 Intermediate Filtering

The charts introduced in this section are assumed to be created after some previous filtering has been done, e.g., but not limited to, the bubble chart or the Sankey diagram presented above.

Network Spectrum Color Map

A network spectrum color map (Fig. 11-2(c)) is a matrix representing the optical links as rows and the spectrum slices as columns; the color of each cell inherits the color of the lightpaths, computed likewise as for the bubbles, i.e., using the L2-norm of its vector. The intention is to find common causes leading lightpaths appreciable BER values or BER trend. A *row-summary column* is additionally displayed to assist the operator in finding those links supporting the highest number of degraded lightpaths.

Column Plot

A column plot (Fig. 11-2(f)) is defined by different ranges of normalized BER for the columns. The pre-processing consists on computing the column's height according to the corresponding number of lightpaths in each range.

11.2.4 Single Element Analysis

The previous charts can led the operator to an accurate filtering, where coming from thousands of lightpaths (left part of Fig. 11-2(a)). At the end of the visualization process, only a small amount of lightpaths need to be further analyzed (right part of Fig. 11-2 (a)).

Fig. 11-2(d) and Fig. 11-2(g) use timeline plots to visualize the evolution of one single measurement with time.

11.2.5 Data Pre-Processing

Task-oriented charts might require examining long monitoring time-series data sets from a large huge of elements; e.g., analyzing BER time-series for the last month for a set of 5,000 lightpaths entails retrieving and processing about 110 MB of data. Although processes behind task-oriented charts are designed to produce the essential chart configuration data, i.e., chart elements, color scheme, etc., to highlight the desired visualization effect, monitoring data should be aggregated at collecting time aiming at reducing processing at visualization time. For instance, charts in Fig. 11-2((e) and (f) require only one data value per each ribbon and column, respectively; if such data pre-computation is done at collecting time, visualization will be accelerated.

11.3 Illustrative Results

To evaluate the proposed data visualization techniques, we carried out simulations on a realistic 30-node and 56-link multilayer network based on the Spanish Telefonica's optical network, where a 14-node VNT was defined. Besides, 800 100 Gb/s lightpaths using 3x12.5GHz frequency slices were set-up sequentially between randomly selected nodes; lightpaths' BER was computed considering the expected OSNR in the links (needed to compute BER_Ref_p) plus a randomly generated amount of errors; expected links' OSNR considered not only link's length but also its load [Po12]. A maximum pre-FEC BER that transponders can support before a lightpath is torn-down, was set up as *BER_Thr*. Finally, we emulated a gradual degradation in link F08-F09, which decreases its OSNR and hence, increases BER of lightpaths using this link.

Fig. 11-3 presents the results of applying the bubble chart algorithm previously defined on two very different scenarios: *i*) stable scenario (right), and *ii*) gradual degradation scenario (left). In both scenarios, lightpath BER measurements of the selected week (36 and 39) are visualized using three bubbles. The defined color palette is also presented.

In the first scenario (Fig. 11-3 right), one can observe that although one bubble appears with high BER, there is no trend, i.e., the cause of the high BER in the lightpaths is now stable. The operator could request to visualize previous weeks to find the period where the degradation happened. This leads to our second scenario, which is presented in Fig. 11-3 left; in this bubble chart for week 36, the operator clearly identifies one bubble with significant BER trend (bubble 3). A summary of the two bubble charts is presented in Table 11-2.

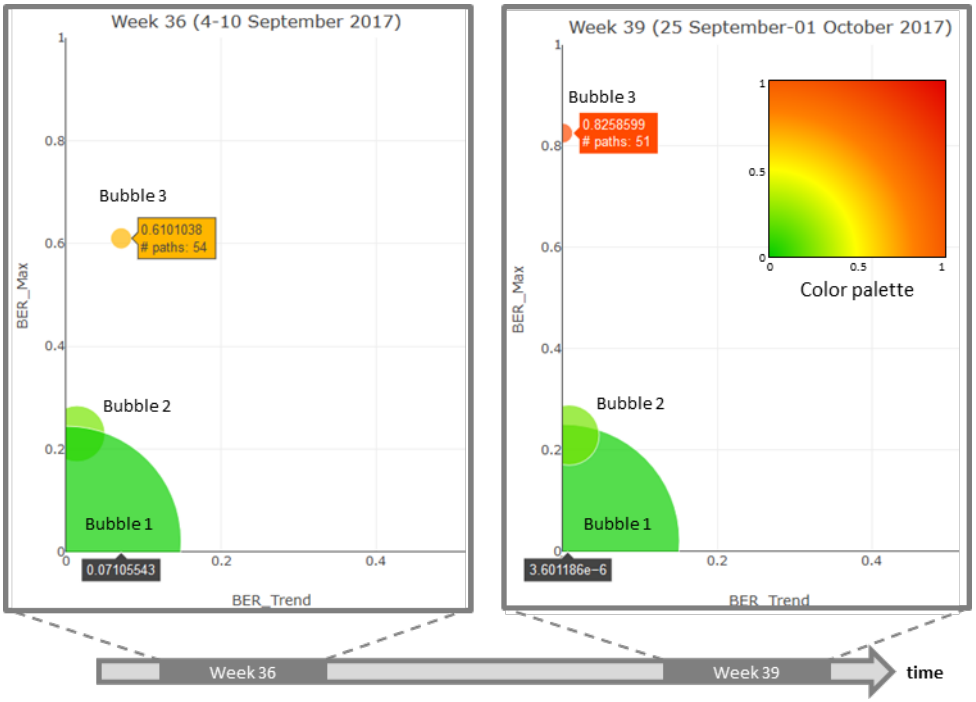


Fig. 11-3. Bubble charts for weeks 36 and 39.

Table 11-2 Bubble Chart Summary

	Bubble	Paths	BER Trend	BER max
Week 36	1	604	0.0	0.02
	2	142	0.01	0.23
	3	54	0.07	0.61
Week 39	1	595	0.0	0.008
	2	154	0.0	0.23
	3	51	0.0	0.826

Once the time when the degradation appeared has been identified, the operator might decide to find whether the cause of failure is in an optical link; to that end, he/she can select another operation-oriented chart to visualize a network spectrum color map. To clearly appreciate the goodness of the proposed visualization process, let us assume that no previous filtering is performed, so Fig. 11-4(a) presents the network spectrum color map when all the paths in the network (800 lightpaths) are selected.

Although a trained eye could perceive that few links might be the responsible for the degradation, such conclusion is not obvious. In fact, the row-summary column in the spectrum color map, which is intended to highlight the most likely degraded links, does not show any clear identification.

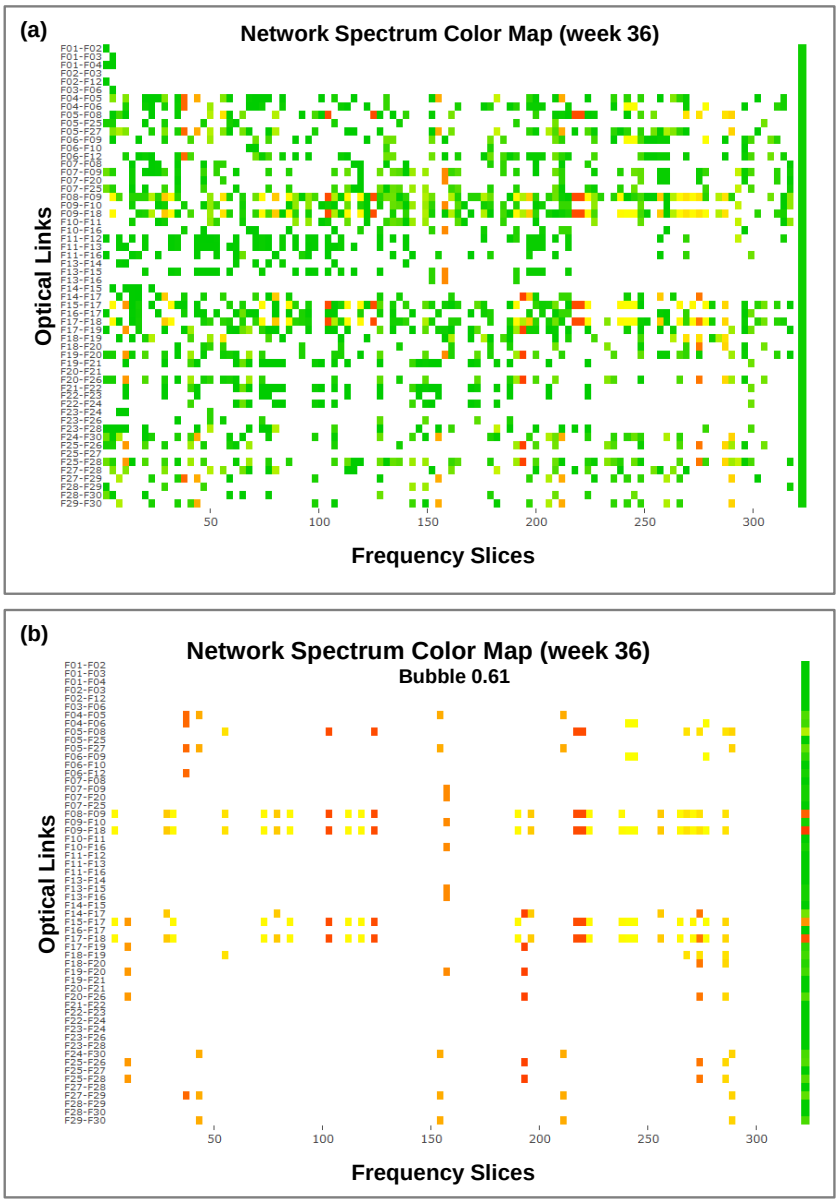


Fig. 11-4. Network Spectrum Color Map computed with all lightpaths (a) and with the lightpaths in bubble 3 (b).

Conversely, computing the spectrum color map with only the paths in bubble 3 that summarizes 54 paths, results is a much clear map (see Fig. 11-4(b)). In this case, the row-summary column identifies four links (out of 56) to likely be the responsible of causing degradation on the lightpaths. It is easy now for the operator to inspect one by one each of these optical links to find whether there is a clear responsible for the degradation.

Considering now the other chained task-oriented charts, the operator would first select the Sankey diagram (Fig. 11-5(a)), which consists in computing a single normalized BER value per each of the lightpaths and time periods (e.g., weeks); we

used the 95% quantile to this end. As stated in the previous section, the proposed Sankey diagram can define different labels. For the upper labels we select four consecutive periods of one week. In contrast, we define three severity levels for lightpaths' normalized BER lower labels as follows:

- *normal*, in the segment (-∞, 2.5];
- *warning*, confined in (2.5, 5]
- *anomalous* in the segment (5, +∞).

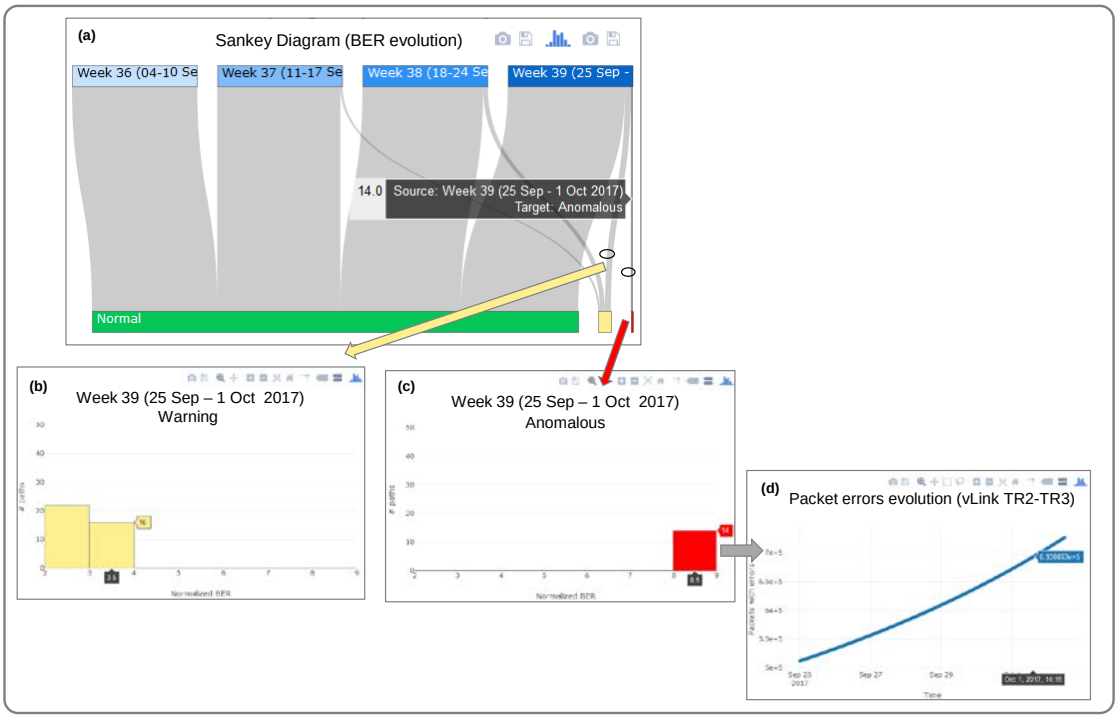


Fig. 11-5. Experimental chained task-oriented (a-c) and timeline (c) charts.

According to Fig. 11-5(a), the operator can then select a particular ribbon, a subset of lightpaths is then chosen, acting as a filter for the next chart or list, in our example, a column plot. Pre-processing for column charts defines different ranges for the columns and computes column's height according to the corresponding number of lightpaths in each range.

Fig. 11-5(b), presents the column plot for a certain number of lightpaths; particularly the ones that the last week have had a normalized BER value in the range of *warning*. Nevertheless, Fig. 11-5(c) depicts the same chart when only the anomalous lightpaths are considered. Finally, the evolution of packets with error (Fig. 11-5(d)) in the vlink supported by a selected lightpath is shown, where the time window in the X-axis in the plot relates to that of the previous plots in the chain. Optionally, the operator may enable "refresh" to see the evolution real-time.

11.4 **Conclusions**

In this chapter, we explored the use of data visualization techniques to guide operators in failure localization tasks. A data visualization process based on advanced graphical representation has been proposed for the localization of soft failures affecting lightpaths. In the first step, two *visualization task-oriented charts* can be used e.g., bubble charts using specific metrics or Sankey diagram have been proposed to identify, if any, those lightpaths deserving deep inspection because of unexpected high and/or increasing BER. Secondly, other charts e.g., network spectrum color map and column plots, have been proposed as an *ad hoc* technique for accurate localization of the failing optical fiber link. As proven by the illustrative results, using the proposed chained charts, operators can easily track network performance and speeding up health diagnosis through a powerful and simple visualization process.

Chapter 12

Closing Discussion

12.1 Main Contributions

The main contributions of this thesis are:

- Chapter 5 proposed a *score-based* different anomaly detection method for improving single traffic anomalies detection at layer 2. Once detected, the ODEON optimization problem was proposed to reconfigure the VNT. Besides, a monitoring and data analytics architecture was devised in order to reduce the amount of data to be conveyed and to minimize anomaly detection times.
- The ALCOR method was devised in Chapter 6 to deal with the case of multiple related traffic anomalies triggered by an external event. By anticipating whether other ODs are anomalous after detecting one anomalous OD pair, KPIs like the number of network reconfigurations, total reconfiguration time, as well as traffic losses were improved.
- Chapter 7 was devoted to BER degradation detection and failure identification at the optical layer. BANDO and LUCIDA algorithms were proposed to, first, detect significant BER changes in optical connections, and then, to identify the most probable failure pattern.
- The SCULPTOR algorithm was proposed in Chapter 8 to be triggered for demand re-routing after receiving certain BANDO notifications regarding significant BER change.
- Devoted to soft failure localization, Chapter 9 proposed two techniques for active monitoring during commissioning testing and for passive in-operation monitoring.

- Chapter 10, experimentally validated the distributed data analytics architecture presented in this PhD thesis through three representative use cases for autonomic networking in multilayer scenarios.
- Finally, Chapter 11 focused on the use of visualization techniques to help operators during failure localization procedure.

12.2 List of Publications

12.2.1 Publications in Journals

[JOCN18] **A. P. Vela**, B. Shariati, M. Ruiz, F. Cugini, A. Castro, H. Lu, R. Proietti, J. Comellas, P. Castoldi, S. J. B. Yoo, and L. Velasco, "Soft Failure Localization during Commissioning Testing and Lightpath Operation", (Invited) IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 10, pp. A27-A36, 2018.

[JLT18] L. Velasco, Ll. Gifre, J.-L. Izquierdo-Zaragoza, F. Paolucci, **A. P. Vela**, A. Sgambelluri, M. Ruiz, and F. Cugini, "An Architecture to Support Autonomic Slice Networking [Invited]," IEEE/OSA Journal of Lightwave Technology (JLT), vol. 36, pp. 135-141, 2018.

[JLT17.2] **A. P. Vela,** M. Ruiz, F. Fresi, N. Sambo, F. Cugini, G. Meloni, L. Potì, L. Velasco, and P. Castoldi, "BER Degradation Detection and Failure Identification in Elastic Optical Networks," IEEE/OSA Journal of Lightwave Technology (JLT), vol. 35, pp. 4595-4604, 2017.

[JLT17.1] L. Velasco, **A. P. Vela**, F. Morales, and M. Ruiz, "Designing, Operating and Re-Optimizing Elastic Optical Networks," (Invited Tutorial) IEEE/OSA Journal of Lightwave Technology (JLT), vol. 35, pp. 513-526, 2017.

[ComCom17] **A. P. Vela**, Marc Ruiz, Luis Velasco, "Distributing Data Analytics for Efficient Multiple Traffic Anomalies Detection," ELSEVIER Computer Communications, vol. 107, pp. 1-12, 2017.

12.2.2 Publications in Conferences

[ECOC18.1] **A. P. Vela**, Ll. Gifre, M. Ruiz, and L. Velasco, "Validating an Architecture for Advanced Data Visualization," submitted to European Conference on Optical Communication (ECOC), 2018.

[ECOC18.2] **A. P. Vela**, Ll. Gifre, O. González de Dios, M. Ruiz, and L. Velasco, "CASTOR: A Monitoring and Data Analytics Framework to Help Operators Understand What is Going on in Their Networks," submitted to European Conference on Optical Communication (ECOC), 2018.

[ONDM18] B. Shariati, **A. P. Vela**, M. Ruiz, and L. Velasco, "Monitoring and Data Analytics: Analyzing the Optical Spectrum for Soft-Failure Detection and Identification [Invited]," in Proc. IEEE/IFIP International Conference on Optical Network Design and Modeling (ONDM), 2018.

[OFC18.1] **A. P. Vela**, M. Ruiz, and L. Velasco, "Applying Data Visualization for Failure Localization," in Proc. IEEE/OSA Optical Fiber Communication Conference (OFC), 2018.

[OFC18.2] L. Velasco, B. Shariati, **A. P. Vela**, J. Comellas, and M. Ruiz, "Learning from the Optical Spectrum: Soft-Failure Identification and Localization [Invited]," in Proc. IEEE/OSA Optical Fiber Communication Conference (OFC), 2018.

[ICTON18.1] **A. P. Vela**, M. Ruiz, and L. Velasco, "Anticipating BER Degradation in Optical Networks", in Proc. IEEE International Conference on Transparent Optical Networks (ICTON), 2018.

[ICTON18.2] **A. P. Vela**, M. Ruiz, and L. Velasco, "Examples of Machine Learning Algorithms for Optical Network Control and Management", in Proc. IEEE International Conference on Transparent Optical Networks (ICTON), 2018.

[ICTON18.3] **A. P. Vela**, B. Shariati, M. Ruiz, J. Comellas, and L. Velasco, "Soft Failure Localization in Elastic Optical Networks", in Proc. IEEE International Conference on Transparent Optical Networks (ICTON), 2018.

[ICTON17.1] **A. P. Vela**, M. Ruiz, F. Cugini, and L. Velasco, "Combining machine learning and optimization for early pre-FEC BER degradation to meet committed QoS," in Proc. IEEE International Conference on Transparent Optical Networks (ICTON), 2017.

[ICTON17.2] **A. P. Vela**, M. Ruiz, and L. Velasco, "Bringing data analytics to the network nodes for efficient traffic anomalies detection," in Proc. IEEE International Conference on Transparent Optical Networks (ICTON), 2017.

[OFC17.1] **A. P. Vela**, M. Ruiz, F. Fresi, N. Sambo, F. Cugini, L Velasco, and P. Castoldi, "Early Pre-FEC BER Degradation Detection to Meet Committed QoS," in Proc. IEEE/OSA Optical Fiber Communication Conference (OFC), 2017.

[OFC17.2] Ll. Gifre, **A. P. Vela**, M. Ruiz, J. López de Vergara, and L. Velasco, "Experimental Assessment of Node and Control Architectures to Support the Observe-Analyze-Act Loop," in Proc. IEEE/OSA Optical Fiber Communication Conference (OFC), 2017.

[ACP16] **A. P. Vela**, M. Ruiz, and L. Velasco, "Reducing Virtual Network Reconfiguration and Traffic Losses under Multiple Traffic Anomalies," in Proc. Asia Communications and Photonics (ACP) Conference, 2016.

[ECOC16.1] M. Ruiz, F. Fresi, **A. P. Vela**, G. Meloni, N. Sambo, F. Cugini, L. Poti, L. Velasco, and P. Castoldi, "Service-triggered failure identification/localization through monitoring of multiple parameters," in Proc. European Conference on Optical Communication (ECOC), 2016.

[ECOC16.2] **A. P. Vela**, A. Via, M. Ruiz, and L. Velasco, "Bringing Data Analytics to the Network Nodes," in Proc. European Conference on Optical Communication (ECOC), 2016.

[ICTON16] **A. P. Vela**, A. Via, F. Morales, M. Ruiz, and L. Velasco, "Traffic generation for telecom cloud -based simulation," in Proc. IEEE International Conference on Transparent Optical Networks (ICTON), 2016.

12.2.3 Book Chapter

[VeRu17] **A. P. Vela**, F. Cugini, M. Ruiz, and L. Velasco, "Chapter 15. Toward Cognitive In-Operation Planning," in *Provisioning, Recovery and In-operation Planning in Elastic Optical Networks*, ISBN 978-1-119-33856-7, Wiley, 2017.

12.2.4 Other Works Not Included In This Thesis

[Ca16.1] A. Castro, **A. P. Vela**, Ll. Gifre, R. Proietti, C. Cen, Y. Jie, X. Chen, Z. Cao, Z. Zhu, V. Mishra, L. Velasco, and S.J.B. Yoo, "Experimental Demonstration of Heterogeneous Cross Stratum Broker for Scientific Applications," in Proc. IEEE/OSA Optical Fiber Communication Conference (OFC), 2016.

12.3 List of Research Projects

12.3.1 European Funded Projects

METRO-HAUL: METRO High bandwidth, 5G Application-aware optical network, with edge storage, compUte and low Latency, H2020-ICT-2016-2. (G.A. 761727)

12.3.2 National Funded Projects

- **SYNERGY**: Service-oriented hYbrid optical NEtwork and cloud infrastructuRe featuring high throuGhput and ultra-low latency. Ref: TEC2014-59995-R, 2015-2017 from MINECO (Ministry of Economy, Industry and Competitiveness)

- **TWINS**: cogniTive 5G application-aware optical metro netWorks Integrating moNitoring, data analyticS and optimization. Ref: TEC2017-90097-R, 2018-2020 from MINECO (Ministry of Economy, Industry and Competitiveness)

12.3.3 Pre-doctoral Funding Scholarship

This thesis has been funded by the grant FPI 2015-2019, related to the SYNERGY project.

12.4 Collaborations

I have completed a 3-month research stay in University California Davis (UC Davis) from mid-March 2017 until mid-June 2017, where I had the change to closely collaborate with people from the Next Generation Networking Systems Laboratory, being able to carry out interesting work published in a journal paper [JOCN18].

Besides, in different stages of the thesis, I had the opportunity to collaborate with people from Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), Pisa, leading to fruitful work compiled as publications in Journals or Conferences.

I am currently involved in the METRO-HAUL project, contributing to the following work packages (WP):

- WP2: Use Cases, Service Requirements, and Network Architecture Definition
    - Research on visualization tools for operational procedures [OFC18].
    - Design and evaluation of algorithms for network and connectivity reconfiguration that are aware of service requirements [ICTON17.1].
- WP3: Metro Node and Optical Transmission Solutions
    - Investigation and proof-of-concept validation of different use cases related to failure detection and localization and traffic anomaly detection, in collaboration with CNIT [JLT17.2], [ICTON17.2], [JOCN18].
- WP4: Network Control & Management of the Software-enabled Metro Network
    - Definition and experimental assessment of a monitoring and data analytics framework supporting network slicing, in collaboration with CNIT [JLT18].

12.5 Other Achievements

During the fall semester of 2016-2017, I had the chance to teach a first-grade course, *Introduction to Computers*, in the Faculty of Informatics (FIB) of UPC.

12.6 Topics for Further Research

Some of the algorithms and architectures devised in this thesis are being implemented on the experimental CASTOR platform. CASTOR, standing for Cognitive Architecture to Support Autonomic Network Slicing, is a monitoring and data analytics framework developed in our research group; focused on the idea of distributing data analytics so local decisions can be made closer to the network devices.

Appendix A – Examples Machine Learning Algorithms

This appendix details examples of R code for applying the different machine learning algorithms that have been introduced in Chapter 3 such as: *i*) regression, *ii*) SVM, *iii*) NB and *iv*) K-means algorithms.

Regression

It is one of the most used algorithms in statistics, based on finding the relation that explains the behavior between variables. In particular, this section details an example of R code (see code in Fig. A-1) for applying linear and non-linear models using the R function *lm*(·). The first part of the code creates a dataset. Later, different models are tried for fitting the real observations. Finally, as presented in Fig. A-1, the different models to fit the experimental points are depicted. Note that high order polynomials are not suitable since they tend to overfit original data.

```
# Creating dataset
x <- 1:10
y <- x + c(-0.3,0.3)
plot(x,y, xlim=c(0,11), ylim=c(-1,12))

# Different models, lineal and polynomial
fit2 <- lm( y~x )
fit3 <- lm( y~poly(x,3) )
fit5 <- lm( y~poly(x,6) )
fit4 <- lm( y~poly(x,9) )

# Plotting
xx <- seq(0,11, length.out=250)
lines(xx, predict(fit2, data.frame(x=xx)), col='orange')
lines(xx, predict(fit3, data.frame(x=xx)), col='green')
lines(xx, predict(fit5, data.frame(x=xx)), col='blue')
lines(xx, predict(fit4, data.frame(x=xx)), col='purple')
```

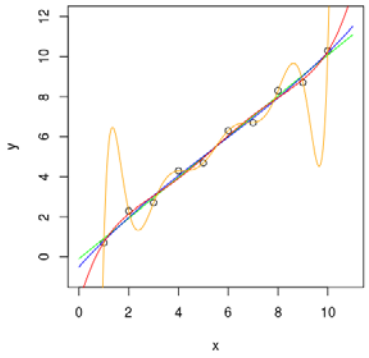


Fig. A-1 Example of R code for regression. The inlet figure points fitted with different equations (lineal and polynomial).

Support Vector Machine

SVM is a supervised learning technique really useful for binary classification since a boundary called *hyperplane* is computed and separates data into groups of similar features.

As an example of how to implement an SVM in R using the *svm*(·) r-package , we present the following Table A-1. As a summary, the pseudocode in Table A-2 resumes this R code. One of the basis of SVMs is to divide the dataset in the training and the testing groups (*training_set, testing_set*) as a percentage of 60%-80% and 20%-40% respectively (lines 1-2). Then, several parameters such as the cost *C* need to be fixed (line 4). Later, applying the package/library for SVMs a model is obtained (line 5); using this model, predictions for classification with the testing dataset as input are performed (line 6). As the correct classification labels for the testing data are available, a further study concerning errors can be done (line 7).

Table A-1 Example of R code for SVM.

```
# Installing libraries
install.packages("e1071")
library("class")
library(e1071)
library(rpart)

# Reading, processing and plotting data
trainingData =read.csv(file="trainingData.txt", header = TRUE, sep=",")
testingData =read.csv(file="testingData.csv", header = TRUE, sep=",")

# Rename columns
names(trainingData)[1]<-"x1"
names(trainingData)[2]<-"x2"

# Models: Linear Kernel changing cost parameter C
n = c(0.01, 0.1, 1, 10, 25, 100, 1000)
model  <-  svm(trainingData[,1:2],trainingData[,3],  type="C-classification",
cost=n, kernel="linear", scale = FALSE)
plot.prediction (model, paste0("linear kernel, C=",1))

# Computing predictions and frequency table of success
svm.pred <- predict(model, testingData[,1:2])
table(pred = svm.pred, true = testingData[,3])

# Getting parameters of hyperplane
plot(trainingData[,-(3:5)],pch=19,col=(trainingData$target+3)/2)
w <- t(model$coefs) %*% model$SV
i <- -model$rho

# in this 2D case the hyperplane is the line w[1,1]*x1 + w[1,2]*x2 + b = 0
abline(a=-i/w[1,2], b=-w[1,1]/w[1,2], col="blue", lty=3)
abline(a=-(i-1)/w[1,2], b=-w[1,1]/w[1,2], col="grey", lty=3)
abline(a=-(i+1)/w[1,2], b=-w[1,1]/w[1,2], col="grey", lty=3)
points(trainingData[model$index,c(1,2)],col="blue",cex=2)
```

Finally, this algorithm returns the model to allow classification of an unknown new dataset.

Table A-2 SVM algorithm pseudocode.

```
INPUT dataset
OUTPUT model(hyperplane, SVs)
training_set ← select a percentage of the dataset at random
testing_set ← dataset \ training_set
do
   select params (C, other)
   model ← fit SVM (training_set, params)
   predicted ← predict(model, testing_set)
   study error
while error intolerable
return model
```

An important property of SVMs is that the determination of the model parameters corresponds to a convex optimization problem, and so any local solution is also a global optimum [Bi06].

In the following examples we present the training dataset in Fig. A-2(a) and Fig. A-2(b) shows the testing dataset. As it can be seen, both are more or less similar. As previously seen, this case presents an example of non-linearly separable training class, therefore we need to solve the optimization problem presented in eq. (3-37) by changing the cost parameter C. Fig. A-3 shows an illustrative example for three different values for C showing the number of support vectors needed in all the cases. Polynomial kernels could also be used, but this simple example is well classified with a linear kernel.

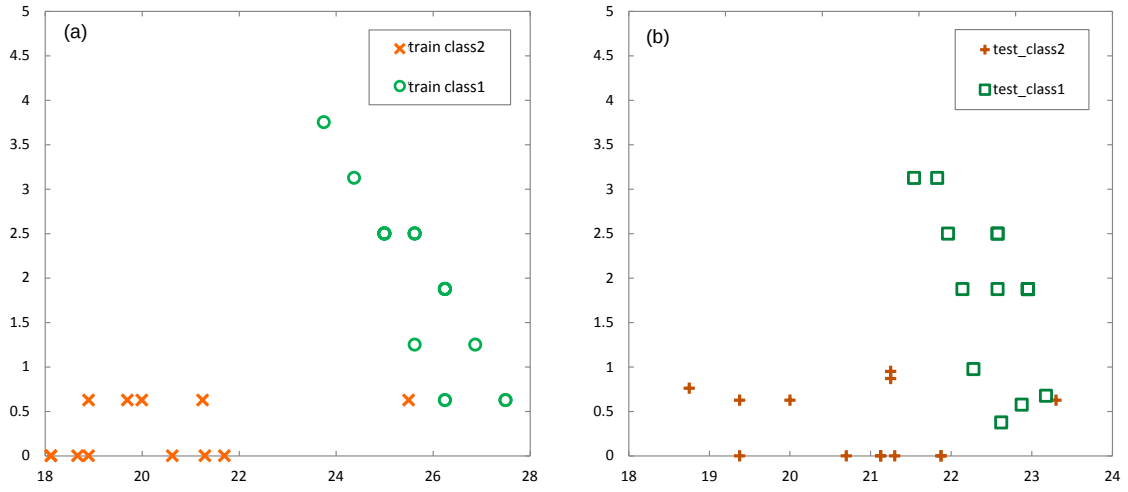


Fig. A-2 (a) Input data for training the SVM and (b) input data for testing the SVM model.

Analysis of parameter cost *C*

Fig. A-3 presents the use of lineal kernel with different *C* parameter values ranging from *C* = 0.1 to *C* =10). The *C* parameter allows to decide how mucho to penalize misclassified points. A low value of *C* parameter prioritizes simplicity (soft margin), while high *C* values prioritize making as few mistakes as possible, probably tending to overfitting. As it can be observed, for small values of *C*, the optimization chooses a bigger-margin hyperplane in contrast with the case of high values of *C*.

Fig. A-3(a) shows the computed hyperplane using *C* = 0.1, for this value, 6 support vectors are needed; then Fig. A-3(b) tests the just obtained model with a new dataset to evaluate its performance (detailed in Fig. A-3). Besides, the same study has been considered for other cost values such as: *C* = 1 (Fig. A-3 (c), (d)) and *C* = 10 (Fig. A-3 (e),(f)). It can be observed that when *C* increases to 1, the number of support vectors decreases down to 4. Note that when *C* = 10 or above, the number of support vectors remains equal to 3. In particular, with this high value of *C*, the constraint penalizes a lot making a classification mistake, therefore not allowing committing any misclassification. As it can be seen, in Fig. A-3 (e) even the most distant point in the training class 2 is correctly classified.

Table A-3 Confusion matrix.

	Real					
	C=0.1		C=1		C=10	
Prediction	Class 1	Class 2	Class 1	Class 2	Class 1	Class 2
Class 1	15	0	15	0	13	0
Class 2	0	13	0	13	2	13

Table A-3 resumes the performance of the models obtained changing the cost C parameter, where both, prediction and the correct classification are presented.

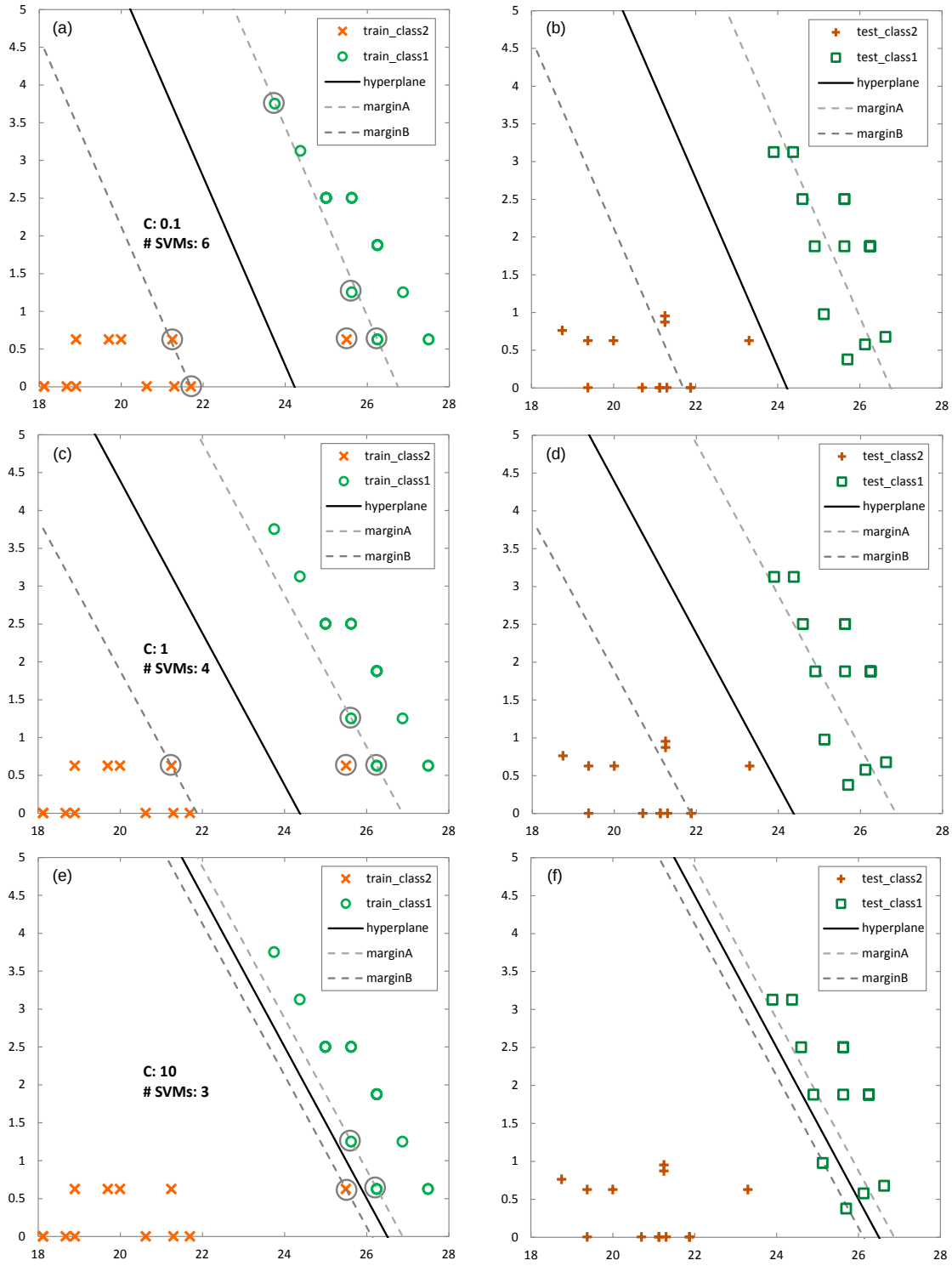


Fig. A-3. SVM hyperplane and margins computed with training data for three different costs (a) 0.1, (c) 1 and (e) 10. (b), (d) and (f) use the SVM to classify the testing data.

Naïve Bayes

This algorithm is a special case of a Bayesian Network (BN). In particular, the Naïve Bayes (NB) assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. However, this assumption is rarely satisfied.

Let us consider both, an example of R code explaining the implementation of *naive.bayes* R package (see Table A-4) and also a pseudocode of NB implementation (see Table A-5). For this example, a dataset of independent variables (*x1*, *x2*, *x3*) and a dependent variable *y* is generated. Firstly, a random dataset is generated and structured as a dataset (Table A-5, lines 1-4). As NB requires variables to be discrete, factorization needs to be performed on both variables (lines 5-7). Finally, using the *naive.bayes* package, the model is created (line 8) with which predictions can be made (line 9). Finally, line 10 returns the obtained model.

Table A-4 Example of R code for Naïve Bayes.

```
# Installing libraries
install.packages("bnlearn")
library(bnlearn) # data(learning.test) head(learning.test)

# Generating data
n=100
x1=runif(n)
x2=runif(n)
x3=runif(n)
y=floor(x1+x2+x3)
x1=floor(x1*5)
x2=floor(x2*5)
x3=floor(x3*5)
df=data.frame(x1,x2,x3,y)

# Discretize values into factors before applying Naïve Bayes
df[,'x1']<-factor(df[,'x1'])
df[,'x2']<-factor(df[,'x2'])
df[,'x3']<-factor(df[,'x3'])
df[,'y']<-factor(df[,'y'])

# Computing the Y prediction separately with each one of the variables
resNB_X1=naive.bayes(df[,c("x1", "y")], "y")
resNB_X2=naive.bayes(df[,c("x2", "y")], "y")
resNB_X3=naive.bayes(df[,c("x3", "y")], "y")
# Plotting Bayesian Network
plot(resNB_X1)
plot(resNB_X2)
plot(resNB_X3)
# Computing Y prediction with all the variables (X1, X2, X3)
resNB=naive.bayes(df, "y")
plot(resNB)

# Computing prediction of Y separately with each one of the variables
```

```
Ypred_X1=as.data.frame(predict(resNB_X1,df[,c("x1","y")], prob=FALSE))
Ypred_X2=as.data.frame(predict(resNB_X2,df[,c("x2", "y")], prob=FALSE)
Ypred_X3=as.data.frame(predict(resNB_X3,df[,c("x3", "y")], prob=FALSE)

# Computing prediction of Y with ALL  of the variables
Ypred = as.data.frame(predict(resNB, df, prob=FALSE))

# Create a dataframe with all Y predictions and Y real.
Yreal = as.data.frame(as.numeric(as.character(df$y)))
Ypredict_df=as.data.frame(c(Ypred_X1,Ypred_X2,Ypred_X3, Ypred, Yreal))
colnames(Ypredict_df)[4]="YpredAllX"
colnames(Ypredict_df)[5]="Yreal"

# Compare predictions with real Y value. Convert factors to numeric
dif_X1=Ypredict_df$Yreal-as.numeric(as.character(Ypredict_df$pred_X1))
dif_X2=Ypredict_df$Yreal-as.numeric(as.character(Ypredict_df$pred_X2))
dif_X3=Ypredict_df$Yreal-as.numeric(as.character(Ypredict_df$pred_X3))
dif_allX=Ypredict_df$Yreal-as.num(as.character(Ypredict_df$YpredAllX))
info_X1=as.data.frame(table(dif_X1))
info_X2=as.data.frame(table(dif_X2))
info_X3=as.data.frame(table(dif_X3))
info_Xall=as.data.frame(table(dif_allX))
posX1_OK = as.numeric(rownames(info_X1[info_X1$dif_X1==0,]))
posX2_OK = as.numeric(rownames(info_X2[info_X2$dif_X2==0,]))
posX3_OK = as.numeric(rownames(info_X3[info_X3$dif_X3==0,]))
posXall_OK = as.numeric(rownames(info_Xall[info_Xall$dif_allX==0,]))

# Computing probabilities of sucess
probOK_X1 = info_X1$Freq[posX1_OK]/sum(info_X1$Freq)
probOK_X2 = info_X2$Freq[posX2_OK]/sum(info_X2$Freq)
probOK_X3 = info_X3$Freq[posX3_OK]/sum(info_X3$Freq)
probOK_Xall = info_Xall$Freq[posXall_OK]/sum(info_Xall$Freq)
```

The example considers the number of points in the dataset n equal to 100 and the number of independent variables j is equal to 3. Besides, y stands for the dependent variable of the model, the one we want to predict.

Table A-5 Bayesian Network pseudocode.

```
INPUT n, j
OUTPUT model
 1: for each j do:
 2:    xj ← runif(n)
 3: y ← floor(sum(xj))
 4: df ← data.frame(x1,x2…xj,y)
 5: for each j do:
 6:    df[,'xj'] ← factor(df[,'x1'])
 7: df[,'y'] ← factor(df[,'y'])
 8: model ← naive.bayes(df, 'y')
 9: pred ← predict(model, df, prob=TRUE)
10: return (model)
```

The results for this example are presented in Table A-6, where, by inspection it can be seen that generating the model with only one of the variables or a combination of them has a direct influence on the probability of success.

Table A-6 Results for success with different models.

Probability of success modeling *Y~X1*	0.69
Probability of success modeling *Y~X2*	0.68
Probability of success modeling *Y~X3*	0.67
Probability of success modeling *Y~X1,X2, X3*	0.91

Clustering

The K-means algorithm is one of the most well-known clustering algorithms; its main steps were introduced in Chapter 3 and they are reproduced in Table A-7 to facilitate reading.

Table A-7 Steps in K-means algorithm.

Step	Action
1	Choose a number *K* of clusters.
2	Select the centroids by selecting *K* points at random, not necessarily from the dataset.
3	Assign each data point to the closest (Euclidean) centroid.
4	Compute and place the new centroid of each cluster.
5	Reassign each data point to the new closest centroid. If any reassignment took place, go over steps 4 and 5, otherwise finish.

The pseudocode for implementing the K-means algorithm is detailed in Table A-8. The input for the algorithm in Table A-8 is represented by *step 1*. Then, *step 2* can be found in lines 2-3. Then, *steps 2* and *3* concerning assignation and re-computation of centroids are stated in lines 2-7. Finally, *steps 4* and *5* are repeated in lines 8-9 until there are no more changes needed for centroid placement.

Consider Fig. A-4 where the steps for the *K*-means algorithm are detailed. This algorithm receives as input: *X* which is the dataset of samples to be clustered, *K* which is the number of clusters and *MaxIters* to limit the maximum number of iterations in this iterative algorithm. Besides, the output of this algorithm contains $C=\{c_1, c_2,\ldots,c_n\}$ which is the set of cluster centroids and *D* which is the set of cluster labels of $X=\{d(x) \mid x=1,2,\ldots,n\}$.

First, Fig. A-4(a) shows the initial dataset, where the selection of the clusters is not straightforward. We decide to set the number of clusters *K* equal to 2 (*step 1*) and place the centroids at random places as presented in Fig. A-4(b) (line 1). Then, we need to assign each of the points to the closest centroid (lines 2-7), where the optimization problem for minimizing the distance is specified in line 5. For the sake

of clarity, in this PhD thesis we are only considering Euclidean distances. Fig. A-4(c) shows a dotted line which is equidistant from both centroids; with this separation in mind, we can color the dataset according to the distance to the closest centroid, forming K clusters [DMR].

Table A-8 K-means pseudocode.

```
INPUT X={x1, x2,…,xn}, K, MaxIters
OUTPUT C={c1, c2,…,cn }
 1: ci ← computeRandomCentroids(·)
 2: for each ci ∈ C do:
 3:   ci ← xi ∈ X
 4: for each xi ∈X do:
 5:   d(xi) ← argminDistance(xi, cj) j∈{1,…K}
 6: change ← false
 7: i ← 0
 8: while changed≠false & iter ≠MaxIters
 9:   for each ci ∈ C do:
10:     updateCluster(ci)
11:   for each xi ∈X do:
12:     minDist ← argminDistance(xi, cj) j∈{1,…K}
13:     if minDist ≠ d(xi) then
14:       d(xi) ← minDist
15:       change ← true
16:     else change ← false
17:   i++
```

Later, *step 4* computes and places the new centroid of each cluster (Table A-8– lines 8-17), i.e., imagine that each of the points weights one *kg* and we need to find the *center of gravity*, as presented in Fig. A-4(d). Then, according to the new equidistant line, some points are now not well classified and we need to reassign each data point to the new closest centroid Fig. A-4(e). A reassignment took place, so then, *step 4* and *step 5* need to be redone as plotted in Fig. A-4 (f) and Fig. A-4(g). In case the equidistant line makes no points to be reassigned, means that the algorithm has converged and that the model is ready. Finally, Fig. A-4(h) and Fig. A-4(i) show the final clusters after an iterative process (variable *MaxIters* from Table A-8 is set to fix the maximum number of iterations allowed).

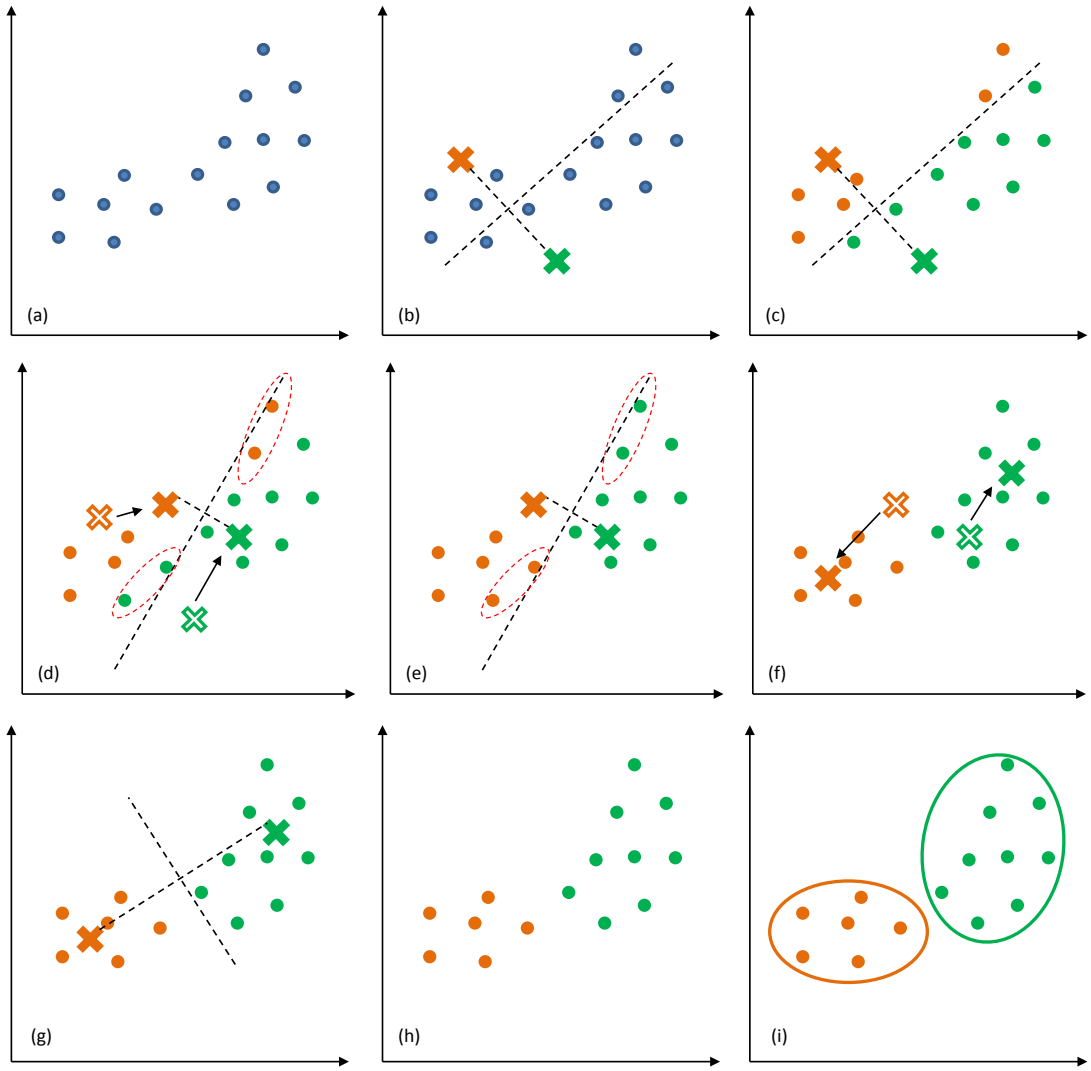


Fig. A-4(a) Input data, (b) selection of random centroids, (c) assigning samples to clusters, (d) new centroid placement, (e) samples reassigned to new centroids, (f),(g) iteration of the previous process, and (h),(i) final clusters.

For the sake of clarity, let us present a quick example of the use of the *kmeans* package in R [RCran].

Table A-9 kmeans R package, description and example.

Function	*kmeans*(*x*, *centers*, *iter.max* = 10)
Input	
x	numeric matrix of data
centers	either the number of clusters *K* or a set of initial cluster centers
iter.max	the maximum number of iterations allowed

Output

cluster	vector of integers (1:*K*) indicating the cluster where each point is allocated
centers	a matrix of cluster centers
totss	total sum of squares
size	number of points in each cluster
iter	number of iterations

Example

```
# Generating 2D data
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")

# Generating trying with different K values (2,3,4)
K=c(2,3,4)
cl <- kmeans(x, K)
plot(x, col = cl$cluster)
points(cl$centers, col = 1:K, pch = 8, cex = 2)
```

When *K* takes different values in Table A-9, we can obtain a set of plots as in Fig. A-5 where *K* ranges from 2 to 4.

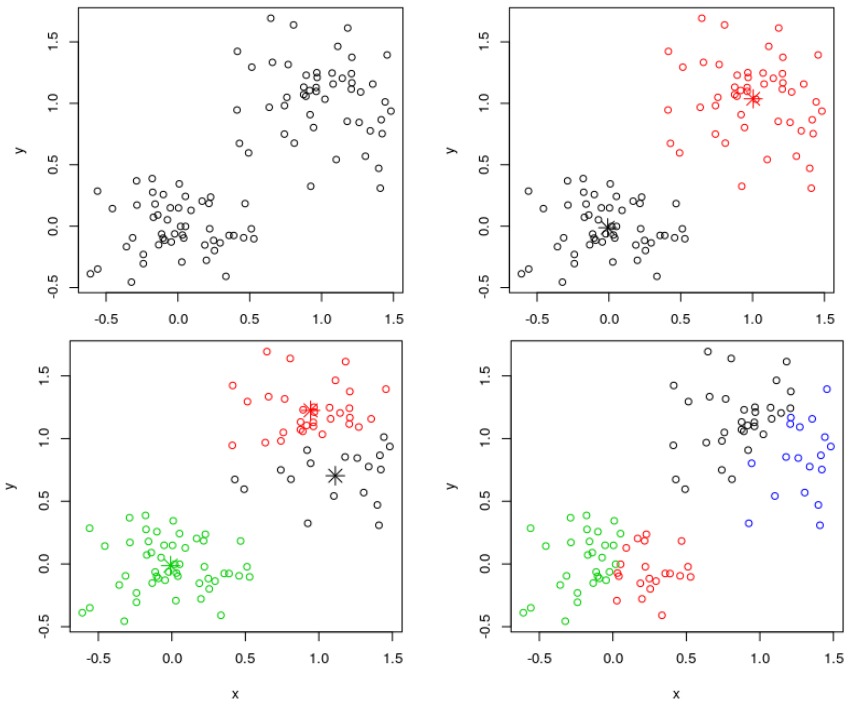


Fig. A-5(a) Input data, clustering with: (b) *K*=2, (c) *K*=2, and (d) *K*=4.

Appendix B - Traffic Modeling and Generation

Traffic generation is an important tool to study and characterize the network. This appendix presents general traffic profiles and the way they can be generated.

Let us denote $f(t;T)$ as the periodic function with period T returning the mean value of the model complex against time traffic. This function $f(t;T)$ can be generated in multiple ways. For instance, one can define a piecewise linear function, i.e., a function composed of a number of linear segments, aiming to reproduce the traffic behavior against time. Another way could consist on a polynomial of some degree, or even a summation of functions, e.g., trigonometric sines, as it will be presented.

In addition, some random values around the average value are usually observed as a result, among others, of the monitoring process. That random function ε_t can be modeled as a probability distribution function, e.g., following the normal (Gaussian) distribution. In such case, $\varepsilon_t \sim (\mu, o^2)$ defined by the mean (μ) and the standard deviation (o). In consequence, the complex traffic profile $Y(t;T)$ to be reproduced can be generated as:

$$Y(t;T) = f(t;T) + \varepsilon_t \qquad \text{(B-1)}$$

Let us now present three different traffic profiles, where $f(t;T)$ functions were generated using piecewise linear functions and $\varepsilon_t \sim \mathcal{N}(0, o^2)$ (Fig. B-1). Fig. B-1 presents three different cases of traffic generation where the mean function $f(t;T)$ and the random function ε_t are represented as strong blue line and red dots , respectively. Fig. B-1a plots a *business*-like traffic with differentiated traffic intensities during working and night hours; intensity experiences a significant increase around 9h reaching its maximum at midday. Fig. B-1(b) shows a similar traffic profile focused on the traffic generated by residential users; refer to this as *content delivery network* (CDN). It can be observed how the maximum value is around 17h. Finally, Fig. B-1(c), plots a totally different traffic profile reproducing the traffic exchange among servers in distant datacenters, e.g., for database synchronization; we call this profile as *DC2DC* traffic. The main characteristic of

this traffic is that the maximum value is reached during night, whereas day time intensity is virtually zero.

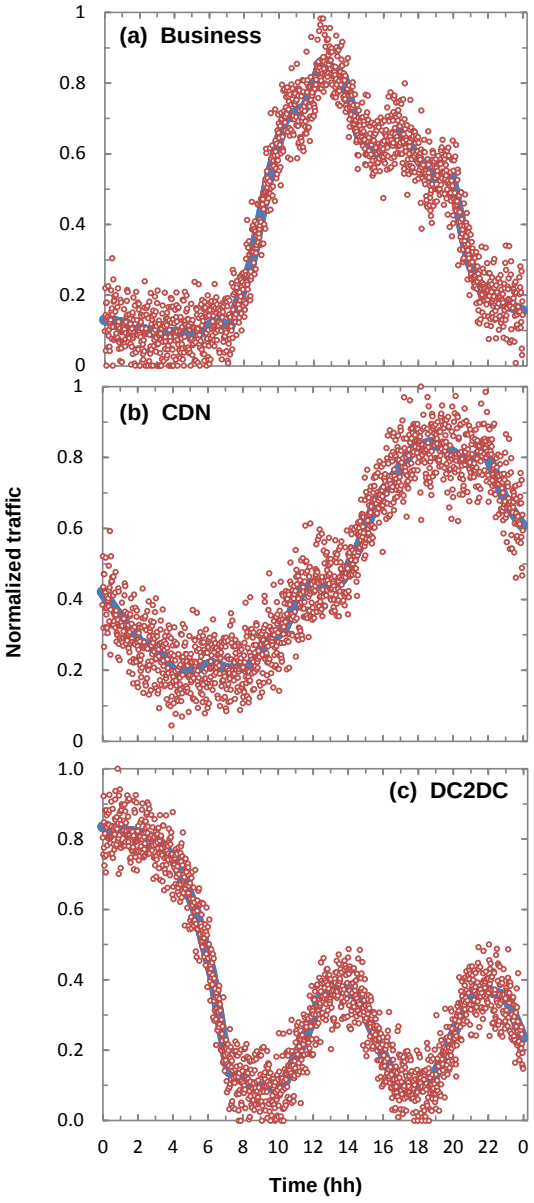


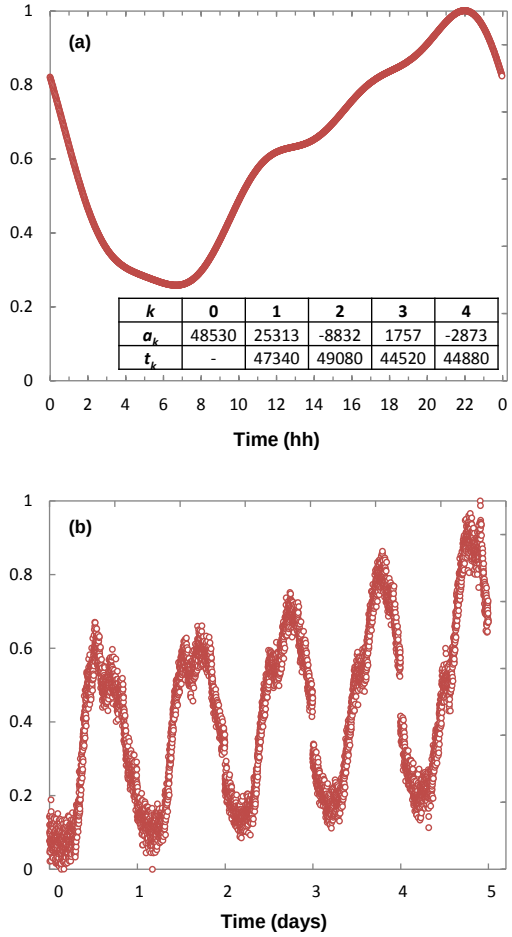
Fig. B-1. Traffic profiles generated: (a) Business, (b) CDN, and (c) DC2DC.

As anticipated above, $f(t;T)$ can be modeled as a summation of trigonometric sines. This alternative way has been used to model periodic traffic due to its repetitive pattern, e.g., in [STRONGEST10], where the authors reproduced traffic obtained from monitoring the Telecom Italia IP national backbone network. Daily profiles from Monday to Friday were very similar but also profiles on Saturday and on Sundays, hence only one model was developed to represent any day of the week (Fig. B-2a). Equation (B-2) reproduces the model, where k is the index for the

corresponding term in the sinusoidal series expansion, a_0 is the mean traffic intensity, a_k is the amplitude, t_k is the time shift in seconds, and daily period T (in seconds).

$$f(t;T) = a_0 + \sum_{k=1}^{K} a_k \cdot \sin\left[2 \cdot \pi \cdot k \frac{t - t_k}{T}\right] \qquad \text{(B-2)}$$

The curve in Fig. B-2a represents the traffic intensity generated with eq. (B-2), applying the values of the coefficients in the table. The equation is limited to the first four terms of the harmonic expansion since higher order terms add hardly any information.



k	0	1	2	3	4
a_k	48530	25313	-8832	1757	-2873
t_k	-	47340	49080	44520	44880

Fig. B-2. (a) Daily network traffic generated as a summation of trigonometric sines. Coefficients are specified. (b) Daily traffic variation with evolutionary traffic profile and incremental intensity.

The previous traffic profiles showed how periodicity can be modeled assuming that the profile observed in one day is similar for the rest of the days. However, traffic volume in usually growing when looking at long periods (e.g., one year) and even changes its daily pattern. Equation (B-3) presents a general formula to generate

evolutionary traffic profiles, where $\alpha(t)$ represents the line with value 1 for t=0 and 0 for t=*max_time*, and parameter β is the growing factor (e.g., 1.5). For instance, Fig. B-2b plots a traffic profile where $f_1(t)$ is the business profile and $f_2(t)$ is the CDN profile, previously presented.

$$Y(t;T) = \alpha(t)\cdot(f_1(t)+\varepsilon_{t1}) + (1-\alpha(t))\cdot(\beta\cdot f_2(t)+\varepsilon_{t2}) \quad \text{(B-3)}$$

Notwithstanding traffic generation is usually focused on modelling traffic, in this work, time evolutionary traffic profiles have also been covered.

List of Acronyms

AIC	Akaike Information Criterion
ALCOR	Anomaly and Network Reconfiguration
BANDO	BER Anomaly Detection
BER	Bit Error Rate
BN	Bayesian Networks
BV-WSS	Bandwidth Variable Wavelength Selective Switch
CASTOR	Cognitive Architecture to Support Telecom clOud Resource sharing
CDF	Cumulative Distribution Function
cFS	Cyclic Filter Shift
cLD	Cyclic Laser Drift
CR	Collected Data Repository
DCF	Dispersion Compensation Fiber
DoS	Denial-of-Service
DP-QPSK	Dual Polarization - Quadrature Phase Shift Keying
DSP	Digital Signal Processing
EDFA	Erbium Doped Fibre Amplifier
EON	Elastic Optical Networks
FEC	Forward Error Correction
FEELING	FailurE causE Localization for optIcal NetworkinG
FS	Filter Shift
FSE	Filter Shift Estimator
FT	Filter Tightening

FTE	Filter Tightening Estimator
KDD	Knowledge Discovery from Data
KPI	Key Performance Indicators
LD	Laser Drift
LDE	Laser Drift Estimator
LSP	Labeled Switched Path
LUCIDA	Failure Identification Algorithm
MDA	Monitoring and Data Analytics System
MILP	Mixed Integer Linear Programming
ML	Machine Learning
MMH	Maximum Margin Hyperplane
MPLS	Multi-Protocol Label Switching
MR	Modeled Data Repository
MTTR	Mean-Time-To-Repair
NMS	Network Management System
OA	Optical Amplifier
OAA	Observe-Analyze-Act
OD	Origin-Destination
ODEON	OD Traffic Anomaly-Triggered Topology Reconfiguration
OOK	On-Off Keying
OSA	Optical Spectrum Analyzer
OSC	Optical Supervisory Channel
OSNR	Optical Signal to Noise Ratio
OTC	Cost Optical Testing Channel
PCA	Principal Component Analysis
PDF	Probability Density Function
PM	Polarization Multiplexed
PMF	Probability Mass Function
PM-QPSK	Polarization-Multiplexed Quadrature Phase-Shift Keying
PRBS	Pseudo-Random Bit Sequence
QoS	Quality of Service

QoT	Quality of Transmission
QPSK	Quadrature Phase-Shift Keying
RMSA	Routing, Modulation And Spectrum Assignments
SDN	Software Defined Network
SLA	Service Level Agreements
SO	Signal Overlap
SQE	Signal Quality Estimation
SVM	Support Vector Machine
TCO	Total Cost Of Ownership
TISSUE	Testing optIcal Switching at connection SetUp timE
TP	Transponders
vlink	Virtual Link
VNT	Virtual Network Topology
VPI	VPIphotonics: Simulation Software and Design Services
WSS	Wavelength-Selective-Switches

List of References

[bAl10] E. Alpaydın, Introduction to Machine Learning, Second Edition, The MIT Press, 2010

[Bh14] M. Bhuryan, D. Bhattacharyya, and J. Kalita, "Network Anomaly Detection: Methods, Systems and Tools," IEEE Communications Surveys & Tutorials, vol. 16, pp. 303-336, 2014.

[Bi06] Ch. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, 2006.

[BluePlanet] Blue Planet Analytics, [On-line] Network Health Predictor. [on-line] http://www.blueplanet.com/ products/ analytics.html, 2017.

[Bo76] J. Boyd, "Destruction and Creation," U.S. Army Command and General Staff College, 1976.

[Ca12] A. Castro, L.Velasco, M. Ruiz, M. Klinkowski, J. P. Fernández-Palacios, D. Careglio, "Dynamic routing and spectrum (re)allocation in future flexgrid optical networks", Computer Networks, vol. 56, pp 2869-2883, 2012.

[Ca13] A. Castro, L. Velasco, J. Comellas, G. Junyent, "Dynamic Restoration in Multi-layer IP/MPLS-over-Flexgrid Networks," in Proc. IEEE International Conference on Design of Reliable Communication Networks (DRCN), 2013.

[Ca99] S. K. Card, J. Mackinlay, and B. Shneiderman, Readings in Information Visualization: Using Vision to Think, 1st ed. San Diego, CA, USA:Academic Press, 1999.

[CAIDA] CAIDA:Center for Applied Internet Data Analysis http://www.caida.org/research/

[Ch02] G. Chen, R. Lockhart, and M. Stephens, "Box-Cox transformations in linear models: Large sample theory and tests of normality," Canadian Journal of Statistics, vol. 30, pp. 177-209, 2002.

[Ch09] V. Chandola, A. Banerjee, and V. Kum "Anomaly Detection: A Survey," ACM Computing Surveys, vol. 41, pp. 1-72, 2009

[Ch15] K. Christodoulopoulos et al., "ORCHESTRA - Optical performance monitoring enabling flexible networking," in Proc. ICTON, 2015.

[Ch16] K. Christodoulopoulos, N. Sambo, E. Varvarigos, "Exploiting network kriging for fault localization," Optical Fiber Communications Conference and Exhibition (OFC), W1B.5, 2016.

[Ch83] V. Chvatal, *Linear Programming*, Ed. Freeman, 1983.

[Chi00] Ed. Chi, "A Taxonomy of Visualization Techniques using the Data State Reference Model ", IEEE 2000 (0-7695-0804-9/2000)

[Ciena] WaveLogic Ai, Ciena, http://www.ciena.com/products/wavelogic/ wavelogic-Ai/

[Cisco] "Cisco VNI Forecast and Methodology, 2015-2020", Cisco Systems, 2016

[Co07] G. Conti, "Security Data Visualization: Graphical Techniques for Network Analysis", ISBN-10:1-59327-143-3, No Starch Press 2007

[Co13] R. Corsini et al., "Blind Adaptive Chromatic Dispersion Compensation and Estimation for DSP-Based Coherent Optical Systems," IEEE/OSA Journal of Lightwave Technology, vol. 31, pp. 2131-2139, 2013.

[CPLEX] CPLEX, http://www-01.ibm.com/ software/ integration/ optimization/ cplex-optimizer/

[Cu13] F. Cugini et al., "Push-pull defragmentation without traffic disruption in flexible grid optical networks," IEEE/OSA Journal of Lightwave Technology, vol. 31, pp. 125-133, 2013.

[Cu15] A. Cuadra-Sanchez, J. Aracil, and J. Ramos de Santiago, "Proposal of a new information-theory based technique and analysis of traffic anomaly detection," International J. of Parallel Emergent and Distributed Systems, vol. 30, pp. 1-14, 2015.

[Cu89] McCullagh, Peter; Nelder, John (1989). Generalized Linear Models, Second Edition. Boca Raton: Chapman and Hall/CRC. ISBN 0-412-31760-5.

[Da16] M. Dallaglio, N. Sambo, J. Akhtar, F. Cugini, P. Castoldi, "YANG Model and NETCONF Protocol for Control and Management of Elastic Optical Networks," OFC 2016.

[DMR] https://en.wikibooks.org/wiki/Data_Mining_Algorithms_In_R/Clustering/K-Means

[Do15] Z. Dong et al., "Optical performance monitoring in DSP-based coherent optical systems," in Proc. OFC, 2015.

[Do16] Z. Dong, F. N. Khan, Q. Sui, K. Zhong, C. Lu and A. P. T. Lau, "Optical Performance Monitoring: A Review of Current and Future Technologies," IEEE/OSA Journal of Lightwave Technology, vol. 34, pp. 525-543, 2016.

[EON16] B. Mukherjee, V. López, L.Velasco, "Elastic Optical Networks; Architectures, Technologies, and Control", Springer, 2016

[Es17] J. Espinosa-Oviedo, *"Big Data Visualization",* PATC-BSC, 2017.

[Finisar] Finisar. Flexgrid High Resolution Optical Channel Monitor (OCM) [On-line] www.finisar.com, 2017.

[Ge12] O. Gerstel, M. Jinno, A. Lord, and S. J. Ben Yoo, "Elastic Optical Networking: A New Dawn for the Optical Layer?," in IEEE Communications Magazine, vol. 50, pp.S12-S20, 2012.

[Gh13] A. Ghazisaeidi et al., "Impact of tight optical filtering on the performance of 28 Gbaud Nyquist-WDM PDM-8QAM over 37.5 GHz grid," in Proc. OFC, 2013.

[Gu16] V. T. Guimaraes et al."A Survey on Information Visualization for Network and Service Management", IEEE Communications Surveys & Tutorials (vol. 18, pp 285 – 323, 2016)

[Hu11] G. Huang, A. Lall, C. Chuah, J. Xu, "Uncovering Global Icebergs in Distributed Streams: Results and Implications," Journal of Network and Systems Management, vol. 19, pp. 84-110, 2011.

[Ja13] S. Jawla, R.K.Singh, "Different Modulation Formats Used In Optical Communication System", IOSR Journal of Electronics and Communication Engineering (IOSR-JECE), vol. 8, pp 15-18, 2013

[Ka02] P. Kamath, K. Lan, and J.Heidemann, J. Bannister, and J.Touch "Generation of High Bandwidth Network Traffic Traces," Proc IEEE/ACM Internation Symnposium on Modeling, Analysis and Simulation of Computer Systems, 2002.

[Ke08] D. Keim, G. Andrienko, J. Fekete, C. Görg, J. Kohlhammer, G. Melancon, *"Visual Analitycs Definition, Process and Challenges*", HAL, 2008

[Kh15] P. Khodashenas, J. Rivas-Moscoso, B. Shariati, D. Marom, D. Klonidis, and I. Tomkos, "Investigation of spectrum granularity for performance optimization of flexible Nyquist-WDM-based optical networks," IEEE/OSA Journal of Lightwave Technology (JLT), vol. 33, pp. 4767–774, 2015.

[Ko16] U. Kozat, G. Liang, K. Kokten, J. Tapolcai, "On Optimal Topology Verification and Failure Localization for Software Defined Networks," IEEE/ACM Transactions on Networking, vol. 24, pp. 2899-2912, 2016.

[Ku00] R. Kuehl, "Design of Experiments", Thomson Learning, 2000.

[La04] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing Network-Wide Traffic Anomalies," in ACM SIGCOMM, 2004.

[La13] Machine Learning with R, Brett Lantz, Packt Publishing Ltd, 2nd Edition 2013

[Li16] C. Liu, M. Malboubi, and C-N. Chuah, "OpenMeasure: Adaptive Flow Measurement and Inference with Online Learning in SDN," IEEE Global Internet Symposium, 2016.

[M.2120] ITU-T Recommendation, "PDH path, section and transmission system and SDH path and multiplex section fault detection and localization procedures," M.2120, 2002.

[Ma00] C.Mas and P. Thiran, "An Efficient Algorithm for Locating Soft and Hard Failures in WDM Networks," IEEE Journal on Selected Areas in Communications, vol. 18, pp. 1900-1911, 2000.

[Ma05] C. Mas, I. Tomkos, and O. Tonguz, "Failure Location Algorithm for Transparent Optical Networks," IEEE Journal on Selected Areas in Communications, vol. 23, pp. 1508-1519. 2005.

[Ma12] F. Mata, J. García-Dorado, J. Aracil, "Detection of traffic changes in large-scale backbone networks: The case of the Spanish academic network," Elsevier Computer Networks, vol. 56, pp. 686–702, 2012.

[Mas14] E. Masala, A. Servetti, S. Basso, and J.C. De Martin, "Challenges and Issues on Collecting and Analyzing Large Volumes of Network Data Measurements," in New Trends in Databases and Information Systems, 2014.

[Me10] M. Medhat Gaber, A. Zaslavsky, S. Krishnaswamy, "Data Stream Mining," Data Mining and Knowledge Discovery Handbook", Springer, pp. 759-787, 2010.

[Mo17] F. Morales, M. Ruiz, Ll. Gifre, L. M. Contreras, V. López, and L. Velasco, "Virtual Network Topology Reconfiguration based on Big Data Analytics for Traffic Prediction," (Invited) IEEE/OSA Journal of Optical Communications and Networking (JOCN), vol. 9, pp. A35-A45, 2017.

[Na15] A. Napoli et al., "Next generation elastic optical networks: The vision of the European research project IDEALIST," IEEE Communications Magazine, vol. 53, pp. 152 162, 2015.

[Na16] Z. Nasralla, T. El-Gorashi, M. Musa, and J. Elmirghani., "Routing Post-Disaster Traffic Floods in Optical Core Networks," in Proc. ONDM, 2016.

[No11] Ali Norouzi, A.H. Zaim, and B. B. Ustundag, "An integrated survey in Optical Networks: Concepts, Components and Problems," IJCSNS, VOL.11 No.1, January 2011

[Og14] N. Ogino, H. Yokota, "Heuristic Computation Method for All-Optical Monitoring Trails Terminated at Specified Nodes," IEEE/OSA Journal of Lightwave Technology, vol. 32, pp. 467-482, 2014.

[OMNet] OMNet++, [Online] Available: http://www.omnetpp.org/

[Pa14] F. Paolucci, A. Castro, F. Cugini, L. Velasco, and P. Castoldi, "Multipath Restoration and Bitrate Squeezing in SDN-based Elastic Optical Networks," (Invited Paper) Springer Photonic Network Communications, vol. 28, pp. 45-57, 2014.

[Po12] P. Poggiolini, "The GN Model of Non-Linear Propagation in Uncompensated Coherent Optical Systems," *IEEE/OSA JLT*, 2012.

[Po17] Y. Pointurier, "Design of low-margin optical networks," IEEE/OSA Journal of Optical Communications and Networking, vol. 9, pp. A9-A17, 2017.

[Pr91] M. Priestley, Spectral Analysis and Time Series, Academic Press, 1991.

[Ra10] R.Ramaswami, K.N.Sivarajan, G. H. Sasaki, “Optical Networks; A Practical Perspective”, Third Edition, 2010

[RCran] https://cran.r-project.org/manuals.html

[RFC7011] B. Claise, B. Trammell, P. Aitken, “Specification of the IP Flow Information Export (IPFIX) Protocol,” IETF RFC 7011, 2013.

[RFC7491] D. King and A. Farrel, “A PCE-based Architecture for Application-based Network Operations,” IETF RFC 7491, 2015.

[RFC7536] M. Linsner, P. Eardley, and F. Sorensen, “Large-Scale Broadband Measurement Use Cases,” IETF RFC 7536, 2015.

[Ru16] M. Ruiz, M. Germán, L. M. Contreras, and L. Velasco, “Big Data-backed Video Distribution in the Telecom Cloud,” Elsevier Computer Communications, vol. 84, pp. 1-11, 2016.

[Sa11] N. Sambo et al., “Modeling and Distributed Provisioning in 10-40-100 Gb/s Multirate Wavelength Switched Optical Networks,” IEEE/OSA Journal of Lightwave Technology, vol. 29, pp. 1248-1257, 2011.

[Sa12] Y. Sakamaki, T. Kawai, T. Komukai, M. Fukutoku, and T. Kataoka, “Evaluation of optical filtering penalty in digital coherent detection system,” IEICE Communications Express, vol. 1, pp. 54-59, 2012.

[Sa16] N. Sambo, F. Cugini, A. Sgambelluri, and P. Castoldi, “Monitoring plane architecture and OAM Handler,” IEEE/OSA Journal of Lightwave Technology, vol. 34, pp. 1939-1945, 2016.

[Sh96] B. Shneiderman, “Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations”, IEEE 1996 (0-8186-7469-5/96)

[Sm08] Alex Smola and S.V.N. Vishwanathan, “Introduction to Machine Learning”, Cambridge University Press, 2008

[So07] A. Soule, F. Silveira, H. Ringberg, and C. Diot, “Challenging the supremacy of traffic matrices in anomaly detection,” in Proc. ACM SIGMCOMM, 2007.

[So17] P. Soumplis, K. Christodoulopoulos, M. Quagliotti, A. Pagano, E. Varvarigos, “Actual Margins Algorithm for Multiperiod Planning”, in Proc. OFC 2017.

[Sr06] N. Sreenath and P. Ramaswamy, “Virtual Topology Reconfiguration for Link Failures in IP-over-WDM Networks,” Proceedings of the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies (ICNICONSMCL’06)

[STRONGEST 10] STRONGEST Project, Deliverable 2.1, “STRONGEST: Scalable, Tunable and Resilient Optical Networks Guaranteeing Extremly-high Speed Transport,” 2010.

[Ta12] J. Tapolcai, P.-H. Ho, L. Rónyai, and B. Wu, “Network-wide local unambiguous failure localization (NWL-UFL) via monitoring trails,” IEEE/ACM Transactions on Networking, vol. 20, pp. 1762-1773, 2012.

[Ta13] N. Tateishi, et al. “Method for visualizing information from large-scale carrier networks”, in Network Operations and Management Symposium (APNOMS), 2013 15th AsiaPacific, Sept 2013.

[Ta14] J. Tapolcai, P. Ho, P. Babarczi, L. Ranyai, “On Signaling-Free Failure Dependent Restoration in All-Optical Mesh Networks,” IEEE/ACM Transactions on Networking, vol. 22, pp. 1067-1078, 2014.

[Ta15] J. Tapolcai et al., “Neighborhood Failure Localization in All-Optical Networks via Monitoring Trails,” in IEEE/ACM Transactions on Networking, vol. 23, pp. 1719-1728, 2015.

[TAnalytics] Nokia Networks, Telecom Analytics [On-line] http://networks.nokia.com/solutions/analytics/

[To18] M. Tornatore, “Data Analytics and Machine learning applied to Transport Layer”, W1D.2 OFC 2018.

[Uc16] V. Uceda, M. Rodriguez, J. Ramos, J. Garcia-Dorado, J. Aracil, “Selective capping of packet payloads at multi-Gb/s rates,” IEEE Journal on Selected Areas in Communications, vol. 34, pp. 1807-1818, 2016.

[Va14] A. Varet, and N.Larrieu, “How to generate realistic network traffic”, IEEE COMPSAC, 38th Annual International Computers, Software & Applications Conference, 2014.

[Ve14.1] L. Velasco, A. Castro, M. Ruiz, and G. Junyent, “Solving Routing and Spectrum Allocation Related Optimization Problems: from Off-Line to In-Operation Flexgrid Network Planning,” (Invited Tutorial) IEEE/OSA Journal of Lightwave Technology (JLT), vol. 32, pp. 2780-2795, 2014.

[Ve14.2] L. Velasco, D. King, O. Gerstel, R. Casellas, A. Castro, and V. López, “In-Operation Network Planning,” IEEE Communications Magazine, vol. 52, pp. 52-60, 2014.

[Ve15] L. Velasco, L.M. Contreras, G. Ferraris, A. Stavdas, F. Cugini, M. Wiegand, and J. P. Fernández-Palacios, “A Service-Oriented Hybrid Access Network and Cloud Architecture,” IEEE Communications Magazine, vol. 53, pp. 159-165, 2015.

[VPI] www.vpiphotonics.com/

[Wa04] C. Ware, “Information Visualization; Perception for design”, 2nd Edition, ELSEVIER, 2004

[We05] Y. Wen, V. W. S. Chan, L.Zheng, “Efficient Fault-Diagnosis Algorithms for All-Optical WDM Networks With Probabilistic Link Failures,” IEEE Journal of Lightwave Technology, vol. 23, pp. 3358-3371, 2005.

[wrp] https://www.rp-photonics.com/optical_heterodyne_detection.html

[Wu09] B. Wu, P. Ho and K. L. Yeung, “Monitoring Trail: On Fast Link Failure Localization in All-Optical WDM Mesh Networks,” IEEE/OSA Journal of Lightwave Technology, vol. 27, pp. 4175-4185, 2009.

[XLi16] X. Li, W. Ji, S. Zhang, K. Huang, “Signaling free localization of link and node failures in an optical mesh-tree network,” IEEE/OSA Journal of Optical Communications and Networking, vol. 8, pp. 263-271, 2016.

[Ze06] H. Zeng, C. Huang, and A. Vukovic, “A novel fault detection and localization scheme for mesh all-optical networks based on monitoring-cycles,” Photonic Network Communications, vol. 11, pp. 277-286, 2006.

[Zh16] Z. Zhong et al, “On QoS-assured degraded provisioning in service differentiated elastic optical networks,” GLOBECOM, 2016.

[Zh17] L. Zheng, D. Liu, W. Liu, Z. Liu, Z. Li, T. Wu, “A Data Streaming Algorithm for Detection of Superpoints with Small Memory Consumption,” in IEEE Communications Letters, vol. PP, pp.1-1, 2017.