SDN Workload Balancing and QoE Control in Next Generation Network Infrastructures

Sapienza Università di Roma

Facoltà di Ingegneria dell'Informazione, Informatica e Statistica

Dipartimento di Ingegneria Informatica, Automatica e Gestionale Antonio Ruberti

Dottorato di Ricerca in Automatica, Bioingegneria e Ricerca Operativa

Curriculum in Automatica

XXX Ciclo – Triennio 2014-2017

Candidato

Federico Cimorelli

Matricola 1245432

Tutor

Prof. Francesco Delli Priscoli

Coordinatore

Prof. Salvatore Monaco

PhD thesis, Sapienza University on Rome
Version: 10 January 2018
Author email: cimorelli@diag.uniroma1.it – cimorellifederico@gmail.com

Abstract

The increasing demand of bandwidth, low latency and reliability, even in mobile scenarios, has pushed the evolution of the networking technologies to satisfy new requirements of innovative services. Flexible orchestration of network resources is increasingly being investigated by the research community and by the service operator companies as a mean to easily deploy new remunerative services while reducing capital expenditures and operating expenses. In this regard, the Future Internet initiatives are expected to improve state of the art technologies by developing new orchestrating platforms based on the most prominent enabling technologies, namely, Software Defined Network (SDN) orchestrated Network Function Virtualization (NFV) infrastructure. After introducing the fundamental of the Next Generation Network, formalized as the conceptual Future Internet Platform architecture, the reference scenarios and the proposed control frameworks are given. The thesis discusses the design of two resources management framework of such architecture, targeted, respectively, (i) at the balancing of SDN Control traffic at the network core and (ii) at the user Quality of Experience (QoE) evaluation and control at the network edge. Regarding the first framework, to address the issues related with the adoption of a logically centralized but physically distributed SDN control plane, a discrete-time, distributed, non-cooperative load balancing algorithm is proposed, based on game theory and converged to a specific equilibrium known as Wardrop equilibrium. Regarding the QoE framework, a cognitive approach is presented, aimed at controlling the Quality of Experience (QoE) of the end users by closing the loop between the provided QoS and the user experience feedbacks parameters. QoE Management functionalities are aimed at approaching the desired QoE level exploiting a mathematical model and methodology to identify a set of QoE profiles and an optimal and adaptive control strategy based on a Reinforcement Learning algorithm. For both the proposed solutions, simulation and proof-of-concept implementation results are presented and discussed, to highlight the correctness and the effectiveness of the proposed solutions.

Keywords- Software Defined Networks, Load balancing, Wardrop equilibrium, Quality of Experience (QoE); User profiling, Reinforcement Learning.

Acknowledgment

I gratefully acknowledge my supervisor Prof. Francesco Delli Priscoli, for the opportunity he gave me to research and work on the thesis topics during the last four years, and in particular for the opportunity to work in national and European projects taking advantage from the collaboration with expert from both industry and academic sectors. I further thank the coordinator of the PhD program Prof. Salvatore Monaco, Professors and PhD colleagues, in particular, Prof. Antonio Pietrabissa, Dr. Letterio Zuccaro and Dr. Raffaele Gambuti for all the fruitful discussions on the main topics reported in this thesis.

I further thank all my colleagues at the Network Control Lab, Vincenzo, Silvia, Francesco, Martina, Federico, Alessandro and Lorenzo for their valuable support during these years.

Rome, December 2017
Federico Cimorelli

Contents

List of Figure

List of Table

Chapter 1
Introduction

This document reports an overview on research activities carried out by the author within the doctoral program ABRO (Automatica, Bioengineering and Operative Research) undertaken at the University of Rome "La Sapienza", Department of Computer, Control and Management Engineering "Antonio Ruberti".

The thesis deals with the application of control algorithms for the transparent, efficient, fair and flexible control of network resources aiming at the fulfilment of the network and users' requirements within Future Internet infrastructures, and within the Software Defined Network (SDN) enabled Network Function Virtualization infrastructures (SDN-NFV).

The results presented in this document include both personal and team research results of the author. A collaboration with industry experts has been exploited through the involvement in National research Project (PLATINO [1]) and EU funded H2020 Projects (FIWARE [2] and its follow-on FICORE [2], and T-NOVA [3]).

1.1. Motivation

A process aimed at interconnecting everything by means of software interfaces has characterized the ICT evolution process. The widespread of interconnected devices with increasing capabilities of storage, processing and transmitting data, span off several novel research areas and related business opportunities. Internet of Things (IoT), Big Data, Future Internet, 5G Networks, etc. are the leaves of the same technology tree having internet as its root.

Enterprises and carriers have found themselves in need of evolving their network infrastructures to satisfy new requirements. Such requirements comprise, among others, an even higher demand of bandwidth and of responsiveness to new data patterns (including machine-to-machine intra-data centre and in-mobility traffic), the ability to scale IT resources and share the same infrastructure among different logically isolated networks, vendor-agnostic tools and applications, and the ability to apply network-wide policies. The lesson learnt in the last decades, is that any internet-related service succeeds if and only if the users are willing to need and desire to use it.

There is an urgent need to rethink the network architectures for rendering them more efficient and to provide carriers, operators and service providers with innovative mechanisms and tools to allow them to easily deploy their services while reducing both capital expenditures and operating costs.

In addition, network service providers are becoming increasingly aware of the importance of customer experience in a more and more competitive market, especially since service quality has started replacing tariffing as the key selling point.

Hence, resources management framework that help network service providers gain a comprehensive view of the end-user experience (together with means and methods for improving it) while adopting a fair exploitation of the network infrastructure resources are keys for their business.

1.2. Objective and Contribution

The thesis objective deals with the design of transparent, efficient, fair and flexible mechanisms for controlling network resources within the context of the most prominent network paradigms and their enabling technologies, in particular, Software Defined Network (SDN) and Network Function Virtualization (NFV) infrastructure.

The scope of this thesis is twofold:

(i) at the network core, on the logical infrastructure orchestration layer, addresses the issues related with the adoption of a logically centralized but physically distributed SDN control plane by proposing a discrete-time, distributed, non-cooperative load balancing algorithm based on game theory and converged to a specific equilibrium known as Wardrop equilibrium;

(ii) on the other hand, at the network edge, a cognitive framework in proposed, aimed at controlling the Quality of Experience (QoE) of the end users by closing the loop between the provided QoS and the user experience feedbacks parameters. QoE Management functionalities are aimed at approaching the desired QoE level exploiting a mathematical model and methodology to identify a set of QoE profiles and an optimal and adaptive control strategy implemented by properly selected User Agents embedding a Reinforcement Learning based algorithm.

1.3. Outline of the Thesis

The thesis is organized as follows.

The next document section reports the list of the contributing research publications produced by the author during the doctoral program.

Chapter 2 reports a concise overview on the context of the thesis. The current identified limitations, the research trends and initiative aiming at overcoming them along with the potential enabling technology were discussed in a concise way.

Chapter 3 presents a novel workload balancing algorithm based on mean-field game theory, proved to converge to a Wardrop equilibrium and aimed at dynamically balancing the control traffic coming from the OpenFlow switches among the SDN Controllers to avoid congestion while improving the plant performance. Both the results of an evaluation of the benefits of the distributed approach and of the balancing control algorithm is presented and discussed through a proof-of-concept implementation of a reference SDN scenario.

Chapter 4 presents a cognitive architecture supporting QoE management, within orchestration algorithms which take control decisions, aiming at the satisfaction of the user/application requirements, and mechanisms to transparently enforce such decisions into the underlying heterogeneous telecommunication networks and cloud infrastructures.

Chapter 5 discusses the conclusion of the work, with an overview on the ongoing and future works.

1.4. Contributing Publications

The contents presented in *Chapter 3* and *Chapter 4* are the main results of the research work performed during the PhD program. Such contents were disseminated through the following publications:

- *"An Approach Based on Reinforcement Learning for Quality of Experience (QoE) Control".* F. Cimorelli, M. Panfili, S. Battilotti, F. Delli Priscoli, C. Gori Giorgi, S. Monaco". Proceedings of the 18th International Conference on Computer (part of CSCC '14). Santorini Island, Greece, July 2014.
- *"A Future Internet interface to control programmable networks".* S. Battilotti, F. Cimorelli, R. Cusani, F. Delli Priscoli, C. Gori, V. Suraci and L. Zuccaro. 23th Mediterranean Conference on Control and Automation (MED), 2015. IEEE.
- *"Profiled QoE based network controller".* S. Canale, F. Cimorelli, F. Facchinei, R. Gambuti, L. Palagi, V. Suraci. 23th Mediterranean Conference on Control and Automation (MED), 2015. IEEE.
- *"A Q-Learning based approach to Quality of Experience control in cognitive Future Internet networks".* L. Ricciardi Celsi, S. Battilotti, F. Cimorelli, C. Gori S. Monaco, M. Panfili, V. Suraci, F. Delli Priscoli. 23th Mediterranean Conference on Control and Automation (MED), 2015. IEEE.
- *"Distributed control in virtualized networks".* L. Zuccaro, F. Cimorelli, F. Delli Priscoli, C.G. Giorgi, S. Monaco, V. Suraci. Procedia Computer Science 56, 276-283. Year 2015.
- *"A distributed load balancing algorithm for the control plane in software defined network".* Federico Cimorelli; Francesco Delli Priscoli; Antonio Pietrabissa; Lorenzo Ricciardi Celsi; Vincenzo Suraci; Letterio Zuccaro. 24th Mediterranean Conference on Control and Automation (MED), Year: 2016, Pages: 1033-1040, DOI:10.1109/MED.2016.7535946, IEEE Conference Publications.
- *"Lyapunov-based design of a distributed Wardrop load balancing algorithm with application to Software Defined Networking".* A. Pietrabissa, L. Ricciardi Celsi, F. Cimorelli, V. Suraci, F. Delli Priscoli, A. Di Giorgio, and S. Monaco. IEEE Transactions on Control Systems Technology. Year 2018 [submitted, under review process].

In addition, thanks to the involvement into collaborative national and European research projects, many contribution flowed to project deliverables, of which some publicly accessible through the project's websites (MIUR PLATINO [1], FIWARE [2], T-NOVA [3]).

Chapter 2

Next Generation Computer Networks

This second chapter provides an overview on the context of application of the thesis. At the beginning the context of work is presented within the identified limitations; then the Future Internet architecture concept -in charge of overcome these limitation- is introduced and its main innovative approaches discussed. In the last the key enabling technologies are shortly reviewed.

2.1. Future Internet Initiatives

The Future Internet [4] general terms indicate the research initiative on new architecture/platform of telecommunication network with the aims of overcomes the nowadays limitation and supports the development of new innovative services and application that will benefit our society.

As reported by the "H2020 EU Framework Programme for Research and Innovation" [5], such initiatives:

- address the limitations of an Internet not designed to support the very large set of requirements imposed by an ever more diversified usage;
- support the advent of more efficient computational and data management models responding to the challenges posed by increased device/object connectivity and data-intensive applications;
- leverage the Internet to foster innovative usages of social and economic value also benefiting from the geospatial capabilities of the Future Internet.

From a functional point of view, the aforementioned Future Internet platform is in charge of meeting two different entities, the actors and the resources, by means of dedicated applications. An actor represents the entity whom requirement fulfillment is the main goal of the Future Internet.

For example, an actor could be an individual user, a content prosumer, an app developer, a network operator, a service provider or a cloud owner. A resource represents any entity that can be exploited to satisfy the actors' needs.

Examples of resources include, but are not limited to, services, contents, terminals, devices, functionalities, storage, computation, connectivity or networking capabilities. An application is any means used by the actors to exploit the available resources with the aim of fulfilling their requirements. Social networking, context-aware services, on-line games, interactive multimedia services, cloud storage and processing, collaborative services or automation services are examples of applications.

The internet evolution will promote those solutions where applications transparently, efficiently and flexibly exploit the available resources while satisfying the expectations of the involved actors.

To enable such Future Internet paradigm, a seamless access to control and manage the underlying technologies is crucial; However, the traditional approaches to model the internet architecture - layered and hierarchical - are all somehow limited, since they intrinsically tend to organize the internet infrastructure into rigid schemes. This slow down the evolution process towards a vendor independent and agnostic infrastructure.

The following limitation (among others) were identified:

1. A first limitation is represented by the traditional multi-layered architecture which forces the network designer to keep algorithms and procedures, lying at different layers, independent of each other. This greatly simplifies the overall design of telecommunication networks, since the network control problem is consequently decoupled in a certain number of much simpler sub-problems. Nevertheless, an obvious limitation of this approach derives from the fact that algorithms and procedures are poorly coordinated, thus impairing the efficiency of the overall network control strategy. The issues above claim for stronger coordination between algorithms and procedures dealing with different tasks.

2. A second limitation derives from the large variety of heterogeneous users, as well as from the large variety of heterogeneous underlying networks and cloud infrastructures which have been developed according to heterogeneous technologies and therefore embed technology-dependent algorithms and procedures. In this respect, the requirement of virtualizing such networks and users so that they can be dealt with in a homogeneous way by the applications claims for the design of a technology-independent, virtualized framework.

3. A third limitation derives from the fact that, at present, most of the algorithms and procedures embedded in telecommunication networks and cloud infrastructures are open-loop, i.e., they are based on some off-line reasonable estimation of the network variables (such as the offered traffic), rather than on real-time measurements of such variables. This limitation is becoming a large obstacle, since the behaviour of telecommunication networks, due to the large variety of supported services and the rapid evolution of the service characteristics, is becoming more and more unpredictable. This claims for an evolution towards advanced closed-loop methodologies which can cope with the dynamic and unpredictable behaviour of the considered scenario and can exploit appropriate real-time network measurements. The current technology developments which offer cheap and powerful sensing capabilities favour this kind of evolution.

4. A fourth limitation derives from the inability to satisfy personalized QoE requirements. The International Telecommunication Union (ITU-T) defines QoE as: The overall acceptability of an application or service, as perceived subjectively by the end-user [6]. As a matter of fact, most of the current approaches are based on the presence of a limited number of Classes of Service. Each Class of Service (CoS) provides specific performance guarantees (e.g., in terms of QoS): then, each connection is statically mapped on the most appropriate CoS and can only avail itself of the guarantees relevant to the selected CoS. Nevertheless, the requirement to satisfy a larger and larger number of applications, as well as to meet (even for the same application) personalized user expectations, implies that CoS assignment and resource management be handled in a more dynamic and personalized way.

A first attempt to face these challenges has been done by the Future Internet Public Private Partnership initiatives (FI-PPP) [6] supported by the European Commission, and by related projects such as, among others, FIWARE [2] and its follow-on FICORE [2], and T-NOVA [3]. Also at national level, the Italian MIUR PLATINO [1] project initiative addresses these challenges. Those projects are the major actors which tried to address the issues raised by the design of the so-called Future Internet.

2.2. Future Internet Architecture Concept

Shared objective of all the research initiative are the design of a transparent, efficient, fair and flexible convergent layer of network resource management by exploiting the virtualization of the resources while implementing a vendor independent and agnostic infrastructure. As shown in the previous sections, internet architecture has several limitations that slow down this evolution process. The traditional approaches to model the internet architecture (e.g. layered or hierarchical) are all somehow limited, since they intrinsically tend to organize the internet infrastructure into rigid schemes. There is the need to abstract from traditional architectural models, trying to be more agnostic as possible.

The virtualization of the underlying technologies will allow the Future Internet to boost its performance and to have a unique access to the available resources through an orchestration layer (see Figure 1). The proposed internet architectural concept can be easily mapped onto the domain of control of 5G network resources. In this respect, the requirement of virtualizing the resources, to ease their management, requires the introduction of a convergent-layer between the resources and the applications.

A valid solution to implement a convergence-layer is to use a virtualization framework. In the specific case of network and cloud resources, the join use of SDN and NFV represents the best candidate to implement the virtualization layer. SDN offers a feasible solution to virtualize the basic, per-flow, monitor and control network functionalities. NFV offers more complex network functionalities for the data management & analytics and for the resource configuration & control. The conjunct use of SDN and NFV allows to overcome all the aforementioned current internet limitations.

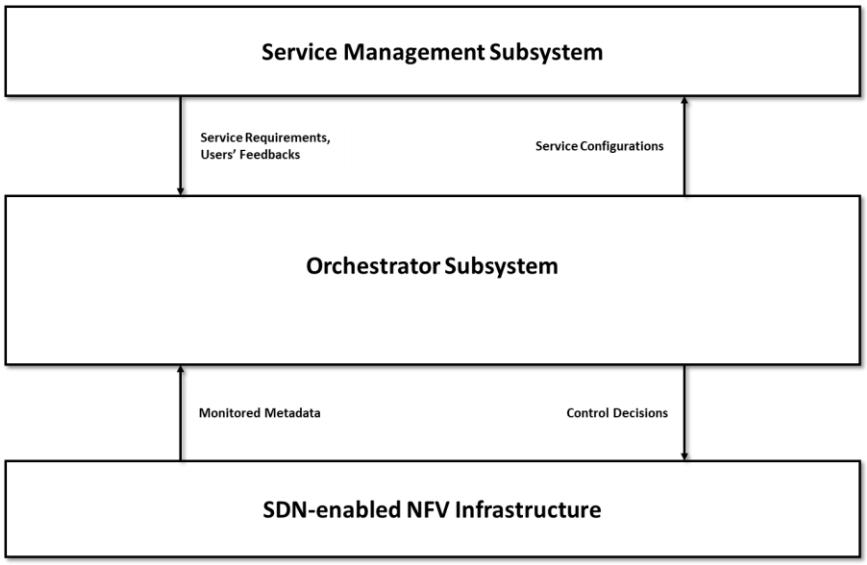

Figure 1 Future Internet Convergent Layers

The architecture shown in Figure 2 evolved to clearly define how actors can exploit TLC and IT resources through proper applications, it consists of three layers:

a) The SDN-enabled NFV (Network Functions Virtualization) Layer, composed of both the SDN-enabled NFV Infrastructure Management Subsystem and the SDN-enabled NFV Infrastructure. This layer includes the underlying telecommunication networks (ranging from Wi-Fi to LTE and even 5G networks) as well as the underlying cloud infrastructures.

b) The Orchestration Layer, composed of the Context Engine and Knowledge Database, the Data Analytics and QoE Evaluation/Control Subsystems and the Resource Control and Notification Subsystem.

c) The Service Management Layer. From now on, whenever we use the word "service," we will also mean "application" and vice versa.

The listed subsystems establish a double closed-loop feedback control system, thus ensuring a double degree of cognition. The functionalities exposed by each block can be implemented by means of distributed hardware or software agents embedded in network nodes such as: mobile terminals, base stations, backhaul nodes, core network devices, etc.

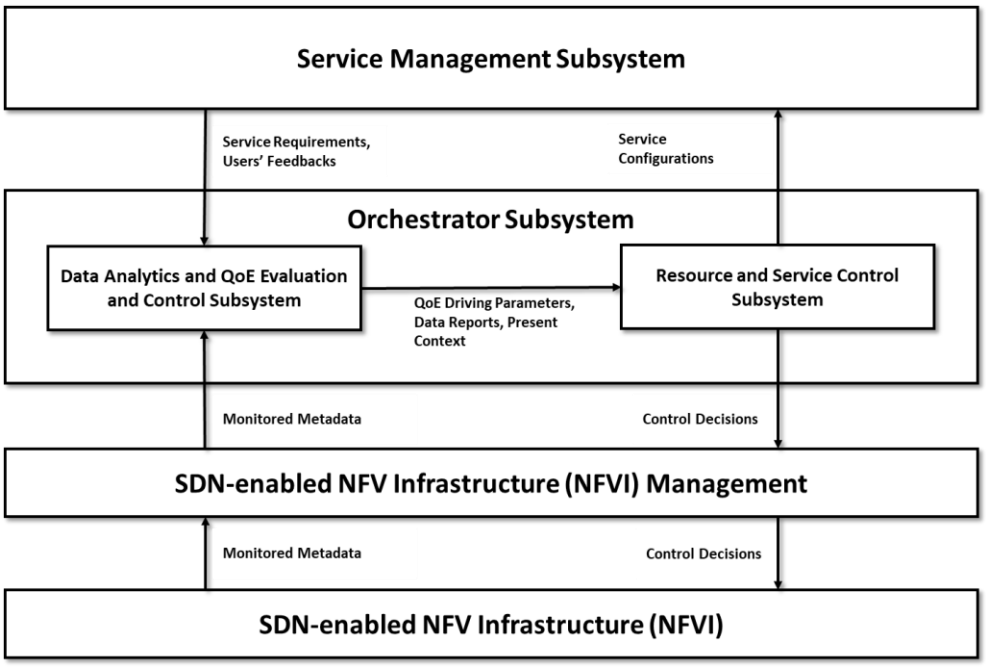

Figure 2 Future Internet: Architecture Concept

The first layer plays two fundamental roles, namely a monitoring role and an actuation one.

About the monitoring role, the SDN-NFV Layer must monitor the network resources of the underlying telecommunication networks, as well as the computing and store resources of the underlying infrastructures. The overall monitoring objective consists in the collection and preliminary processing of the available multi-layer, multi-network, heterogeneous and technology-dependent information (available in terms of network performance, measurements relevant to end-to-end service transactions, network tomography reports, etc.), as well as its translation into homogeneous and technology-independent metadata, namely the so-called Monitored Metadata, which should express the current network status, thus producing a key valuable feedback input for all of the control functionalities embedded in the Orchestrator.

About the actuation role, the SDN-NFV Layer has to put in place the Control Decisions, taken by the Orchestrator and related to the management of the network, computing and storage resources. This actuation role includes the conversion and adaptation of said technology-independent Control Decisions into technology-dependent Actuation Commands, which are enforced in the underlying telecommunication networks and cloud infrastructures. The Orchestration Layer includes the cognitive features of the overall architecture.

The Data Analytics and QoE Evaluation/Control consist of a set of cooperative, technology independent algorithms and procedures which oversee the formal description of the Monitored Metadata from the SDN-NFV Infrastructure as well as the metadata relevant to the Service Parameters and the Users' Feedbacks from the Service Management Layer, as well as of the proper aggregation of these metadata to form a multi-layer, multi-network Present Context. In addition, this module is in charge of assessing the so-called Perceived QoE, i.e., the QoE that is currently being perceived by the user and at the satisfaction of the personalized QoE requirements, namely the Target QoE by the computation, in real-time, of proper QoE Driving Parameters, namely personalized performance target values, which will then be exploited in order to ensure the desired minimization of the QoE Error associated with each service.

Such inputs are exploited by the Resource Control and Notification Subsystem in order to provide technology-independent control functionalities in charge of making appropriate coordinated and technology-neutral Control Decisions which will then affect the underlying network infrastructures, as well as of producing automatic Service Notifications associated with the detection of network/service/computing anomalies due to security problems or faults.

Even the Service Management Layer includes monitoring and actuation functionalities. As for the monitoring role, it is in charge of identifying the characterizing parameters of each service, of translating them into proper metadata and of providing them to the Orchestration functionalities.

Furthermore, the Service Management Layer should put in place mechanisms for user feedback monitoring, as well as to convert such feedbacks into metadata to be provided to the Orchestrator. As for the actuation role, the Service Management Layer should dynamically manage the provision of each service to the requesting users and, driven by the Service Notifications provided by the Orchestrator, to ensure at service level the most suitable security and fault recovery provisions – e.g., by means of a Hybrid Intrusion Detection System (HIDS).

2.3. Enabling Technologies

As explained in the previous sections, the virtualization of the storage, networking and computing resources represents nowadays the main approach to build a convergence-layer able to overcomes the heterogeneity of the access and transport networks.

With this respect, and in the area network and cloud resources management, a promising technology is the join use of Software Defined Network (SDN) (enabled by the OpenFlow protocol) and Network Function Virtualization (NFV). They represent the best candidate technologies.

In the next section, such new network paradigms will be shortly reviewed.

2.3.1. Software Defined Network (SDN)

Traditional networks present a coupled control and data plane, i.e. a network comprises a set of elements – routers, switches – that exchange information to build a view of the visible network in each node.

Each of these devices have its own Control Plane integrated for its functionalities like MAC learning of forwarding tables building and its own Data Plane in change of forwarding traffic according with its configuration (see Figure 3). This mean that there isn't a single point a control access to the network, each network device has its own control plane and an operator has to push a configuration on a per-device basis through SSH, SNMP, etc.

As defined by the Open Network Foundation (ONF), "Software Defined Network (SDN), is a network architecture where network control is decoupled from forwarding into the devices and is directly programmable" (see Figure 5) [7].

While the so-called forwarding plane (in charge of physically routing the data packets) resides in the network nodes, i.e., in the switches, all the intelligence related to network control is logically centralized into a single software entity called Software Defined Network (SDN) Controller, responsible for the network behaviour (see Figure 4).

In short, the SDN Controller is in charge of computing the routes of the packets and it is directly programmable from the upper layer (i.e., the application plane) through programmable interfaces.

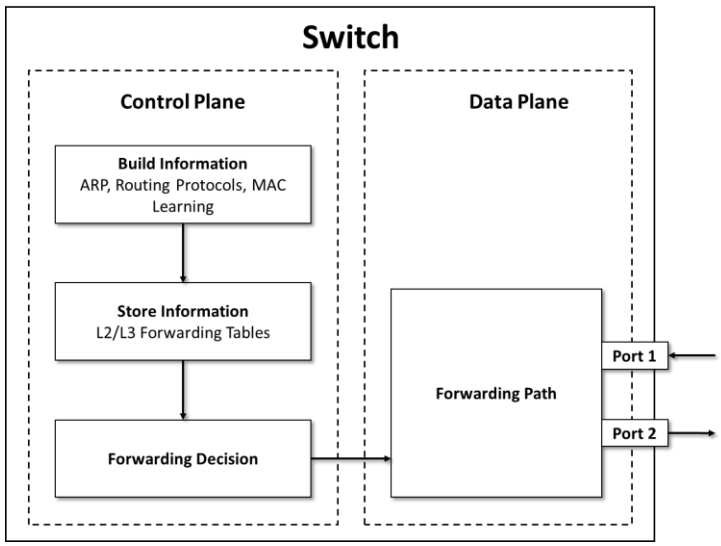

Figure 3 Traditional Switch Architecture

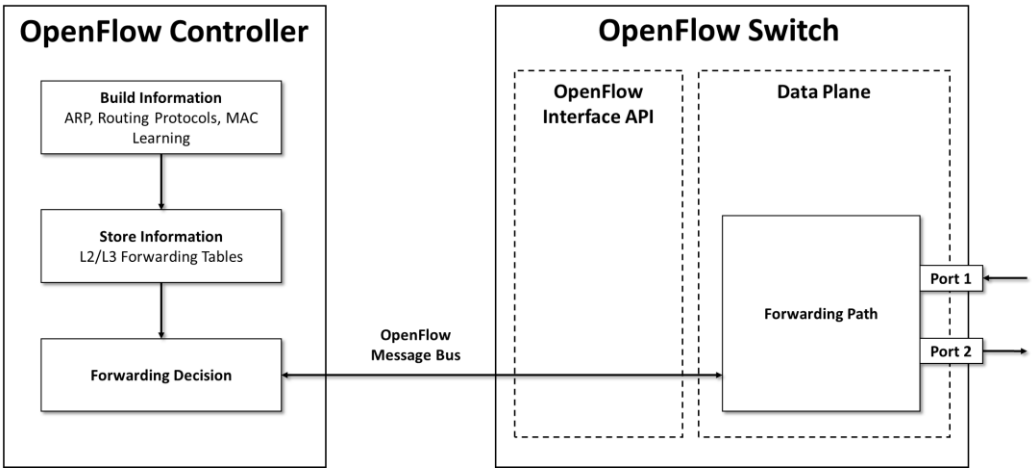

Figure 4 Externally Controller Switch Architecture

The SDN Controller maintains a global view on the underlying forwarding plane and makes it directly programmable from the upper layer (i.e., the application plane) through programmable interfaces. The interfaces are then used by applications and services to manage the nodes' forwarding behaviour while abstracting all the heterogeneity and the complexity of the network plant. SDN acts as a network operating system by implementing control software that interacts with compliant clients (i.e. switches devices) using well-defined interfaces in order to allow the development of innovative network management applications, such as traffic steering, QoS and QoE control algorithms, ACL, etc.

A key concept introduced by the SDN, in addition to the decoupled Control and Data Plane, is the abstraction of the network resources through an open interface for network abstraction layers. This abstraction is implemented by the control plane. From such plane, a flow table configuration can be enforced into the underlying devices without directly connect to them using API. Also, an application can use API to communicate with the controller, and this latter takes care of all the details to enforce configuration into the network.

While it is possible to implement single purpose controllers, e.g. for L2 forwarding or routing, available SDN controller implementations typically provide an extendable software platform on top of which SDN applications may be developed and deployed. Such a controller framework offers easy to use (northbound) APIs to the functionality provided by the SDN substrate. Further, it may include helper functions that provide, for example topology discovery or flow statistics collection. As a result, an SDN controller may be regarded as a layer between the SDN substrate and the SDN application layer, which implements the logic for concrete network services. Typically, SDN controllers are executed on commodity server hardware. While conceptually SDN controllers are centralised, in real world deployments the controller functionality may be distributed across multiple devices to ensure scalability and failure resilience.

OpenDayLight [8] is currently the newest and largest SDN controller platform. It is backed by the Linux Foundation and developed by an industrial consortium, which includes Cisco, Juniper and IBM, among many others. OpenDayLight includes numerous functional modules which are interconnected by a common service abstraction layer. Further, OpenDayLight provides a flexible northbound interface using Representation State Transfer APIs (REST APIs), and includes support for the OpenStack cloud platform.

As showed in the next figure, a set of switches in controlled by an SDN Controller. This latter has the role of building and storing network-information, and making forwarding decisions to be enforced into the underlying network.

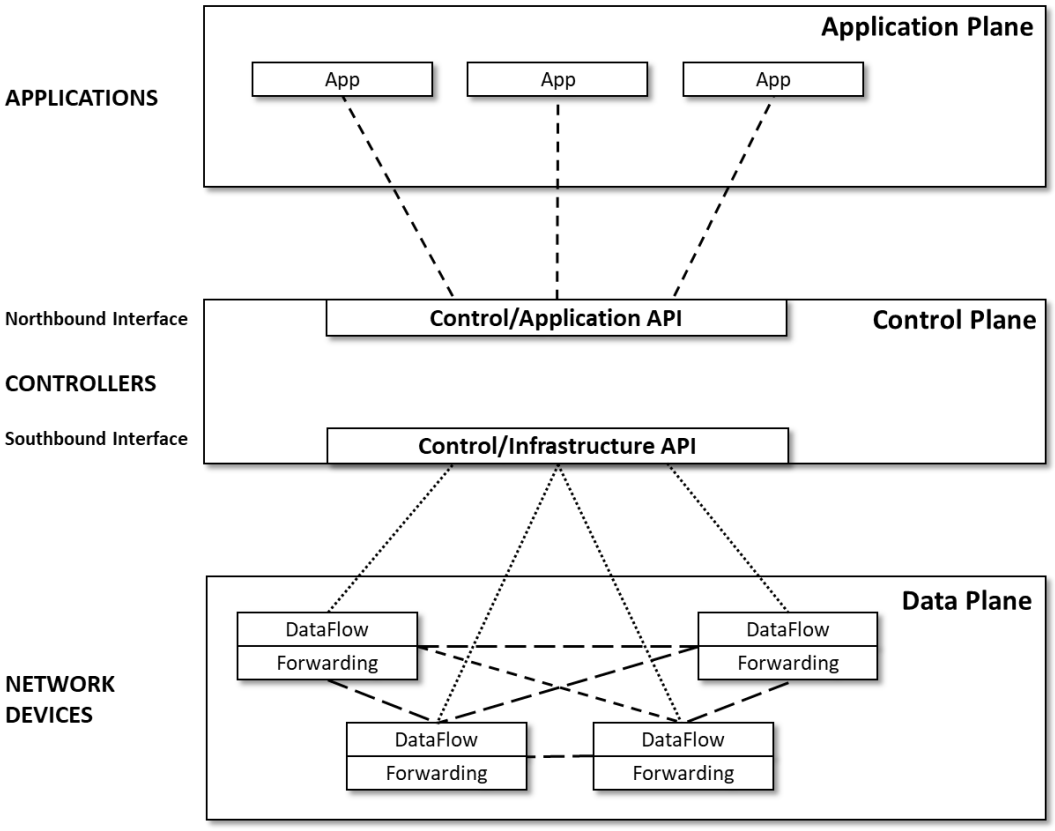

Figure 5 Software Defined Network, High Level Architecture

2.3.2. OpenFlow Protocol

The main SDN technology, provided by the ONF, is the OpenFlow standard [7]. OpenFlow is an open standard, defines an open protocol of communication among the forwarding plane (i.e., the compliant switches) and the control plane (i.e., the SDN Controller) enabling control and decoupling of the planes for higher functionalities and programmability. Since the OpenFlow protocol is an open standard supported and development by an open consortium within networking and services providers company

Within SDN and OpenFlow, a network switch has only to store the forwarding rules coming from the SDN controller and taking forwarding decisions according to them when a packet is coming on a port. Since forwarding rule computation is demanded to the SDN Controller, the switches have to no longer maintain context information to make forwarding decisions with consequent minor specs requirements in terms of CPU and memory.

As detailed in the OpenFlow Switch Specification document [9] and showed in Figure 6: "An OpenFlow Logical Switch consists of one or more flow tables and a group table, which perform packet lookups and forwarding, and one or more OpenFlow channels to an external controller. The switch communicates with the controller and the controller manages the switch via the OpenFlow switch protocol."

The main advantage of the OpenFlow protocol is that it is an open standard. It adoption enables the interoperability of switches produced by different vendors even in the case they are shipped with other proprietary function and interfaces.

OpenFlow allows to programmatically and remotely configure the forwarding rules and actions into the packets forwarding flow tables into the network elements. Such flow configuration is computed on-demand by the centralized SDN OpenFlow Controller and then enforced into the underlying network data plane (i.e. the network OpenFlow switches). The controller can then decide to modify existing flow configuration on one or more switches or to deploy new rules since it as an overall view on the network plant.

The OpenFlow protocol is based on the Transmission Control Protocol (TCP), and prescribes the use of Transport Layer Security (TLS) for security aspects. The common adopter TCP port by controllers for listening OpenFlow protocol messages is the port 6633.

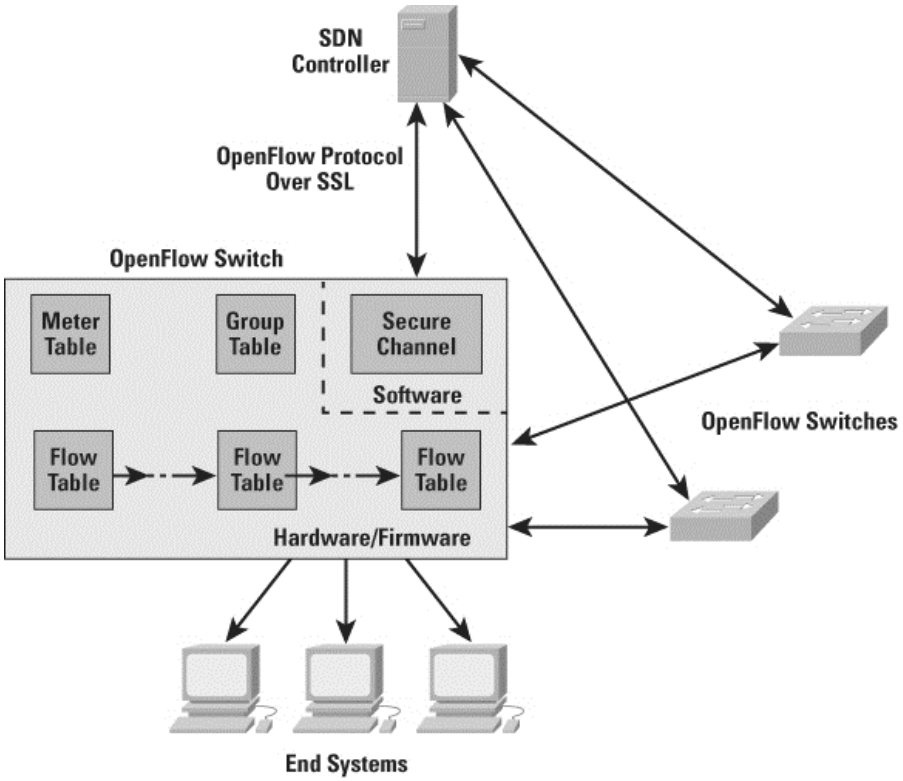

Figure 6 OpenFlow Switch Architecture

2.3.3. Network Function Virtualization (NFV) Infrastructure (NFVI)

As highlighted by the ETSI ISG NFV in its first white paper [10], the scenario which defines the situation faced by most network operators nowadays, relates to the physical components of their networks, which are characterised using a wide range of proprietary hardware appliances. This problem of appliance and technology diversity continues to grow for operators as new equipment is added to previous generations of equipment in the network.

This leads to significant challenges related to the launch of new services, increasing energy costs and capital investments coupled with the difficulty of finding people with the most appropriate skills to handle the design, integration and operation of increasingly complex hardware-based appliances. In addition, the trend towards shorter operational lifespan of hardware also affects revenues, leading to situations where there is no return on investment or where there is no time for innovation.

Network Functions Virtualization (NFV) will address these challenges by leveraging standard Information Technology (IT) virtualization technology to consolidate various network equipment types onto industry standard high-volume servers, switches and storage located in Data Centers, Network Nodes and in the end user premises.

In this context, NFV refers to the virtualization of network functions carried out by specialized hardware devices and their migration to software-based appliances, which are deployed on top of commodity IT (including Cloud) infrastructures.

Virtualizing Network Functions potentially offers many benefits, including:

- Reduction in both equipment costs and power consumption,
- Reduced time to market,
- Availability of network appliances that support multiple-versions and multi-tenancy, with the ability to share resources across services,
- Targeted service introduction based on geography or customer type, where services can be quickly scaled up/down as required,
- Enabling a wide variety of eco-systems,
- Encouraging openness within the ecosystem.

Software applications must be specifically designed or rewritten to run optimally in virtualized telecom environments to meet carrier grade requirements. Otherwise, applications ported to virtualized environments may experience significant performance issues and may not scale appropriately to the required network load.

An additional challenge for virtualization in a telecom network environment is the requirement to deliver low latency to handle real-time applications such as voice and video traffic. In addition to performance, other operational characteristics that are crucial to successful deployments include: maturity of the hypervisor; Reliability, Availability, and Serviceability (RAS); scalability, security, management and automation; support and maintainability.

Deploying NFV also incurs other well-defined risks, e.g. scalability in order to handle carrier network demands; management of both IT and network resources in support of network connectivity services and Network Functions (NFs) deployment; handling of network fault and management operations; Operations Supporting System (OSS) / Business Supporting System (BSS) backwards compatibility in migration situations; interoperability required to achieve end-to-end services offerings, including end-to-end Quality of Service (QoS). In addition, essential software appliances should achieve performance comparable to their hardware counterparts which is currently not always possible due a variety of reasons such as the performance of the virtualization technologies.

As explained in the previous sections: Cloud Manager technologies, Software Defined Network (SDN) and Network Function Virtualization (NFV) represent the most prominent innovative trend toward modernizing the IP Infrastructure. In particular:

- Cloud Manager technologies operate as cloud operating system in charge to virtualize and orchestrate a large pool of compute, storage and networking resources throughout a data center;
- SDN offers a centralization of the network orchestration and control; The SDN Controller relays information and orchestrates traffic on the network to switches and routers, and to the applications with so-called northbound Application APIs;
- NFV offers the virtualization of network service able to speed up the deployment of new network services to foster revenue and growth plans.

The integration of these technologies defines what the modern IP infrastructure looks like. Such components can be used to enable the transformation of the telecommunications network to simplify operations, administration, maintenance and provisioning and to lay the network foundation for new access technologies (e.g., 5G).

With a complementary approach to SDN, NFV provides innovation in services deployment and management. To exploit such technologies, several providers created the NFV ISG under the European Telecommunications Standards Institute (ETSI). The creation of ETSI NFV ISG resulted in the foundation of NFV's basic requirements and architecture.

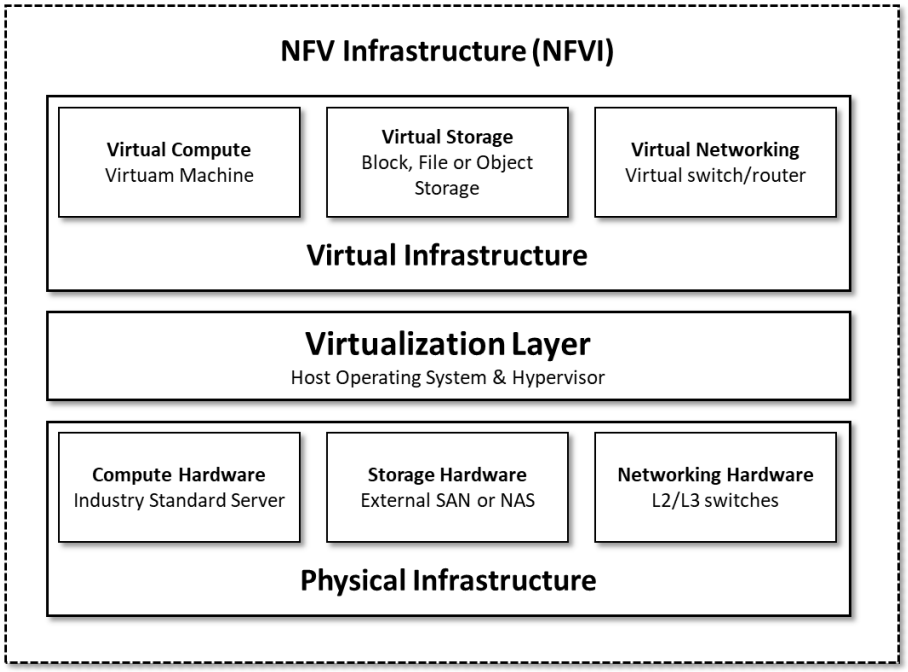

Figure 7 NFVI High Level Architecture

The ETSI Institute has defined the network functions virtualization management and orchestration (NFV-MANO) architecture, and its main blocks [11]:

- VIM manager, responsible for controlling and managing the NFV infrastructure (NFVI) compute, storage, and network resources;
- VNF manager, responsible for the lifecycle management of the NFV entities;
- NFV orchestrator. responsible for the management of new network services (NS) and virtual network function (VNF) packages; NS lifecycle management; global resource management; validation and authorization of network functions virtualization infrastructure (NFVI) resource requests.

These functional blocks help standardize the functions of virtual networking to increase interoperability of software-defined networking elements. Such SDN-NFV networking approach is a win-win strategy: from the manufacturers' point of view, the benefit is that compliant devices seamlessly work with all the other compliant devices and controllers; from the network operators' side, this reduces Capex and Opex costs by extending the life of the devices, by reducing the replacement costs and by speeding up the deployment of new services.

Anyway, such innovative architecture introduces some issues that must be properly addressed, mainly the pros and the constraints related to the adoption of a 'logically centralized but physically distributed' control plane.

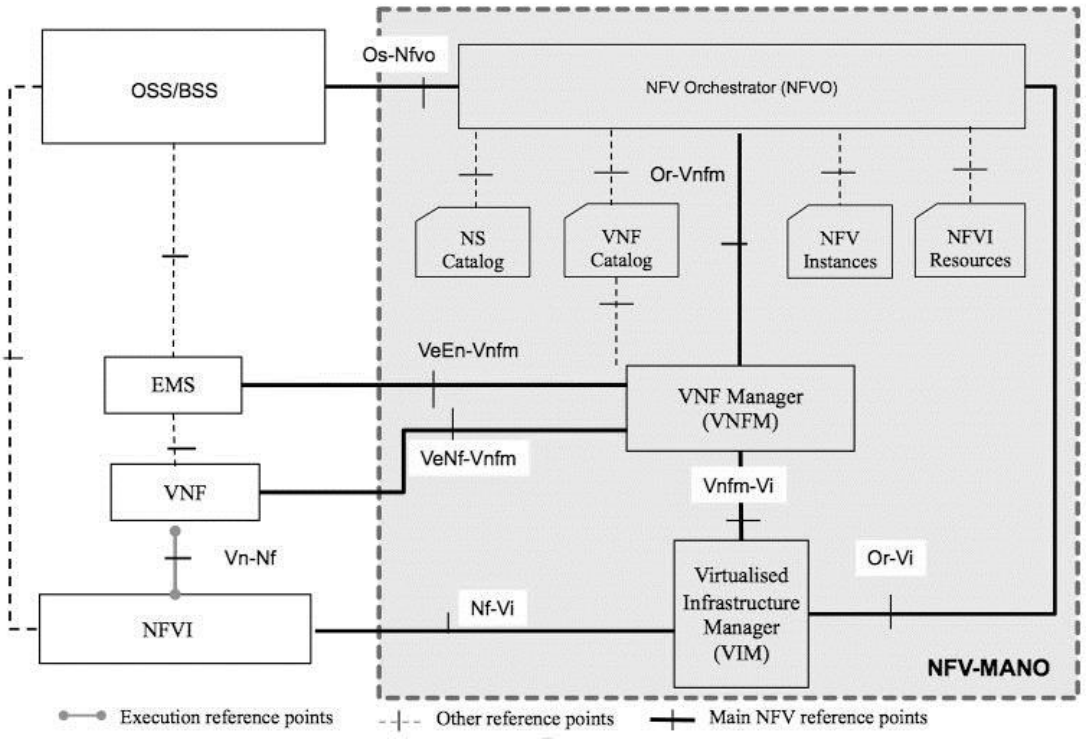

Figure 8 ETSI MANO Architecture

Chapter 3

Load Balancing a Distributed SDN Control Plane

With reference to the Infrastructure layer of the Future Internet architecture presented in the previous section, this chapter addresses the limitation of the SDN infrastructure. An analysis of the issues is presented, then the scalability and reliability issues are addressed by proposing a distributed SDN Control Plane and applying on it a novel workload balancing algorithm. Such algorithm is an original discrete-time, distributed, non-cooperative load balancing algorithm, based on mean-field game theory, which does not require explicit communications. The algorithm is proved to converge to an arbitrarily small neighbourhood of a specific equilibrium among the loads of the providers, known as Wardrop equilibrium. Thanks to its characteristics, the algorithm is suitable for the Software Defined Networking (SDN) scenario, where service requests coming from the network nodes, i.e., the switches, are managed by the so-called SDN Controllers, playing the role of providers. The proposed approach is aimed at dynamically balancing the requests of the switches among the SDN Controllers to avoid congestion. Both the results of an evaluation of the benefits of the distributed approach and of the balancing control algorithm is presented and discussed through an implementation of the algorithm in a proof-of-concept SDN scenario.

3.1. Introduction

As defined by the Open Network Foundation (ONF), “SDN is a network architecture where network control is decoupled from forwarding and is directly programmable.” While the so-called forwarding plane (in charge of physically routing the data packets) resides in the network nodes, i.e., in the switches, all the intelligence related to network control is logically centralized into a single software entity called SDN Controller, responsible for the control of the network behaviour. In short, the SDN Controller is in charge of managing all network information, thus making, and enforcing into the underlying nodes, suitable forwarding decisions.

Taking advantage of the whole view of the network, a SDN network controller is capable for building and maintaining network information, and thus making forwarding decisions. Anyway, the adoption of the SDN-OpenFlow approach to massively scalable data centres need to be supported by a resilient and scalable architecture. When the network size grows, controller performances, with particular attention to throughput and latency metrics, are topics that must be carefully addressed in order to prevent the SDN Controller from becoming the performance bottleneck of the plant.

The issues related with the adoption of the SDN paradigm that must be addressed should be grouped in the following points below:

- *Scalability:* Since the network intelligence is shifted from the switches to the SDN Controller, any forwarding decision consists of a unit of workload for the controller machine and the management network. Since an OpenFlow switch not maintains enough network information to take forwarding decisions it sends request for per-packets-related decision to the SDN OpenFlow Controller and waits for a forwarding flow; this potentially produces millions of requests per second forwarded immediately by each switch without delay. For instance, according to [12], in highly dynamic scenarios (e.g., in a server cluster of 1.5k machines) the SDN workload has a median flow arrival rate of 100k flows per second. Handigol et al. [13] provides benchmark results of controllers' performance resulting in 30k flows per second supported with 10ms of install time. This raises scalability concerns since the performance of the controller could become the performance bottleneck of the plant.
- *Controller Placement:* the mapping between the forwarding plane and the control plane, i.e., the association of each switch to a given SDN Controller, could be static or dynamic. In the literature, the problem of associating switches and SDN Controllers is known as "SDN Controller placement problem," and the proposed algorithms are mostly based on the location of the SDN Controllers with respect to the switches – see, e.g., [14] presenting a comparison between brute-force and greedy algorithm solutions, [15] presenting a heuristic approach, and [16] solving the problem by adopting operational research theory. Heller et al. [17] presents a comparison of various placement metrics. Since, in real networks, the network workload varies in time, the main drawback of static mapping is the necessity to find a new solution to the controller placement problem whenever some SDN Controller workload exceeds its processing power (congestion occurrence).
- *Reliability:* The SDN architecture introduces the issue of a single-point of failure in the plant since in the SDN architecture the network intelligence is physically centralized in a single point (ie the SDN Controller machine). The SDN Controller acts as Network Operation System by programming the switches behavioural through open API exploiting L2 and L3 topologies and hosts information learnt from the OF switches. It is then subject to overload or failure issues, in addition the vertical scaling of the computational and memory capacity may not be enough in very large networks. The centralized design offers better performance in terms of throughput and latency [18], anyway it should evolve in order to form a cluster of machine following the concept of a "distributed, but logically centralized" SDN control plane [19] [20].

Since the network intelligence is shifted from the switches to the SDN Controller, each switch sends to the SDN Controller requests for forwarding decisions, each of which constitutes a unit of workload, or job. Although the SDN Controller can be vertically scaled, it can still saturate, thus becoming a relevant bottleneck for the network plant in terms of throughput and latency.

To overcome the scalability and reliability issues, the solution proposed in this thesis provides the adoption of a distributed SDN Control Plane against a single instance scenario. A pool of SDN Controllers can be arranged in a cluster to form a physically distributed but logically centralized Control Plane sharing the overall network information, workload and control effort. A drawback of such an approach resides in the fact that, in the currently adopted proximity-based approaches, each switch (and, consequently, its workload) is statically associated with the closest SDN Controller. However, this static approach is not effective, especially when the workload is not evenly spatially distributed and/or is dynamic (which is always true in communication networks).

Therefore, load balancing mechanisms and algorithms are required to dynamically allocate the workload among all the available SDN Controller instances with the overall aim of optimizing the network performances. In [21], a switch migration protocol was devised for this purpose, yet such an approach requires an advanced load estimation mechanism for both SDN Controllers and switches, and an intra-controller protocol to guarantee liveness and safety features.

Anyway, the limitation of the distributed Control Plane is that the mapping between the data plane (i.e. the network devices) and each instance belonging to the cluster, are static among the operation time. To address this problem, a novel balancing algorithm for the OpenFlow traffic and its reference design is proposed. It was designed to shift the workload across the cluster in order to improve the performance of the overall system.

In the following section, the details of such evaluation and solution are presented.

3.2. Distributed Control Plane

To overcomes the scalability limitation presented in the above section multiple controllers can be arranged to form a synchronized cluster of SDN Controllers, thus providing high availability and replication [21]. In this way, the controller virtualization may help in overcoming scalability and centralization issues, which affect the SDN controller performances in large data centres.

Therefore, even if the SDN Controller is logically a single centralized entity, it is realized through multiple software and hardware entities distributed across a cluster of multiple controllers, sharing the overall network information and control capabilities through a distributed data-store across all the cluster and providing a single point of access through APIs to the so-called "northbound" applications (i.e., the network applications as depicted in Figure 9).

With reference to the conceptual Future Internet architecture presented in Section 2.1, the following figure focuses on the high-level architecture of the SDN Control Plane designed to support deployments in a distributed scenario by exploiting the virtualization of the controller machine. In this regard, new functional elements have been identified. Their scope is to extend the well-known centralized SDN architecture, where each Control Plane (CP) block corresponds to a single instance of the control plane.

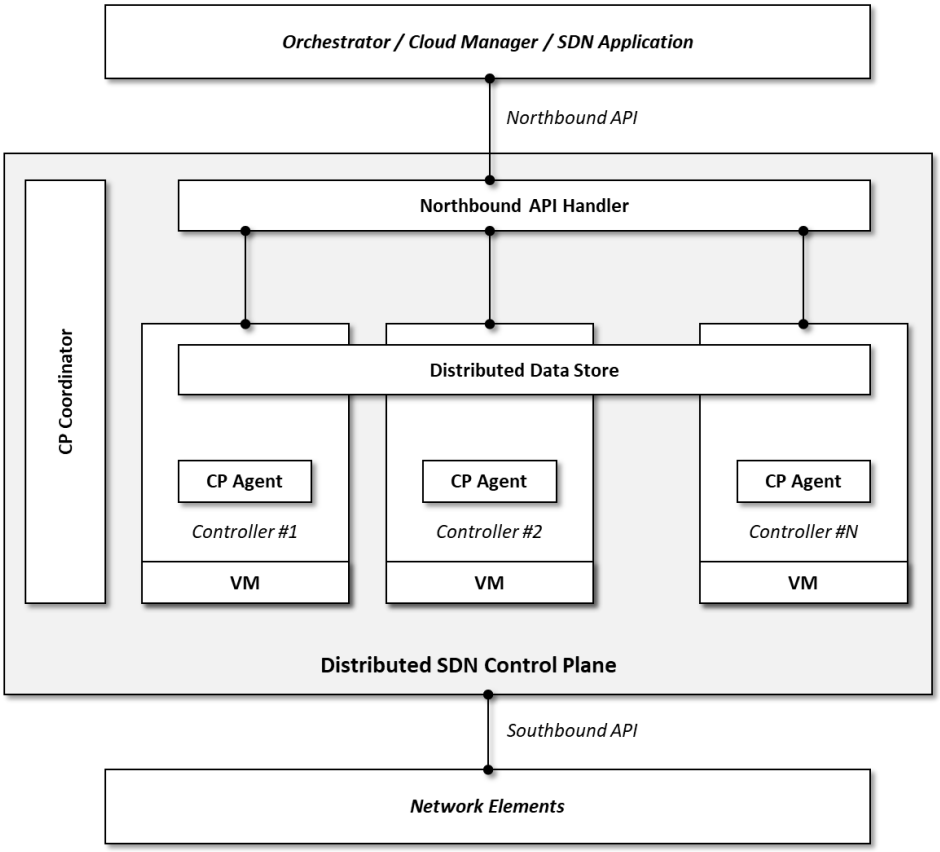

Figure 9 Distributed SDN Control Plane Architecture

Within the above figure, we can address the following components:

- Distributed Data Store: This component is responsible for consistently maintaining a global view of the network across the control plane instances belonging to the cluster. The collected information includes the topology and the state of the network, including switch, port, link, and host status. Northbound applications/internal CP components can take advantage of the global network view in making forwarding and policy decisions. It is a fact that relying on distributed data store may incur in delays for exchanging messages to maintain a consistent view of the whole network between all the instances of controller. In this respect, the innovation introduced by the virtualized SDN Control Plane is the capability to split, distribute and store the global network view into multiple caches. Each cache is responsible for maintaining the status of a portion of the network, through which the traffic of a set of NFV appliances is routed.
- Northbound Request Handler: Mainly responsible for distributing northbound requests among the available controller instances. It is essential to make the network control plane accessible by the northbound API as a unique single instance.
- CP Coordinator: The Control Plane Coordinator supervises and coordinates the operation in the cluster. Specifically, it has to (i) properly instruct the Northbound Requests Handler in spreading the northbound requests; (ii) dynamically configure the controller-to-switch mapping by connecting each switch to one or more controller instances; (iii) decide whether to add or remove a controller instance to the cluster depending on the network needs; (iv) monitor the workload on the cluster; (v) assign data caches to a subset of controller instances depending on the deployed NFV applications. The role of coordinator can be carried out by one of the CP instances available in the cluster, by means of a procedure of leader election.
- CP Agent: The CP Agent is in charge of collecting information about the current resource utilization (CPU load, memory usage, control messages arrival rate, etc.) at each CP instance and enforcing the switch-to-controller instance connection rules as established by the Coordinator. These rules are used by each switch to identify the controller instance/s to which the southbound requests must be forwarded.

From a technological point of view, the OpenFlow protocol, since version 1.3, has regulated its architectural model by defining the concept of SDN Controller role (equal/master/slave) for a switch. Each OF switch could maintain multiple OpenFlow channel with multiple OpenFlow controllers, each of them marked with a role that defines its behaviour. This enabled two modes of operation when multiple SDN Controllers exist in a network: master/slave interaction and equal interaction. In the master/slave interaction each switch can be associated with all the SDN Controllers but managed by only one (the master), responsible for all the events corresponding to that switch, whereas the others (slaves) are used as backup controllers. On the contrary, in equal interaction, each switch can be associated with multiple SDN Controllers and have more than one master association, that is, several equal associations.

On the SDN Controller side, Onix [19] was the first SDN Controller to implement a distributed SDN controller model. It originally targeted network virtualization in datacenters and remained closed-source.

More recently, OpenDaylight [8], an open-source project sponsored by a large consortium of networking companies, has introduced a cluster-based high-availability model to increase reliability and fault tolerance. Similarly, to OpenDaylight, ONOS [20] was designed to run on a cluster of servers for high availability, using a distributed data store to maintain the global view of the network.

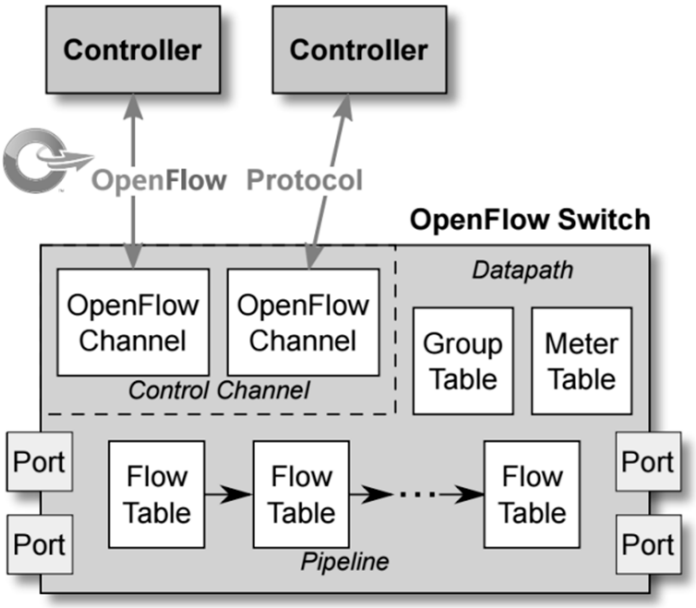

Figure 10 Components of OpenFlow Switch with Multiple Controllers' Connections

3.2.1. Benefit Evaluation

To evaluate the benefits of having a Distributed Control Plane across a cluster of SDN controllers, a testbed was deployed, and experimental tests performed in order to benchmark the proposed solution.

In the following text, the testbed is briefly described, and the performance evaluation results presented and discussed, for more details please refers to the contributing author publication "*Distributed control in virtualized networks*".

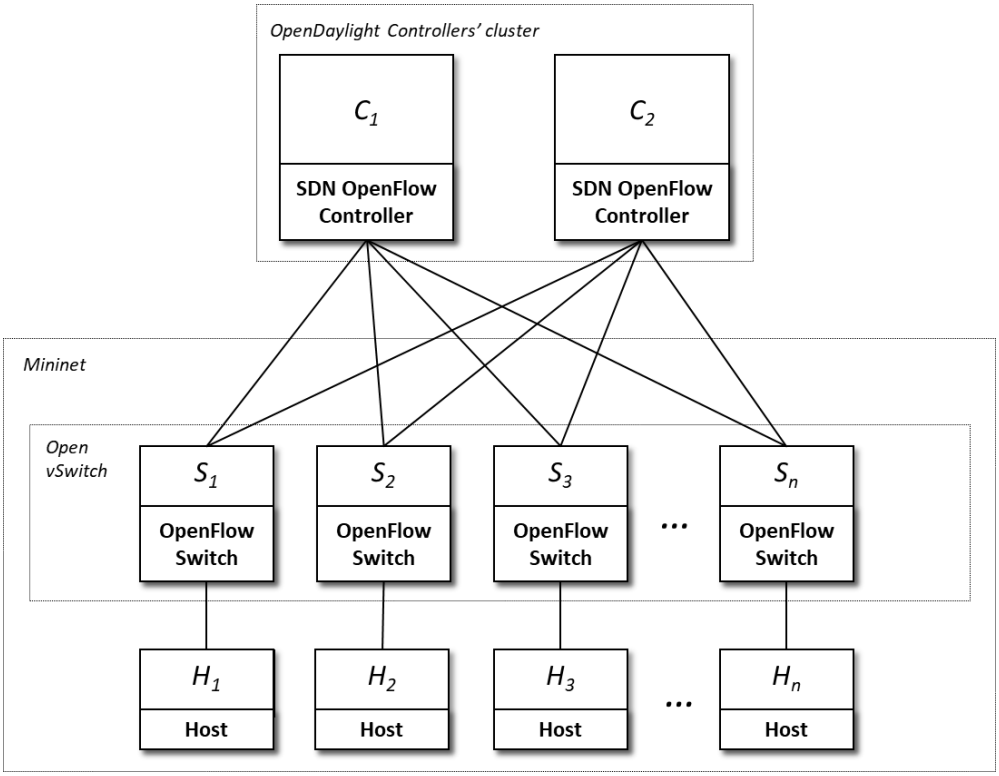

Figure 11 Testbed Architecture

To evaluate different distributed Control Plane scenario's performances, different configurations are deployed. A first test is performed to evaluate the benefit from adopting a Distributed Control Plane with a cluster of 2 controller instances against a single Controller scenario.

When multiple controllers are available a switch may connect to all. For example, a cluster with two controllers can be arranged in the following testing scenarios:

- *Single Connection*: all the switches are connected to one instance of Controller (see Figure 12);
- *All Connection:* all the switches are connected to all the instances of Controller (see Figure 13);
- *Selective Connection:* there are two Controllers and the first 13 switches are connected to the first one while the remaining ones are connected to the second Controller (see Figure 14).

The second scenario is the more interesting from a control point of view. Using the default implementation of the distributed control plane provided by the OpenDaylight we are evaluating the behaviour when every switch sends the packet-in control message to more than one controllers; just one controller should answer to the request while the other one should ignore it.

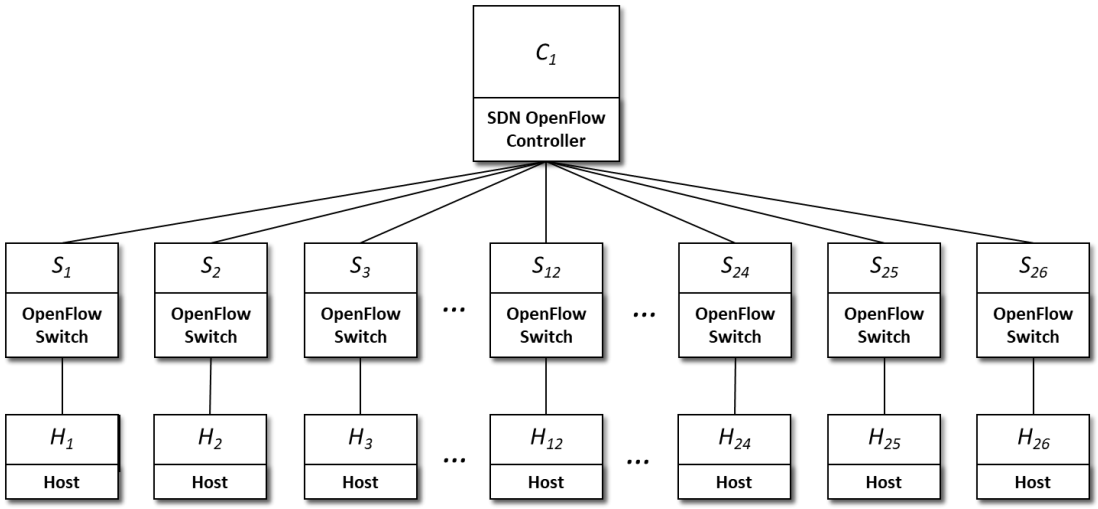

Figure 12 Single Connection Testing Scenario

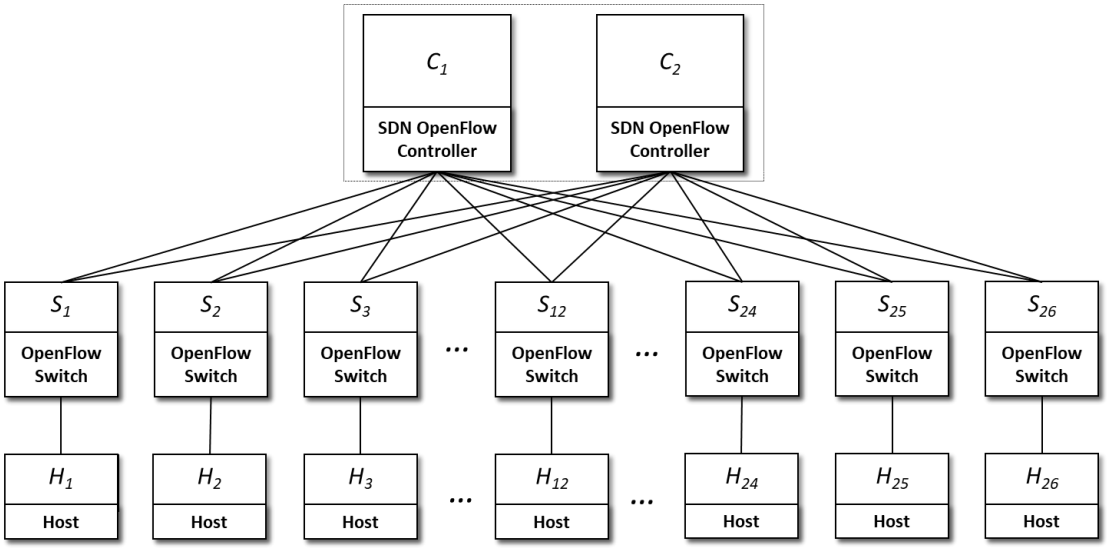

Figure 13 All Connection Testing Scenario

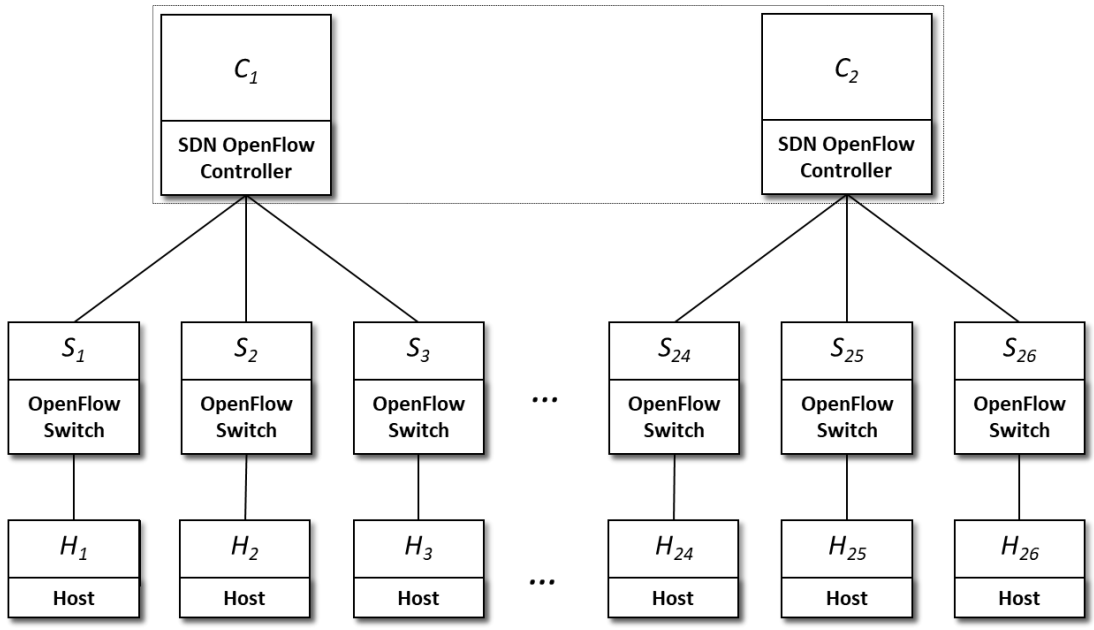

Figure 14 Selective Connection Testing Scenario

3.2.1.1. Testbed Setup

The following software components are integrated into such testbed:

- OpenDayLight [8], the SDN controller most adopted and supported by the industry and Open Source community;
- Mininet [22] a networking testing suite that provides Python APIs able to virtualize custom networks;
- Open Virtual Switch (ovs) [23], "a production quality, multi-layer virtual switch licensed under the open source Apache 2.0 license".
- Open vSwitch Database Management Protocol (OVSDB) [24] for the operation management of the virtual switches.

The testing environment (see Figure 11) is composed by a SDN controller cluster of 2 instances and 26 virtual switches. Every switch *Si* is connected to the *Si+1* switch and to the host *Hi* that incorporates a data traffic generator. To stimulate the switches to generator control traffic, each host generates ARP requests.

The control traffic is generated by the switch when there is no flow entry for an incoming data packet (the ARP request) that the host wants to send. Hence, the switch encapsulates the data packet in a control packet (Packet-In message) and sends it to its controllers. Then, exactly one controller should send a message containing the flow entry (the Flow-Mod message) that the switch will install in its table. The performance is then evaluated by counting the packet-in and flow-mod messages exchanged between the switched and the SDN Controllers.

3.2.1.2. Evaluation Results

Figure 15 depicts the 95-percentile response time of the 3 cases described above. As expected, the difference between a cluster with one controller and one with two is noticeable as the number of packet-in increases. The difference between the All connection and the Selective connection case becomes relevant when the number of packet-in/s is greater than 40,000. From 50,000 the last connection scheme clearly outperforms the first.

Similar results are reported in Figure 16. When the network load is high the throughput of the Selective test is the highest one, being 66% greater than the one in the All test scenario and 100% greater than the one in the Single test scenario.

The results clearly highlight the benefits of the multi-controller configuration, and particularly of the selective connection configuration. In the case of single connection, the unique controller instance saturates at approx. 20.000 pin/s. When adopting the all connection configuration, the two controller instances receive the requests from all the switches, which causes a performance saturation at approx. 30.000 pin/s, that is a +50% performances against the single connection configuration. It is worth to note that adopting the selective connection configuration, the performance saturation goes beyond the 40.000 pin/s, that is a +100% performance against the single connection configuration and a +33% performance against the all connection configuration).

We can explain these results considering that in single connection configuration we have half of the computation power, since just one instance of the controller is operative. Increasing the number of operative controllers, influences the performance, but the switch-to-controller association strategy makes the real difference in performance. Indeed, with an all connection configuration, even if there are two controllers operating at the same time, each controller receives requests from all the switches. In the selective connection configuration, each controller receives requests only from a subset of the switches, so each controller can handle a higher number of requests.

On the light of the above results, we can make some considerations:

- increasing the cluster size may improve the performance but this has relevant results only in congested situations, when the load of control plane requests saturates the single controller processing capabilities. The drawback of increasing the number of CP instances is the need of more computation power and related increased operational cost;
- the load of control plane is variable and often unpredictable, being strongly related to the amount of control messages exchanged between switches and controllers. Therefore, a dynamic mechanism able to keep balanced the traffic across the whole control plane is needed.

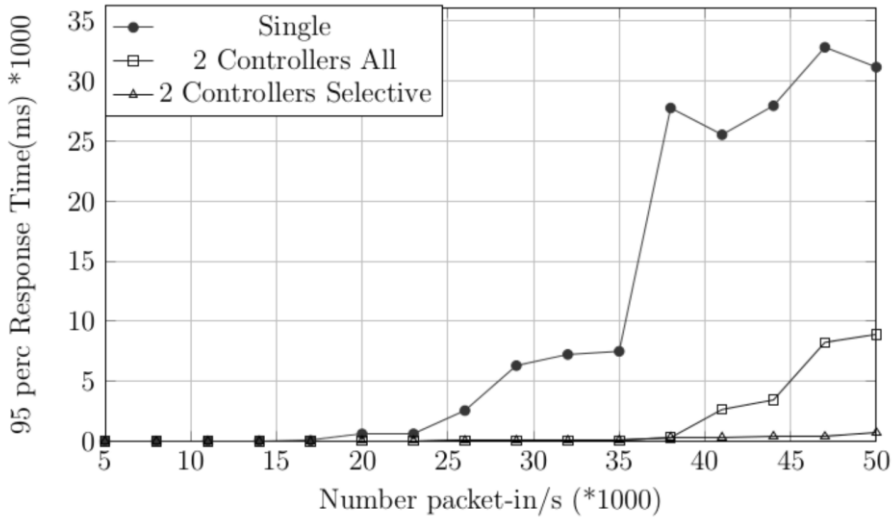

Figure 15 95-percentile response time comparison between the Single, All and Selective configurations

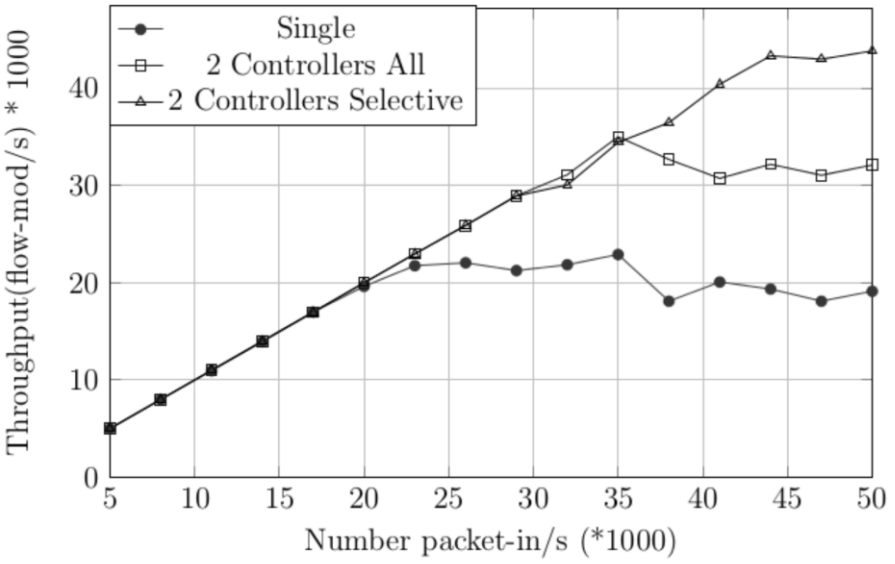

Figure 16 Throughput comparison between the Single, All and Selective configurations

3.3. Load balancing SDN Control Plane Algorithm

Even if adopting a distributed control plane produce benefit (as depicted in the previous section), a drawback of current SDN distributed Control Plane approaches resides in the fact that the mapping between the forwarding plane and the overall control plane (i.e. the association of each switch to a given SDN Controller into the cluster), is statically regulated, currently adopting a proximity-based approaches, each switch (and, consequently, its workload) is statically associated with the closest SDN Controller. However, this static approach is not effective, especially when the workload is not evenly spatially distributed and/or is dynamic (which is always true in communication networks). Since, in real networks, the network workload varies in time, the main drawback of static mapping is the scenario where some SDN Controller workload exceeds its processing power (congestion occurrence); with an impact on the control plane reactiveness, and consequently having degradation of network latencies. Based on real measurements from production data centers, we estimate that the peak-to-median ratio of flow arrival rates is almost 1-2 orders of magnitude1 (more details in Section 2. There are also some other traffic variations in spatial and temporal terms, depending on where are generating flows. Some switches could observe a larger number of flows generation which could results in overloading the corresponding (static over the time) controller.

Therefore, load balancing mechanisms and algorithms are required to dynamically allocate the workload among all the available SDN Controller instances with the overall aim of optimizing the network performances.

With respect to the problem of dynamically balance the SDN Control Plane traffic, in [25] the authors propose a Switch-Migration based algorithm, were at each switch is associated a load measure; the algorithm then uses defined thresholds to compute the load of every controller and migrate the control of a switch from an instance to another. Anyway, such approach poses some issues to be addressed in terms of liveness and safety during the migration policy and requires an advanced load estimation mechanism for both SDN Controllers and switches. In addition, this approach isn't transparent to the actual Control Plane (i.e. the SDN Controller software support).

To address those issues, the solution presented in this chapter consist of an original discrete-time, distributed, non-cooperative load balancing algorithm, based on mean-field game theory, which does not require explicit communications. The algorithm is proved to converge to an arbitrarily small neighborhood of a specific equilibrium among the loads of the providers, known as Wardrop equilibrium.

Thanks to its characteristics, the algorithm is suitable for the Software Defined Networking (SDN) scenario, where service requests coming from the network nodes, i.e., the switches, are managed by the so-called SDN Controllers, playing the role of providers. The proposed approach is aimed at dynamically balancing the requests of the switches among the SDN Controllers to avoid congestion. Such balancing is implemented in a new software entity named *SDN-Proxy* (refer to section 3.3.1) to be placed on the Control plane to act in a transparent way for the adjacent software interfaces.

Below, for the sake of clarity, a table containing the notations adopted with regards to the Load Balancer algorithm.

$C = \{1,2, \dots, c\}$	Set of commodities
$\lambda^i, \quad i = 1, \dots, c$	Per-commodity flow demand
$(s^i, d^i), \quad i = 1, \dots, c$	Commodity (source, destination) nodes V Set of providers
$x_p^i, \quad i \in C, p \in V$	Load of commodity i relying on provider p
$\boldsymbol{x} = \left(x_p^i\right), i \in C, p \in V$	Flow vector
$\mathcal{X}$	Feasible state space
$l_p(x_p)$	Latency of provider p with load x_p
$\mathcal{X}_{\mathcal{W},\varepsilon}$	ε-Wardrop equilibrium set
$\Phi(\boldsymbol{x}$	Beckmann, McGuire, and Winsten potential
$\mathcal{L}(\boldsymbol{x})$	Candidate Lyapunov function

Table 1 Load Balancer nomenclature

3.3.1. Reference Architecture

The OpenFlow protocol, since version 1.3, has regulated its architectural model by defining the concept of SDN Controller role for a switch. Switches should have multiple controller connection, each of them associated with a controller role (namely master, slave and equal role). This enabled two modes of operation when multiple SDN Controllers exist in a network: *master/slave* interaction and *equal* interaction.

In the *master/slave* interaction each switch can be associated with all the SDN Controllers but managed by only one (the *master*), responsible for all the events corresponding to that switch, whereas the others (*slaves*) are used as backup controllers. On the contrary, in *equal* interaction, each switch can be associated with multiple SDN Controllers and have more than one *master* association, that is, several *equal* associations.

In this work, a scenario where the SDN network works in *equal* interaction mode across a cluster of controllers is considered. A set of *n* switches can communicate with a set of different controllers. The proposed innovation is that the controllers do not have the task of deciding which switch to manage (i.e., which controller manages the request flow of each switch), but in our proposal the load-balancer decide which controller to use on a request-by-request basis; the decisions are taken without communications among the switches themselves.

To comply with the scope, a new entity is introduced, named *SDN-Proxy* (see Figure 17).

The switches are statically connected to the nearest SDN-Proxy. Each *SDN-Proxy* relates to all the instances in the SDN Controller cluster, receives the requests of its switches and has the task of forwarding them to one of the available SDN Controllers, based on a load balancing algorithm described in the following sections. Multiple entities of *SDN-Proxy*, each with its load balancing algorithm could be arranged in a large network scenario.

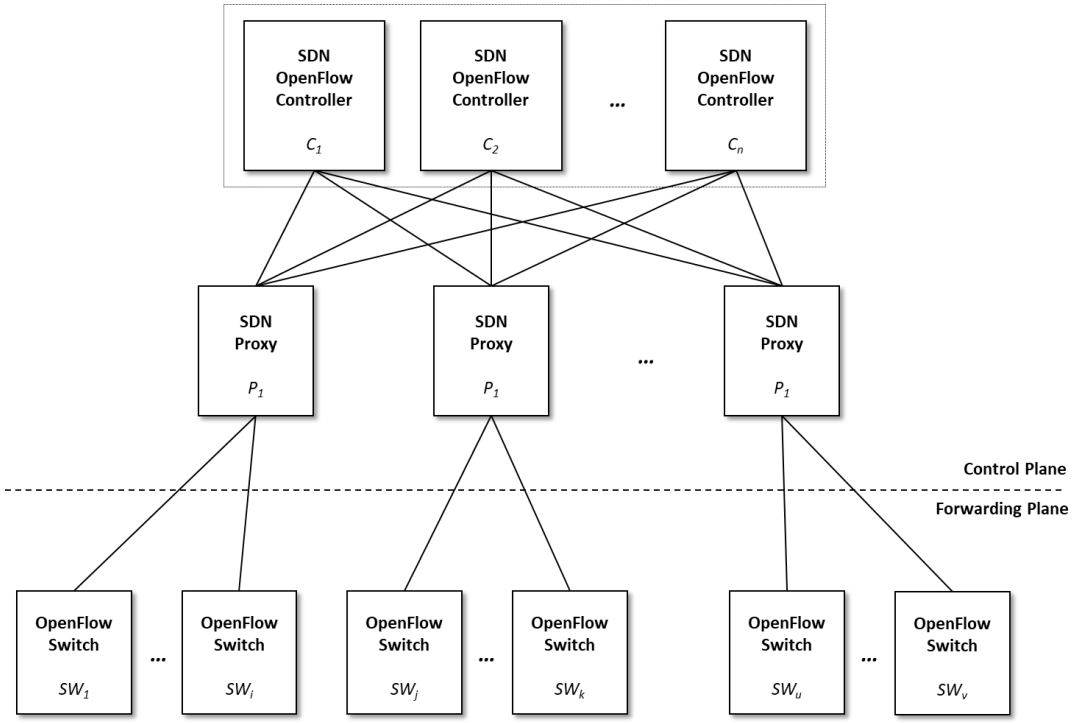

Figure 17 SDN Control Plane Load Balancer, Reference Architecture

3.3.2. State of the art and proposed innovation

Several dynamic load balancing approaches have been proposed in the literature: namely, in [26] it is suggested to classify load balancing algorithms as based on global, cooperative or non-cooperative approaches.

Global algorithms require that each node, by means of an extensive interconnected system, transmits its current state to a centralized load balancer, which judiciously assigns a job to each resource, while simultaneously optimizing a specific objective (e.g., the response time of the entire system over all jobs). This is a classical approach that has been studied extensively using different techniques (e.g., nonlinear optimization) until it has been outperformed by the other two above-mentioned approaches. In cooperative algorithms, several decision makers agree upon making a coordinated decision so that each one of them operates at its own optimum. Instead, non-cooperative algorithms entail the presence of several decision makers which optimize their own response time independently of the others, since cooperation is not allowed. In such a case, a Nash equilibrium condition is reached where no decision maker can receive any further benefit by changing its own decision unilaterally. In other words, the stability of the network under said algorithms is analysed in terms of reaching a load distribution in which no single job can move to any other node with a lesser number of jobs.

Furthermore, load balancing algorithms can be classified as either static or dynamic. Static load balancing relies on the available knowledge of the application load, whereas dynamic load balancing algorithms are required for settings where the load distribution is not known a priori and succeed in performing their decision-making process based on the current state of the system, which is generally made available via feedback.

From the large body of literature on load balancing, we recall [27] [28] as examples of centralized static cooperative load balancing, [29] [30] as examples of centralized static non-cooperative load balancing, [31] and [32] as examples of centralized dynamic load balancing, and we also recall [33], which, instead, addresses the problem of distributed dynamic load balancing relying upon local cooperation among neighbouring network nodes.

The scenario considered in this paper requires a non-cooperative dynamic load balancing approach. This kind of algorithms are widely investigated in game-theoretic frameworks, where the problem can be described as a dynamic load balancing game, in which users distribute their loads in a non-cooperative and selfish fashion [34] (in some applications, these algorithms are also referred to as selfish routing ones). Moreover, in this thesis a renowned game-theoretic traffic model due to Wardrop [35] is considered, introduced to represent road traffic with an infinite number of agents, each being responsible for an infinitesimal amount of traffic.

In this thesis, a distributed, non-cooperative and dynamic load balancing algorithm is consequently developed on the ground of mean-field game theory; specifically, the algorithm considers each request from a switch as an agent (whose decision is to determine the SDN Controller such a request has to be routed to), and is based on the measured response time of the SDN Controllers themselves: the algorithm is such that the agent decisions lead to an equilibrium, known in mean-field game theory as Wardrop equilibrium, where the values of the latency functions of the SDN Controllers are equalized.

Within this framework, a certain amount of traffic, or flow demand, has to be routed from a given source to a given destination via a collection of paths. Each agent has the possibility to distribute its own flow among a set of admissible paths. The network is characterized by non-decreasing latency functions depending on the flows on the edges. A combination of flows such that the latencies of all the employed paths are minimal is called a Wardrop equilibrium for the network. Indeed, a Nash equilibrium is said to become a Wardrop equilibrium whenever the number of decision makers is assumed to be infinite [36].

The main motivations behind this work are then (i) to prove, using Lyapunov arguments, how the difference equation governing the global state of the system (and macroscopically abstracting the microscopic evolution of the single agents involved) converges to an arbitrarily small neighbourhood of a Wardrop equilibrium, and (ii) to show the effectiveness of such an approach through its application to a real SDN scenario.

Distributed algorithms designed to make concurrent users converge to some game-theoretical equilibrium conditions often rely on re-allocating resources in a round-based fashion (see, for instance, [34] [37], and [38]). Some of these algorithms are based on the concept of sampling the different strategies at each round, in order to guarantee a certain degree of exploration of the surrounding environment and, at the same time, to favour the use of the best strategies (exploitation). This last approach is used in several learning techniques (see, for instance the reinforcement learning approach in), thus leading the algorithm to eventually converge to the optimal solution while providing acceptable solutions in the transitory phase.

However, in communication networks it is preferred to distribute the flows more regularly by splitting the transmission flows associated with each user into smaller flows, each one directed to one of the available providers, in a rate-based load balancing fashion. On the other hand, such approaches are usually presented as continuous-time algorithms which cannot be seamlessly implemented in a real communication network and whose advantages are highlighted only from a methodological point of view.

In the next sections, a discrete-time rate-based load balancing algorithm is designed, which retains the advantages of the rate-based approaches while being implementable. The algorithm is designed so as to dynamically learn a Wardrop equilibrium efficiently and in a distributed fashion; it can be regarded as a discrete-time version of the algorithm presented in [37] and is proven to converge to an equilibrium where all the latencies are equalized up to a given tolerance ε.

The performance of the SDN Controllers is defined by a latency function, which describes how the response time of the SDN Controller grows with its workload.

The objective of the load balancing algorithm is then to direct the requests of the switches to the SDN Controllers in such a way that the values of the latency functions of the SDN Controllers are equalized. The two main problems in the algorithm development are:

1. the fact that the latency functions are not known (e.g., the load/delay curve of an SDN Controller depends on its specific hardware and software implementation);
2. a distributed approach is needed since a centralized approach would require too much control traffic to exchange information among the SDN Controllers and potentially thousands of switches.

3.3.3. Preliminaries on Wardrop Equilibrium

This work further develops a well-known model for selfish routing [37] named Wardrop theory, where an infinite population of agents carries an infinitesimal amount of load each, following the previous works [39] concerning distributed load balancing algorithms.

In Wardrop theory, each agent is an infinitesimal portion of a specified commodity, whose objective is to minimize the cost sustained to reach its destination by a proper flow assignment. In the considered scenario, a single request of the request flow could be approximately considered as an agent. The population of each switch is then distributed among the available controllers according to the current flow vector (*strategy*); the *SDN-Proxy* then may decide to change the distribution of their own population among the controllers (*migration*). The aim of the switches is to balance the load of the controllers in a distributed fashion.

Let $C = \{1,2, \dots c\}$ denote a set of commodities with flow demands, or rates, $\lambda^i > 0, i \in C$, generally expressed in jobs per unit of time. For the sake of simplicity, each commodity $i \in C$ is associated with a (source, destination) couple of nodes, denoted with (s^i, d^i). The λ^i's are also such that $\sum_{i=1}^{c} \lambda^i = \lambda$. Let V denote a set of providers, which are used to transmit the flows for every commodity $i \in C$. All source nodes are connected by the network to the available providers, which, in turn, connect them to the destination nodes. As an example, we may think of the considered model as a description of a network consisting of a set of edges, over which the controllers arrange proper paths to connect source and destination nodes. In the considered scenario, the SDN Controllers are the providers, the SDN Proxies identify the commodities, and the λ^i's are their request loads, expressed in requests per unit of time.

The definition of *agent* is also required. As defined, for instance, in [38], each agent is an infinitesimal portion of a specified commodity, whose objective is to minimize the cost sustained to reach its destination by a proper flow assignment. In the considered scenario, a single request of the flow is approximately considered as an agent: in fact, even if the number of requests is finite, if the flow rates are sufficiently high, the population acceptably approximates the infinite population constraint required by Wardrop theory (see [37]).

Let x_p^i be the volume of the agents, or bandwidth, of commodity i relying on a provider $p \in V$. The vector $\boldsymbol{x} = \left(x_p^i\right)_{p \in V, i \in C}$ is the *flow vector* (in the literature also referred to as *population share* or *job vector)*, describing the overall amount of jobs per unit or time of commodity i.

Definition 1 (Feasible states).

The feasible state space, i.e., the closed set of feasible job vectors, is:

$$\mathcal{X} \coloneqq \left\{\boldsymbol{x} \in \mathbb{R}^{|V| \times |C|} \mid x_p^i \geq 0, \forall p \in V, \sum_{p \in V} x_p^i = \lambda^i, \forall i \in C\right\}. \quad (1)$$

where the λ^i's are the transmission rates required by each commodity i. ■

Let $x_p \coloneqq \sum_{i \in C} x_p^i$ denote the load of provider $p \in V$, and let each provider be characterized by a continuous cost function, referred to as *latency function* and denoted with $l_p(\cdot)\colon [0, \lambda] \to \mathbb{R}_+$. The latency of a provider p is a function of its load x_p, i.e., $l_p\left(x_p\right)$ is the latency of controller p with load x_p.

An instance of the load balancing game is then:

$$\Gamma = \left\{V, \left(l_p\right)_{p \in V}, \left(s^i, d^i, \lambda^i\right)_{i \in C}\right\}. \quad (2)$$

The load balancing problem is formulated below as the problem of determining the strategies which will lead the flow vector to reach an arbitrarily small neighbourhood of a Wardrop equilibrium. In Wardrop theory, *stable* flow assignments are the ones in which no agent (i.e., no "small" portion of a commodity directed from a source to a destination) can improve its situation by changing its strategy (i.e., the set of used providers) unilaterally. This objective is achieved if all agents reach a Wardrop equilibrium.

Definition 2 (*Wardrop equilibrium*, [37]).

A feasible flow vector $\boldsymbol{x}$ is at a Wardrop equilibrium for the instance Γ of the load balancing game if, for each provider $p \in V$ such that $l_p(x_p) > 0$, the following relation holds: $l_p(x_p) \leq l_q(x_q), \forall q \in V$. The set of all Wardrop equilibria is the following subset of $\mathcal{X}$:

$$\mathcal{X}_{\mathcal{W}} \coloneqq \{\boldsymbol{x} \in \mathcal{X} \mid l_p(x_p) - l_q(x_q) \leq 0,\ l_p(x_p) > 0, \forall p, q \in V\}. \tag{3}$$

■

In practice, at the Wardrop equilibrium, the latencies of all the loaded providers have the same value: therefore, provided that the latency functions properly represent the provider performances, a fair exploitation of the resources is achieved by driving the flows towards a Wardrop equilibrium.

In the framework of researches on Wardrop equilibria, a key role is played by the Beckmann, McGuire, and Winsten potential [40], given by:

$$\Phi(\boldsymbol{x}) \coloneqq \sum_{p \in V} \int_0^{x_p} l_p(s)ds, \tag{4}$$

whose properties are summarized in Property 1 below, under mild assumptions on the l_p's.

Assumption 1 (Latency functions).

The latency functions exhibit the following properties:

- $l_p(x)$ is positive and non-decreasing with $x \in [0, \lambda]$, $\forall p \in V$;
- $l_p(x)$ is Lipschitz continuous in $x \in [0, \lambda]$, with Lipschitz constant η_p, $\forall p \in V$. ■

Property 1 ([41], [42]).

Under Assumption 1, the potential (4) is continuous and has the following properties:

a) a flow minimizes Φ if and only if no agent can improve its own latency, implying that the set of Wardrop equilibria coincides with the set of flows minimizing Φ;

b) at least one positive minimum Φ_{min} of Φ over the set of feasible flows (and thus at least one Wardrop equilibrium) exists;

c) if the latency functions are strictly increasing, the minimizing flow is unique;

d) if the latency functions are strictly increasing and $l_p(0) = l_q(0), \forall p, q \in V$, the unique minimum of Φ is achieved when the latencies of all the providers are equalized. ■

The following definition and theorem on set stability (i.e., on the stability of a set of points in the state space) are also recalled with respect to the nonlinear discrete-time dynamics:

$$\boldsymbol{x}[k+1] = f(\boldsymbol{x}[k]), \quad \boldsymbol{x}(0) \in \mathcal{X}. \tag{5}$$

Definition 3 (*Positive definiteness*, [43]).

Let $\mathcal{X}$ be an invariant set for system (5), let $\mathcal{A}$ be a closed subset of $\mathcal{X}$ and let $d(\boldsymbol{x}, \mathcal{A}) \coloneqq \inf_{\boldsymbol{y} \in \mathcal{A}} |\boldsymbol{x} - \boldsymbol{y}|$ be the distance from a point $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{A}$ to $\mathcal{A}$. The function $\mathcal{L}(\boldsymbol{x}): \mathcal{X} \to \mathbb{R}_+$ is positive definite with respect to the set $\mathcal{A} \subset \mathcal{X}$ if there exists an increasing continuous function $\psi: \mathbb{R}_+ \to \mathbb{R}_+$ such that $\psi(0) = \psi_{min}$ and $\psi(d(\boldsymbol{x}, \mathcal{A})) \leq \mathcal{L}(\boldsymbol{x}), \forall \boldsymbol{x} \in \mathcal{X} \setminus \mathcal{A}$. ■

Let $\Delta\mathcal{L}(\boldsymbol{x}[k]) \coloneqq \mathcal{L}(\boldsymbol{x}[k+1]) - \mathcal{L}(\boldsymbol{x}[k])$ denote the difference of a Lyapunov function $\mathcal{L}(\boldsymbol{x})$ along the solutions of system (5). Lyapunov's second method can be applied to verify if a set is a Globally Asymptotically Stable Set (GASS) as follows [43].

Theorem 1 (Globally Asymptotically Stable Set).

Given a closed subset $\mathcal{A} \subset \mathcal{X}$ and a Lyapunov function $\mathcal{L}(\boldsymbol{x})$ in $\mathcal{X} \setminus \mathcal{A}$, if $\mathcal{L}(\boldsymbol{x})$ and $-\Delta\mathcal{L}(\boldsymbol{x}[k])$ are positive definite with respect to $\mathcal{A}$, then $\mathcal{A}$ is a GASS for system (5).

3.3.4. Proposed Load Balancing Control Rule and Convergence Proof

3.3.4.1. Load Balancing Algorithm

Let the system dynamics (5) be expressed component-wise by:

$$x_p^i[k+1] = x_p^i[k] + \delta \cdot \left(\sum_{q \in V} r_{qp}^i[k] - \sum_{q \in V} r_{pq}^i[k] \right), \quad \forall p \in V, \forall i \in C, k = 0,1, \ldots, \tag{6}$$

where δ is the sampling period and $r_{pq}^i[k]$ is the so-called *migration rate* from provider p to provider q. Inspired by the continuous-time algorithm in [37], the migration rate is defined as:

$$r_{pq}^i[k] = x_p^i[k] \cdot \sigma_{pq}^i[k] \cdot \mu_{pq}^i\left(l_p\left(x_p[k]\right), l_q(x_q([k])\right), \quad \forall p, q \in V, \forall i \in C, k = 0,1, \ldots, \tag{7}$$

where $\sigma_{pq}^i[k]$ is the control gain, which sets the rate with which the population share of provider p migrates to provider q, and $\mu_{pq}(l_p, l_q)$ is the so-called *migration policy*, representing the decision whether (and in which percentage) the population share assigned to provider p migrates to provider q.

The proposed migration policy has the following property:

$$\begin{cases} \mu_{pq}^i(l_p, l_q) = 0, & \text{if } l_p \leq l_q + \varepsilon, \\ \mu_{pq}^i(l_p, l_q) \in \left[\underline{\mu}, \bar{\mu}\right], 0 < \underline{\mu} < \bar{\mu} < +\infty, & \text{otherwise}, \\ \forall p, q \in V, \forall i \in C, \varepsilon > 0 \end{cases} \tag{8}$$

where ε is a tolerance on the maximum acceptable mismatch between the couples of latency values and $\underline{\mu}$ and $\bar{\mu}$ are positive lower- and upper-bounds, respectively. As shown in the following, the tolerance ε is introduced since the usual migration policies adopted in the continuous-time algorithms (obtained from (8) by setting $\varepsilon = 0$) cannot guarantee convergence in the discrete-time case, however small the sampling period (see, e.g., [44]).

Let the total migration rate from provider p to provider q be defined as $r_{pq}[k] \coloneqq \sum_{i \in C} r_{pq}^i[k]$. For notational simplicity, whenever unambiguous, $\mu_{pq}^i[k]$ will be used in place of $\mu_{pq}^i\left(l_p\left(x_p[k]\right), l_q(x_q([k])\right)$.

3.3.4.2. Convergence Proof

Before analysing the algorithm convergence, the following definition of ε-Wardrop equilibrium is introduced.

Definition 4 (ε-Wardrop equilibrium).

A feasible flow vector $\boldsymbol{x} = \left(x_p^i\right)_{p\in V, i\in C}$ is defined to be at an ε-Wardrop equilibrium for the instance Γ of the load balancing game if, for each provider $p \in V$ such that $l_p(x_p) > 0$, the following relation holds: $l_p\left(x_p\right) \leq l_u(x_u) + \varepsilon, \forall u \in V$, for $0 < \varepsilon < \bar{l} - \underline{l}$, where $\bar{l} \coloneqq \max_{p\in V} l_p(\lambda)$ and $\underline{l} \coloneqq \min_{p\in V} l_p(0)$ are the maximum and minimum latency values, respectively. The set of all ε-Wardrop equilibria is the following closed subset of $\mathcal{X}$:

$$\mathcal{X}_{\mathcal{W},\varepsilon} \coloneqq \left\{\boldsymbol{x} \in \mathcal{X}, \varepsilon > 0 \;\middle|\; l_p\left(x_p\right) \leq l_j\left(x_j\right) + \varepsilon, l_p\left(x_p\right) > 0, \forall j \in V, \forall p \in V,\ 0 < \varepsilon < \bar{l} - \underline{l}\right\}. \tag{9}$$

■

In practice, at an ε-Wardrop equilibrium, the latencies of all the loaded providers have the same value up to the tolerance ε.

Hereafter, the following assumptions on the latency functions and on the migration policy (8) will be considered.

Assumption 2.

The latency functions, the migration policy (8) and the control gain exhibit the following properties:

a) $l_p(x)$ is increasing with $x \in [0, \lambda]$, $\forall p \in V$;

b) $l_p(x)$ is Lipschitz continuous in $x \in [0, \lambda]$, with Lipschitz constant η_p, $\forall p \in V$;

c) $l_p(0) = \underline{l}$, $\forall p \in V$;

d) $\mu_{pq}^i\left(l_p, l_q\right)$ is Lipschitz continuous $\forall l_p, l_q \in \left[\underline{l}, \bar{l}\right], \forall i \in C$;

e) $\sigma_{pq}^i[k]$ is constant and equal to $\sigma = \frac{\varepsilon}{|V|\cdot\lambda\cdot\bar{\eta}\cdot\bar{\mu}\cdot\delta}$, where $\bar{\eta} \coloneqq \max_{p\in V} \eta_p$;

f) $\varepsilon < \bar{l} - \underline{l}$. ■

Assumptions 2.a) and 2.b) are slightly more restrictive than Assumption 1.

Assumption 2.a), introduced for the sake of simplicity in the system analysis, yields that, by Property 1, the Wardrop equilibrium and the corresponding flow vector, denoted with $l_{\mathcal{W}}$ and $\boldsymbol{x}_{\mathcal{W}}$, respectively, are unique.

Assumption 2.b) states that limited population differences lead to limited differences in the latency values.

Definition 5 (distance).

Let the norm of a state be defined as $\|\boldsymbol{x}\| \coloneqq \max_{p \in V}\left(l_p(x_p) - l_W\right)$, and let the distance between a state $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{W,\varepsilon}$ and the set $\mathcal{X}_{W,\varepsilon}$ be defined as:

$$d\left(\boldsymbol{x}, \mathcal{X}_{W,\varepsilon}\right) \coloneqq \|\boldsymbol{x}\| - \max_{\boldsymbol{y} \in \mathcal{X}_{W,\varepsilon}} \|\boldsymbol{y}\| > 0. \quad \blacksquare$$

Under Assumption 2, the set of Wardrop and ε-Wardrop equilibria can be written as:

$$\mathcal{X}_W \coloneqq \{\boldsymbol{x} \in \mathcal{X} \mid \|\boldsymbol{x}\| = 0\} = \{\boldsymbol{x}_W\}; \tag{10}$$

$$\mathcal{X}_{W,\varepsilon} \coloneqq \{\boldsymbol{x} \in \mathcal{X}, \varepsilon > 0 \mid \|\boldsymbol{x}\| \leq \varepsilon\}. \tag{11}$$

and the following properties hold.

Property 2.

The feasible set $\mathcal{X}$, and the set of the ε-Wardrop equilibria $\mathcal{X}_{W,\varepsilon}$ are such that:

(P1) $\mathcal{X}_{W,\varepsilon} = \mathcal{X}$ if $\varepsilon \geq \bar{l} - \underline{l}$;

(P2) $\mathcal{X} \supset \mathcal{X}_{W,\varepsilon_2} \supset \mathcal{X}_{W,\varepsilon_1} \supset \{\boldsymbol{x}_W\}, \forall \varepsilon_1, \varepsilon_2 \mid 0 < \varepsilon_1 < \varepsilon_2 < \bar{l} - \underline{l}$;

(P3) $\mathcal{X}_{W,\varepsilon} \to \{\boldsymbol{x}_W\}$ as $\varepsilon \to 0$. $\blacksquare$

The set convergence to an arbitrarily small neighborhood of the Wardrop equilibrium is proven by using the Beckmann, McGuire, and Winsten potential (4) to build a candidate Lyapunov function. The following lemma demonstrates some properties of the potential which will be used in the convergence proof of the subsequent Theorem 2.

Lemma 1 (Properties of the potential).

Under Assumption 2, the following properties hold for the nonlinear discrete-time system (3), (6), (7), (8), with total flow rate $\lambda > 0$:

(L1) $\Phi(\boldsymbol{x}) > \Phi_{min}, \forall \boldsymbol{x} \in \mathcal{X} \setminus \{\boldsymbol{x}_W\}, \Phi(\boldsymbol{x}_W) = \Phi_{min}$;

(L2) $\Phi(\boldsymbol{x}[k+1]) - \Phi(\boldsymbol{x}[k]) < 0, \forall \boldsymbol{x}[k] \in \mathcal{X} \setminus \mathcal{X}_{W,\varepsilon}$;

(L3) $\Phi(\boldsymbol{x}[k+1]) - \Phi(\boldsymbol{x}[k]) = 0, \forall \boldsymbol{x}[k] \in \mathcal{X}_{W,\varepsilon}$. $\blacksquare$

Proof. Conditions (L1) hold thanks to Property 1.

To verify condition (L2), the variation of the Lyapunov function along any trajectory of the considered system is written as follows:

$$\begin{aligned}
&\Delta\Phi(\boldsymbol{x}[k]) = \Phi(\boldsymbol{x}[k+1]) - \Phi(\boldsymbol{x}[k]) = \textstyle\sum_{p\in V}\int_{x_p[k]}^{x_p[k+1]} l_p(s)ds \\
&\leq \textstyle\sum_{p\in V}\big(x_p[k+1] - x_p[k]\big)\cdot l_p\big(x_p[k+1]\big) \\
&= \textstyle\sum_{p\in V}\big(\sum_q r_{qp}[k] - \sum_q r_{pq}[k]\big)\cdot \delta \cdot l_p\big(x_p[k+1]\big) \\
&= \textstyle\sum_{p\in V}\sum_{q\in V} r_{qp}[k]\cdot\delta\cdot l_p\big(x_p[k+1]\big) - \sum_{p\in V}\sum_{q\in V} r_{pq}[k]\cdot\delta\cdot l_p\big(x_p[k+1]\big) \\
&= \textstyle\sum_{p\in V}\sum_{q\in V} r_{pq}[k]\cdot\delta\cdot l_q\big(x_q[k+1]\big) - \sum_{p\in V}\sum_{q\in V} r_{pq}[k]\cdot\delta\cdot l_p\big(x_p[k+1]\big) \\
&= \textstyle\sum_{p\in V}\sum_{q\in V} r_{pq}[k]\cdot\delta\cdot\big[l_q\big(x_q[k+1]\big) - l_p\big(x_p[k+1]\big)\big], \qquad (12)
\end{aligned}$$

where the inequality holds from geometrical considerations (see Appendix A).

The following shows that (i), if $l_p\big(x_p[k]\big) - l_q\big(x_q[k]\big) > \varepsilon$, the corresponding term of the summation in the last row of (12) is negative, whereas (ii), if $l_p\big(x_p[k]\big) - l_q\big(x_q[k]\big) \leq \varepsilon$, then the term is null.

(i) If $l_p\big(x_p[k]\big) - l_q\big(x_q[k]\big) > \varepsilon$, from Assumptions 2.a)-2.c) it follows that $x_p[k] > 0$ and that $l_p\big(x_p[k]\big) > 0$. Moreover $\mu_{pq}^i[k] > 0$ from equation (8) and, thus, $r_{pq}^i[k] > 0, \forall i \in C$.

Now we need to show that $l_p\big(x_p[k+1]\big) - l_q\big(x_q[k+1]\big) > 0$. From equation (6) the following inequality holds:

$$l_p\big(x_p[k+1]\big) - l_q\big(x_q[k+1]\big) \geq l_p\big(x_p[k] - \textstyle\sum_{q\in V} r_{pq}[k]\cdot\delta\big) - l_q\big(x_q[k] + \sum_{p\in V} r_{pq}[k]\cdot\delta\big), \forall p,q \in V, k = 0,1,\ldots. \qquad (13)$$

In equation (13), the worst-case system dynamics over δ is considered, in which no commodities migrate part of their population to provider p and from provider q. From Assumption 2, since $\bar{\eta}$ is an upper-bound for the largest derivative of the l_p's, it holds that:

$$\begin{aligned}
&l_p\big(x_p[k]\big) - l_p\big(x_p[k] - \textstyle\sum_{q\in V} r_{pq}[k]\cdot\delta\big) \leq \bar{\eta}\cdot\sum_{q\in V} r_{pq}[k]\cdot\delta; \\
&l_q\big(x_q[k] + \textstyle\sum_{p\in V} r_{pq}[k]\cdot\delta\big) - l_q\big(x_q[k]\big) \leq \bar{\eta}\cdot\sum_{p\in V} r_{pq}[k]\cdot\delta.
\end{aligned}$$

Since we are analysing the $l_p\big(x_p[k]\big) - l_q\big(x_q[k]\big) > \varepsilon$ case, the following inequality holds:

$$l_p\big(x_p[k+1]\big) - l_q\big(x_q[k+1]\big) \geq \varepsilon - \bar{\eta}\cdot\delta\cdot\big(\textstyle\sum_{q\in V} r_{pq}[k] + \sum_{p\in V} r_{pq}[k]\big). \qquad (14)$$

From equation (7) and Assumption 2.a), and considering that $x_p^i[k] \leq \lambda^i, \forall i \in C$, the following upper-bound holds:

$$r_{pq}^i[k] = x_p^i[k]\cdot\sigma_{pq}^i[k]\cdot\mu_{pq}^i[k] \leq \lambda^i\cdot\sigma\cdot\bar{\mu}, \forall p,q \in V, \forall i \in C. \qquad (15)$$

Then, considering that there are at most $(|V|-1)$ terms in the first summation of the second term of (14), it is upper-bounded by:

$$\sum_{q\in V} r_{pq}[k] = \sum_{i\in C}\sum_{q\in V} r_{pq}^{i}[k] \leq \sigma\cdot\bar{\mu}\cdot(|V|-1)\cdot\lambda. \qquad (16)$$

Also, the second summation of the second term of (14) is upper-bounded by:

$$\sum_{p\in V} r_{pq}[k] = \sum_{i\in C}\sum_{p\in V} r_{pq}^{i}[k] \leq \sigma\cdot\bar{\mu}\cdot\sum_{i\in C}\sum_{p\in V} x_{p}^{i}[k] \leq \sigma\cdot\bar{\mu}\cdot\lambda. \qquad (17)$$

From equations (16) and (17), we obtain that a sufficient condition for the right-hand side of equation (14) to be non-negative is the following:

$$\varepsilon \geq |V|\cdot\sigma\cdot\lambda\cdot\bar{\mu}\cdot\bar{\eta}\cdot\delta. \qquad (18)$$

which holds by Assumption 2.e).

(ii) If $l_p\big(x_p[k]\big) - l_q\big(x_q[k]\big) \leq \varepsilon$, $r_{pq}[k] = 0$ by equations (7) and (8), and the corresponding term of the summation in the last row of (12) is null.

From (i) and (ii) it follows that property (L2) holds since, if $\boldsymbol{x}[k] \notin \mathcal{X}_{\mathcal{W},\varepsilon}$ (i.e., there exists at least one couple $(p,q)\in V^2$ such that $l_p\big(x_p[k]\big) - l_q\big(x_q[k]\big) > \varepsilon$, which, in turn, entails that $x_p[k] > 0$), at least one term of equation (12) is negative; property (L3) holds since, if $\boldsymbol{x}[k] \in \mathcal{X}_{\mathcal{W},\varepsilon}$ (i.e., for all couples $(p,q)\in V^2$ with $l_p\big(x_p[k]\big) > 0$ we have that $l_p\big(x_p[k]\big) - l_q\big(x_q[k]\big) \leq \varepsilon$), all the terms of equation (12) are null. ■

Theorem 2 (ε-Wardrop equilibrium set as a GASS).

Under Assumption 2, $\mathcal{X}_{\mathcal{W},\varepsilon}$ is a GASS for the nonlinear discrete-time system (3), (6), (7), (8), with total flow rate $\lambda > 0$. ■

Theorem 2Proof.

The proof is structured as follows: first, it is shown that the feasible state space is a positively invariant set (A); secondly, the asymptotic set stability is proven (B).

A. *Feasibility*.

It is shown in the following that, since the initial job vector is feasible (i.e., from Definition 1, $\sum_{p\in V} x_p^i[0] = \lambda^i$ and $x_p^i[0] \geq 0, \forall p \in V, \forall i \in C$), the job vector is feasible during the entire system dynamics. In fact, it follows from equation (6) that:

$$\sum_{p\in V}(x_p^i[k+1] - x_p^i[k]) = \sum_{p\in V}\sum_{q\in V}\left(r_{qp}^i[k] - r_{pq}^i[k]\right)\cdot\delta = \sum_{p\in V}\sum_{q\in V} r_{qp}^i[k]\cdot\delta - \sum_{q\in V}\sum_{p\in V} r_{qp}^i[k]\cdot\delta = 0, \quad (19)$$

and, therefore, that $\sum_{p\in V} x_p^i[k] = \sum_{p\in V} x_p^i[0] = \lambda^i, \forall k \geq 0$.

By induction, since $x_p^i[0] \geq 0, \forall p \in V$, and given equation (6), in order to prove that $x_p^i[k] \geq 0, \forall k \geq 0$, it is sufficient to assume that $x_p^i[k] \geq 0, \forall p \in V, \forall i \in C$, for a given k, and to prove that:

$$x_p^i[k+1] = x_p^i[k] + \sum_{q\in V}\left(r_{qp}^i[k] - r_{pq}^i[k]\right)\cdot\delta \geq 0, \forall p \in V, \forall i \in C. \quad (20)$$

In this respect, it can be observed that the following inequality holds (considering that, in the worst-case, no commodities migrate part of their population to provider p):

$$x_p^i[k+1] \geq x_p^i[k] - \sum_{q\in V} r_{pq}^i[k]\cdot\delta, \forall p \in V, \forall i \in C. \quad (21)$$

From definition (8) it follows that $r_{pp}^i[k] = 0$, so there are at most $(|V| - 1)$ terms in the summation of equation (21). Thus, considering that $r_{pq}^i[k] \leq x_p^i[k]\cdot\sigma\cdot\bar{\mu}$, the condition in (21) is met if the following inequality holds:

$$x_p^i[k] - x_p^i[k]\cdot(|V| - 1)\cdot\sigma\cdot\bar{\mu}\cdot\delta \geq 0, \forall p \in V, \forall i \in C. \quad (22)$$

If $x_p[k] = 0$, inequality (22) is verified. If $x_p[k] > 0$, inequality (22) is verified provided that:

$$\sigma \leq \frac{1}{(|V|-1)\cdot\delta\cdot\bar{\mu}}, \tag{23}$$

which holds by Assumption 2.e), considering that $\frac{\varepsilon}{\bar{\eta}\cdot\lambda} < 1$ (in fact, by the definitions of $\bar{\eta}$ and $\bar{l}$, it holds that $\bar{\eta}\cdot\lambda \geq \bar{l}$, and, by Assumption 2.f), it holds that $\bar{l} > \varepsilon$).

B. *Global asymptotic set stability*.

Let $\mathcal{L}(\boldsymbol{x}) \coloneqq \Phi(\boldsymbol{x}) - \Phi_{min}$ be the candidate Lyapunov function, where $\Phi(\boldsymbol{x})$ is the potential (3) and Φ_{min} is its minimum value, which is unique thanks to Assumption 2.

If $\boldsymbol{x} \in \mathcal{X}_{\mathcal{W},\varepsilon}$, from Lemma 1 it follows that $\mathcal{L}(\boldsymbol{x})$ is positive definite and that $\Delta\mathcal{L}(\boldsymbol{x}[k]) = 0$. If $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$, it is shown below that $\mathcal{L}(\boldsymbol{x})$ and $-\Delta\mathcal{L}(\boldsymbol{x}[k])$ are positive definite with respect to the closed set $\mathcal{X}_{\mathcal{W},\varepsilon}$.

B1. $\mathcal{L}(\boldsymbol{x})$ *is positive definite with respect to* $\mathcal{X}_{\mathcal{W},\varepsilon}$

Let $\psi\colon \mathbb{R}_+ \to \mathbb{R}_+$ be defined as follows: $\psi\big(d(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon})\big) \coloneqq \gamma_1 d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big)$, with $\gamma_1 > 0$. By definition, we have that $\psi(0) = 0$ and that $\psi\big(d(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon})\big)$ is increasing with $d(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon})$. We have to show that $\psi\big(d(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon})\big) = \gamma_1 d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big) \leq \mathcal{L}(\boldsymbol{x}) = \Phi(\boldsymbol{x}) - \Phi_{min}, \forall \boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$, i.e., that a value for γ_1 exists such that the following inequality holds:

$$\gamma_1 \leq \frac{\Phi(\boldsymbol{x}[k]) - \Phi_{min}}{d\big(\boldsymbol{x}[k], \mathcal{X}_{\mathcal{W},\varepsilon}\big)}, \forall k = 0,1,2,\ldots$$

Since $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$, we have that $d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big) > 0$; moreover, $d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big)$ is upper-bounded by $(\bar{l} - \underline{l})$ (by Assumption 2.d)). By geometrical considerations (see Appendix B), it turns out that $\Phi(\boldsymbol{x}) - \Phi_{min} > \frac{\varepsilon^2}{4\bar{\eta}} > 0$ for all $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$. Therefore, a suitable choice for γ_1 is $\gamma_1 = \frac{\varepsilon^2}{4\bar{\eta}(\bar{l}-\underline{l})}$.

B2. $-\Delta\mathcal{L}(\boldsymbol{x}[k])$ *is positive definite with respect to* $\mathcal{X}_{\mathcal{W},\varepsilon}$

Let $\psi\colon \mathbb{R}_+ \to \mathbb{R}_+$ be defined as $\psi\big(d(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon})\big) \coloneqq \gamma_2\, d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big)$, with $\gamma_2 > 0$. Also, consider that $\Delta\mathcal{L}(\boldsymbol{x}[k]) = \Delta\Phi(\boldsymbol{x}[k])$ and that, from Lemma 1, $\Delta\Phi(\boldsymbol{x}[k]) < 0, \forall \boldsymbol{x}[k] \in \mathcal{X} \setminus \boldsymbol{x}_{\mathcal{W}}$.

We have to show that $\psi\big(d(\boldsymbol{x}[k], \mathcal{X}_{\mathcal{W},\varepsilon})\big) = \gamma_2\, d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big) \leq -\Delta\mathcal{L}(\boldsymbol{x}[k]) = -\Delta\Phi(\boldsymbol{x}[k]), \forall \boldsymbol{x}[k] \in \mathcal{X} \setminus \mathcal{X}_{\mathcal{W},\varepsilon}$, i.e., that there exists a value for γ_2 such that the following inequality holds:

$$\gamma_2 \leq \frac{-\Delta\Phi(\boldsymbol{x}[k])}{d\big(\boldsymbol{x}[k], \mathcal{X}_{\mathcal{W},\varepsilon}\big)}, \forall k = 0,1,2,\ldots \tag{24}$$

For the numerator of equation (24), the following inequality holds (see Lemma 1, equation (12)):

$$-\Delta\Phi(\boldsymbol{x}[k]) = -\sum_{p\in V}\int_{x_p[k]}^{x_p[k+1]} l_p(s)ds \geq \sum_{p\in V}\sum_{q\in V}\delta\cdot r_{pq}[k]\left(l_p\big(x_p[k+1]\big) - l_q\big(x_q[k+1]\big)\right), \forall k = 1,2,\ldots \quad (25)$$

where the terms of the last summation are either null or positive.

From equations (14) and (18) it follows that $l_p\big(x_p[k+1]\big) - l_q\big(x_q[k+1]\big) \geq 0$. Let us consider the provider p^* which has the maximum latency value at time k, i.e., $p^* = \text{argmax}_{p\in V} l_p\big(x_p[k]\big)$. We thus write from equations (25), (7) and (8):

$$-\Delta\Phi(\boldsymbol{x}[k]) \geq \delta\cdot x_{p^*}[k]\cdot\sigma\cdot\underline{\mu}. \quad (26)$$

Since $\boldsymbol{x}[k] \in \mathcal{X}\setminus\mathcal{X}_{\mathcal{W},\varepsilon}$, we know that $l_{p^*}\big(x_{p^*}[k]\big) - l_q\big(x_q[k]\big) > \varepsilon, \forall q \in V$, yielding $l_{p^*}\big(x_{p^*}[k]\big) > \varepsilon$; since $\bar{\eta}$ is the upper-bound for the Lipschitz constants of the l_p's, we have that $\bar{\eta}\cdot x_{p^*}[k] \geq l_{p^*}\big(x_{p^*}[k]\big) > \varepsilon$. Thus, recalling that $d\big(\boldsymbol{x}, \mathcal{X}_{\mathcal{W},\varepsilon}\big) \leq (\bar{l} - \underline{l})$, the following choice for γ_2 lets inequality (24) hold for all $k = 0,1,\ldots$: $\gamma_2 = \frac{\delta\cdot\varepsilon\cdot\sigma\cdot\underline{\mu}\cdot}{\bar{\eta}\cdot(\bar{l}-\underline{l})}$. ■

Remark 1.

In the control law (7), $\sigma_{pq}^i[k]$ can be interpreted as the control gain, and the interpretation of Theorem 2 is that it sets an upper-bound σ on the control gain, with the twofold objective of keeping the dynamics feasible and of driving the system trajectories towards a neighbourhood of $\boldsymbol{x}_{\mathcal{W}}$.

3.3.4.3. Implementation Considerations

If the selected latency functions are increasing with limited slope (see Assumption 2), the implementation of the control law (7)-(8) requires the determination of the parameters appearing in Assumption 2.e): the number of providers $|V|$ is a scenario parameter; the parameter $\bar{\lambda}$ can be set by practical knowledge either of the maximum offered load in the considered use case or of the maximum load which can be managed by the set of available providers;

The control time δ and the maximum tolerated latency mismatch ε are set according to a practical trade-off between tight control (small values of δ and ε) and traffic overhead/convergence time (small values of δ imply more latency measures, whereas the maximum control gain σ increases with ε);

The determination of the maximum latency value $\bar{l}$ and of the maximum Lipschitz constant $\bar{\eta}$ is less straightforward and is explained in the following.

In practice, it is usually simple to find a suitable latency function representing the provider performances, since they typically degrade with the load (in fact, the typical load-response time curve $f(x)$ is often modelled in the theory of M/M/1 queues by monotonically increasing functions such as $f(x) = \frac{1}{\lambda - x}$ or $f(x) = \frac{x}{\lambda - x}$).

In most cases, the latency values represent an actual measure of the performance of the providers and can be upper-bounded by a value $\bar{l}$, set according to realistic considerations.

The latency value is then set equal to $\bar{l}$ whenever the value of the latency computed from the provider performance measures is larger than $\bar{l}$. For instance, in the scenario described in the following Section, the latency represents the controller response time; in practice, there are Quality-of-Service constraints which should be met by the provider, in terms of maximum response time, defining the upper-bound $\bar{l}$.

The maximum Lipschitz constant $\bar{\eta}$ of the latency functions, instead, must be inferred from actual provider performance measures by estimating the maximum slope of the measured latency curves. We note that, by limiting the maximum value $\bar{l}$ of the latency functions according to practical consideration (as described above), we also limit the Lipschitz constant $\bar{\eta}$ since, usually, the slope of the latency functions increases with the load, with beneficial effects on the control effectiveness since the gain σ is increased (see Assumption 2.e)).

At the start of the control round k:

- Each SDN Proxy $i = 1, \dots, |C|$:
  - For each SDN Controller $p = 1, \dots, |V|$:
    - It computes the measured latency $l_p^{meas}[k]$ by averaging, over the last period δ, the measures of the delays between the transmission of a request to the SDN Controller p and its response. If no request was sent to the SDN Controller p in the last round, it sends a fake request to obtain the response time measure.
    - It updates the value of the latency function by following a simple exponential averaging approach:
      $l_p[k] \leftarrow \alpha l_p[k-1] + (1-\alpha) l_p^{meas}[k]$, with $\alpha \in (0,1)$.
  - It computes the migration rates $r_{pq}^i[k], \forall p, q, \in V$ according to equation (7), with $\sigma_{pq}^i[k] = \sigma$ as defined in Assumption 2.e).
  - It computes the flow rates $x_p^i[k], \forall p \in V$, according to equation (6).

During the control round k of duration δ:

- Each SDN Proxy $i = 1, \dots, |C|$:
  - It sends the requests received from its associated SDN Switches during round k to the SDN Controllers according to a weighted round-robin scheduling, with weights proportional to the flow rates $x_p^i[k], p = 1, \dots, |V|$.

Table 2 Algorithm Implementation

3.4. Evaluation and Results

In order to evaluate the correctness and the performance of the proposed balancing algorithm a preliminary phase of problem modelling and evaluation through numerical simulation was performed; then the presented balancing algorithm was designed and its convergence proved; a proof of concept implementation of the proposed load balancing control algorithm was developed using the target enabling technologies.

Results of both phases are reported and discussed in the following sections.

3.4.1. Simulation Modelling and Results

In [17], an algorithm to dynamically learn a Wardrop equilibrium efficiently and in a distributed fashion is defined. The algorithm is mainly based on the concept of weighing between *exploitation* and *exploration* to guarantee the convergence of the algorithm to a Wardrop equilibrium. The *exploitation* policy, for a learning algorithm, simply consists in using the best strategy computed by the algorithm so far. The *exploration* policy, by contrast, is aimed at trying new unexplored strategies in order to estimate their effectiveness.

The *round-based* approach described in the following is an application of the approach proposed in [45] to the scenario considered in this evaluation phase.

In particular, the proposed algorithm is aimed at dynamically learning the most efficient combination of flow rates from each network switch to any of the available controllers, thus (i) ensuring the convergence to stable policies and consequently balancing the OpenFlow control traffic efficiently, (ii) increasing the overall throughput, and (iii) minimizing the control connection latencies.

Let the relative slope, or elasticity, of a function be defined as follows:

Definition 3 (*Elasticity*, [45]):

The elasticity e_l of a differentiable function $l(x)$ with $x \in [0,1]$ is defined as:

$$e_l(x) := \frac{dl(x)/dx}{l(x)} \cdot x. \quad (1)$$

For instance, the exponential function $l(x) = a \cdot e^{kx}$, with $x \in [0,1]$, has elasticity at most equal to k; the polynomial function $l(x) = a \cdot x^k$ has elasticity k.

In the model considered, each switch transmits a flow of requests towards the controllers. The switch has to decide, for each request of the flow, which controller to use among the available ones.

In detail, periodically, with period T_{ctrl}, the switches decide the percentages of the request flow to be transmitted to each available controller. The time-scale is then discrete with decision instants $\tau_k = k \cdot T_{ctrl}, k = 0,1, \dots$.

Let us consider that a switch $s_i, i = 1, \dots, C$, is currently sending a rate $f_{i,j}$ of its requests to the controller $c_j \in \Xi$. In every round of T_{ctrl} seconds, i.e., at each time instant τ_k, a request (that is, an agent) is "activated" with constant probability γ, i.e., a request generated by the switch can decide to change the controller with probability γ.

More precisely, every T_{ctrl} seconds, the decision algorithm named (α, β)-exploration-replication policy (see [45]) is performed in two steps, as described in Table 3.

Step	Action	Description
1.	*Sampling*	With probability $(1-\beta)$ perform step 1.a) and with probability β perform step 1.b)
1.a)	*Proportional sampling*	Sample controller $c_{j'}$ with probability $f_{i,j'}/\nu_i, j' = 1, \dots, C$
1.b)	*Uniform sampling*	Sample controller $c_{j'}$ with probability $1/\lvert\Xi_i\rvert, j' = 1, \dots, C$
2.	*Migration*	If $l_{j'} < l_j$, migrate to controller j' with probability $\frac{1}{e_{ub}}\frac{l_j - l_{j'}}{l_j + \alpha}$

Table 3 (α,β)-exploration-replication policy of switch S_i

In practice, at regular intervals, with probability γ, a switch $s_i \in \Sigma$ decides that a request which was planned to be sent to a controller $c_j \in \Xi$ has to be migrated towards another controller $c_{j'} \in \Xi$ such that $j' \neq j$.

The algorithm in Table 3 is therefore executed and, in Step 1, the new controller $c_{j'} \in \Xi$ is selected (sampled) by using one out of two sampling techniques:

- under proportional sampling, used with probability $(1-\beta)$, the probability of sampling a controller is proportional to the flow rate of the requests from switch s_i to controller $c_{j'}$ at the current round, i.e., the "reputation" of the controllers is used as an indicator of their performance (*exploitation* of successful strategies);
- under uniform sampling, used with probability β, every controller is sampled with uniform probability, in such a way that every controller has a positive probability of being sampled (*exploration* of the strategy space).

It turns out that the parameter β determines the balance between exploitation and exploration.

In Step 2, the algorithm decides if the request will be sent to the old controller c_j, or to the new one $c_{j'}$ selected in Step 1. The probability of choosing the new controller is proportional to the latency gain between the old and the new controller.

The parameter $\alpha > 0$ (see Step 2 in Table 3) is introduced to prevent small latency values from causing too large migrations. The parameter e_{ub} is an upper bound for the elasticity of the latency functions, namely, it is such that $e_l(x) \leq e_{ub}$ for every $x \in [0,1]$.

The migration policy of Table 3 calculates the amount of requests that are shifted between any pair of controllers within one round. Given a flow vector f, for all switches $s_i, i = 1, \dots, S$, and for all couples of controllers $c_j, c_j' \in \Xi$ such that $l_{j'} < l_j$, the (α, β)-exploration-replication policy migrates a fraction of agents $\rho_{i,j,j'}(f)$ from controller j to controller j'. The expected migration rates are equal to:

$$\mathcal{E}\{\rho_{i,j,j'}(f)\} = \gamma \cdot f_{i,j} \cdot \frac{1}{e_{ub}}\frac{l_j - l_{j'}}{l_j + \alpha} \cdot \left[(1-\beta)\frac{f_{i,j'}}{\nu_i} + \beta\frac{1}{|\Xi_i|}\right], \quad (2)$$

where $\mathcal{E}\{\cdot\}$ denotes the expected value operator.

Note that the migration rules are such that:

$$\rho_{i,j,j'}(f) = -\rho_{i,j',j}(f), \forall i = 1, \dots, S, \forall j, j' = 1, \dots, C. \quad (3)$$

Let $f_{i,j}(\tau_k)$ be the request rate of switch $s_i \in \Sigma$ transmitted to controller $c_j \in \Xi$ at the k-th round.

The dynamics of the request flow $f_{i,j}$ is then:

$$f_{i,j}(\tau_{k+1}) = f_{i,j}(\tau_k) - \sum_{j'=1,\dots,C} \rho_{i,j,j'}, k = 0,1, \dots . \quad (4)$$

Note that, thanks to equation (5), the transmitted request rate keeps constant over time:

$$\sum_{j=1,\dots,C} f_{i,j}(\tau_k) = v_i, \forall i = 1, \dots, S, \forall k = 0,1, \dots . \quad (5)$$

The following Theorem ensures that the described algorithm converges.

Theorem 1 (*Convergence to a Wardrop equilibrium*, [45]):

By activating an agent with constant probability $\gamma = 1/32$, the (α, β)-exploration-replication policy leads system (6) to a Wardrop equilibrium if the elasticity of the latencies is upper-bounded by $e_{ub} \geq 1$, if the parameter $\alpha > 0$ and if the parameter β satisfies the following inequality:

$$\beta \leq \frac{\min_{c_j \in \Xi} l_j(0) + \alpha}{\max_{c_j \in \Xi} \max_{x' \in [0,\beta]} \left(\frac{dl_j(x)}{dx}\right)_{x=x'}}. \quad (6)$$

Equation (6) computes the ratio between the minimum controller latency and the maximum derivative of the controller latencies for "small" flow rates, i.e., with rates smaller than or equal to β.

Since β is the probability that the exploration policy is used, equation (6) can be regarded as the maximum exploration of the strategy space that the system can stand without affecting the algorithm convergence. The convergence time of the algorithm depends on the elasticity of the latency functions, and is approximated as $\mathcal{O}\left(\frac{d}{\varepsilon^2\delta^2} \log \frac{d \max_x \frac{dl(x)}{dx}}{\min_x l(x)}\right)$, where α and ε define an approximation of the Wardrop equilibrium (see [17] for a detailed discussion).

One of the key factors in the success of the load balancing problem is the choice of the latency functions, which have to intrinsically contain significant information about the behavior of each controller when a particular joint strategy (for all switches) is chosen.

This paper considers the normalized response time of the controller, expressed in ms, as its latency function. This choice is motivated by the fact that the response time grows with the controller load and thus (i) it is a reliable indicator of the controller congestion status, and (ii) it is a non-decreasing function of the request rate and therefore a suitable latency function.

Heterogeneous controllers are also implicitly considered: controllers with different processing capabilities have the same congestion level when their response time is the same.

Another advantage of this choice is that the response time can be easily computed by each switch, since it must only measure the delay between a request to a controller and the controller response minus the round-trip transport delay from the switch to the controller (easily measured, e.g., by a ping message).

3.4.2. Numerical Simulation Results

The proposed approach has been implemented in MATLAB®. The algorithm is executed by each switch every T_{ctrl} seconds and determines the switch strategy to be pursued during the next control period, i.e., the percentage of requests, for each switch, to be sent to each controller. As mentioned in the previous Section, the strategies are computed based on the latencies of the controllers, which directly depend on their response times.

In the simulations, the response time, i.e., the latency function, is modelled as a quadratic function:

$$l_j\left(f_j(t)\right) = a_j \cdot f_j^2(t), j = 1, \dots, C, \tag{9}$$

where a_j is a constant characterizing the processing capabilities of controller j.

The scenario consists of a 100×100 square Euclidean plane, in which the controllers are regularly spatially distributed, and the switches are distributed according to a Gaussian distribution with mean $\mu = (33.3, 33.3)$ and standard deviation $\sigma = (6.25, 6.25)$. Figure 18 shows an example of displacement of controllers and switches. The transport delay between a switch and a controller is simply assumed to be equal to their Euclidean distance.

For each request, the proposed switch decision process is described by the following two simple steps:

1. the Load Balancer sends a request to one of the available SDN controllers with a given probability, which depends on the estimated probability density over the SDN controllers;
2. the Load Balancer measures then the time needed by the selected controller to answer; the request-answer delay is regarded as a measure of the controller congestion status and used to update the probability density over the SDN controllers exploited at step 1.

The proposed approach is therefore scalable, since no communications among the switches is needed and no centralized load balancing algorithm must be executed by the SDN controllers.

The objective of the simulation is to show that the algorithms converges to an equilibrium where the SDN controllers are equally loaded by the requests workload.

The proposed approach has been implemented in MATLAB®. The algorithm is executed by each switch every T_{ctrl} seconds and determines the switch strategy to be pursued during the next control period, i.e., the percentage of requests to be sent to each controller. As mentioned, the strategies are computed based on the latencies measures of the controllers, which directly depend on their response times.

In the simulations, the response time, i.e., the latency function, is modelled as a quadratic function:

$$l_j\left(f_j(t)\right) = a_j \cdot f_j^2(t), j = 1, \dots, C, \tag{9}$$

where a_j is a constant characterizing the processing capabilities of controller j.

The scenario considered consists of a 100×100 square Euclidean plane, in which the controllers are regularly spatially distributed, and the switches are distributed according to a Gaussian distribution with mean $\mu = (33.3, 33.3)$ and standard deviation $\sigma = (6.25, 6.25)$. Fig. 2 shows an example of displacement of controllers and switches. The transport delay between a switch and a controller is simply assumed to be equal to their Euclidean distance.

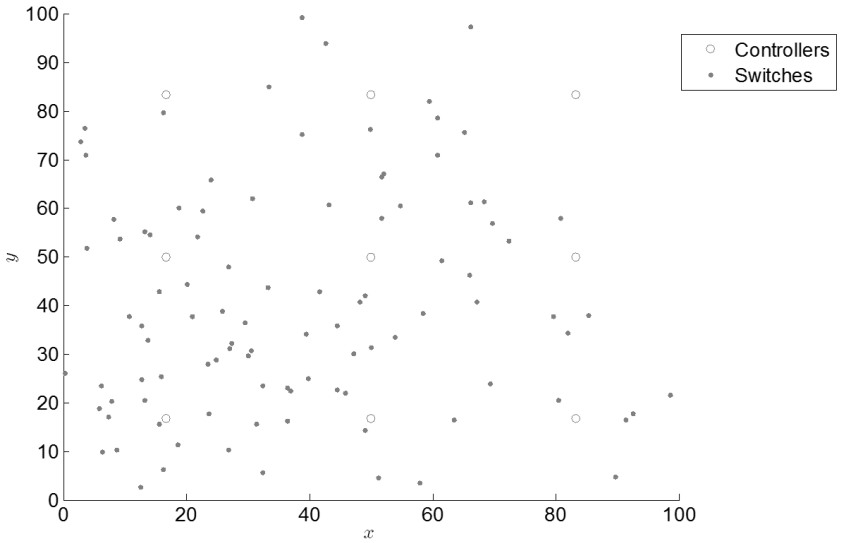

Figure 18 Load Balancing Splitting

Ten simulations have been carried out, each one repeated ten times. In each simulation $q = 1, \dots, 10$ the positions of the switches are randomly assigned at the beginning of each run. The average request rate of the switches is computed as $\bar{v} = q \cdot 10^6$ $[\frac{req}{s}]$, and the request rate of each switch is computed as $v_j = \bar{v} \cdot unif\{0.5, 1.5\},\ j = 1, \dots, C$, where $unif\{a, b\}$ denotes a random number between a and b extracted from a uniform distribution. Similarly, the latency constants are computed as $a_j = 10^{-12} \cdot unif\{0.5, 1.5\},\ j = 1, \dots, C$. Table 3 shows the chosen simulation parameters.

The proposed algorithm has been compared to a static switch-controller association strategy, in which each switch sends its requests to the nearest controller. Figure 19 and Figure 20 show the simulation results, averaged over the ten runs executed for each scenario, in terms of average delay and of the standard deviations of the delays, respectively. Such a delay is computed as the time needed by the controller to execute a request plus the transport delay from the switch to the controller.

Figure 19 shows that Wardrop load balancing outperforms the "nearest controller" strategy, halving the average response time as the request load increases. Even more interesting are the results shown in Figure 20: by equalizing the latencies, the Wardrop load balancing algorithm equalizes the response times of the controllers, therefore the standard deviations of the delays only depend on the fact that the transport delays are different for each (switch, controller) couple. On the contrary, the nearest controller strategy determines different loads for the controllers and therefore different response times.

To show how the improvement was achieved, Figure 21-Figure 24 provide some details on one simulation run with average request rate $\bar{v}_q = 1.3 \cdot 10^6$ $[\frac{req}{s}]$.

Figure 21 shows how the algorithm distributes the request load among the nine available controllers. Since the controllers are heterogeneous (i.e., their response time or latency curves are different), the loads converge to different values for each controller. Figure 22 shows that the obtained load distribution is such that the latencies, and therefore the response times, of the controllers are indeed equalized.

Figure 23 shows the total delays, whose variations depend on the fact that the transport delays from the switches to the controllers are different. Finally,

Figure 24 shows the eventual strategy determined by the Wardrop algorithm for a subset of switches $s_i, i = 1, \dots, 10$. Considering a switch s_i, the corresponding bar in Figure 24 is divided into segments: each segment is proportional to the percentage of requests that switch s_i sends to a given controller, identified by the figure legend (for example, switch s_6 sends 3.70% of requests to controller c_7, 0.53% of requests to controller c_8, 95.77% of requests to controller c_9 and no requests to the other controllers).

The Wardrop strategy is then capable of counteracting the non-homogeneous spatial distribution of switches and controllers, with no explicit communications either among controllers or among switches.

The numerical simulations show improved performances in comparison with a static switch-controller association strategy – defined so that each switch sends its requests to the nearest controller –, therefore suggesting the applicability of the presented approach to real scenarios.

Parameter	Value
S	100
C	9
T_{ctrl}	$1s$
γ	$1/32$
β	0.1
α	1
e_{ub}	2
T_{CTRL}	$1s$
λ	0.5

Table 4 Simulation Parameters

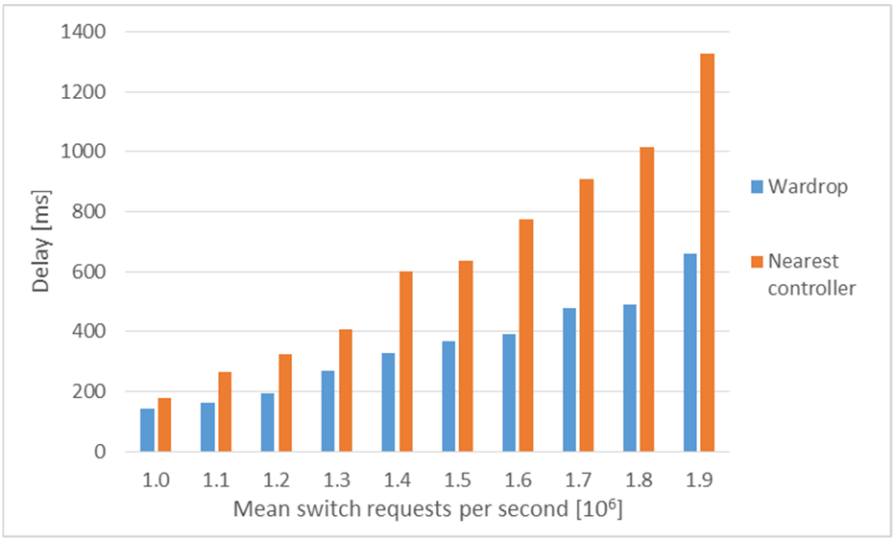

Figure 19 Delays with Wardrop strategy and nearest controller strategy

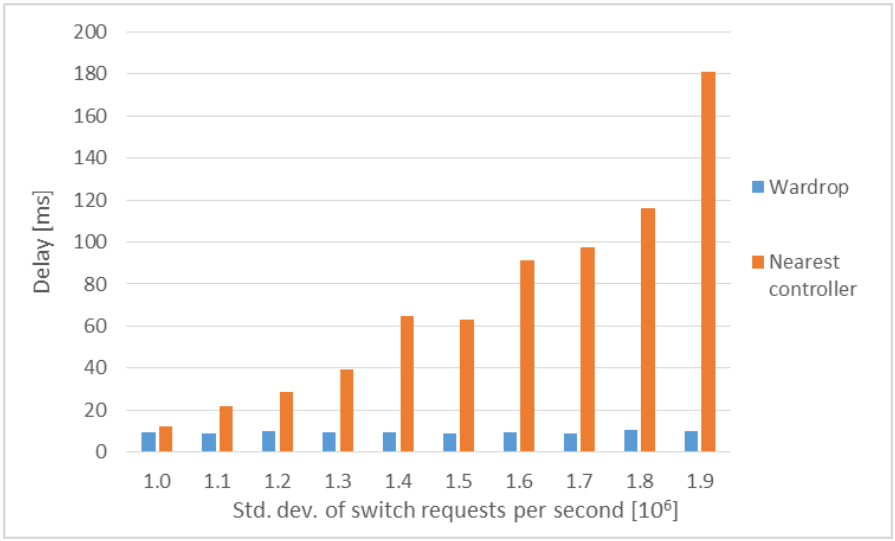

Figure 20 Standard deviations with Wardrop strategy and nearest controller strategy

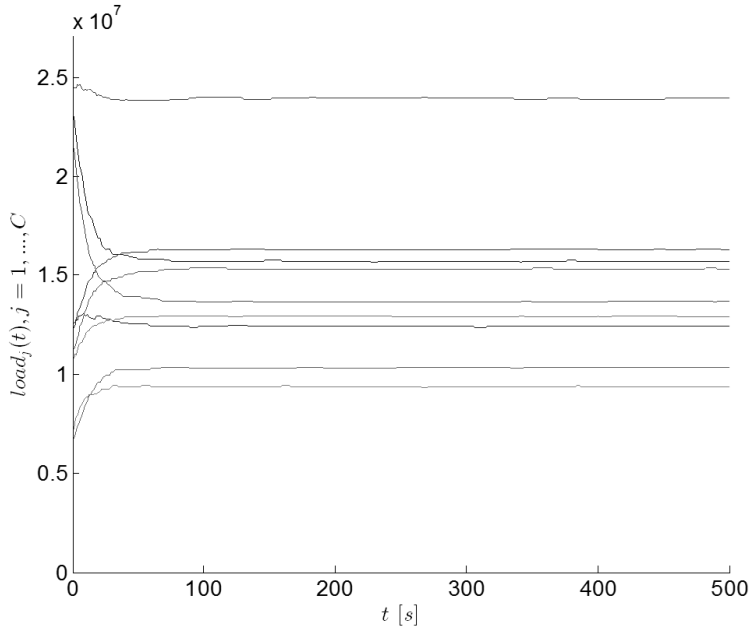

Figure 21 Controller loads

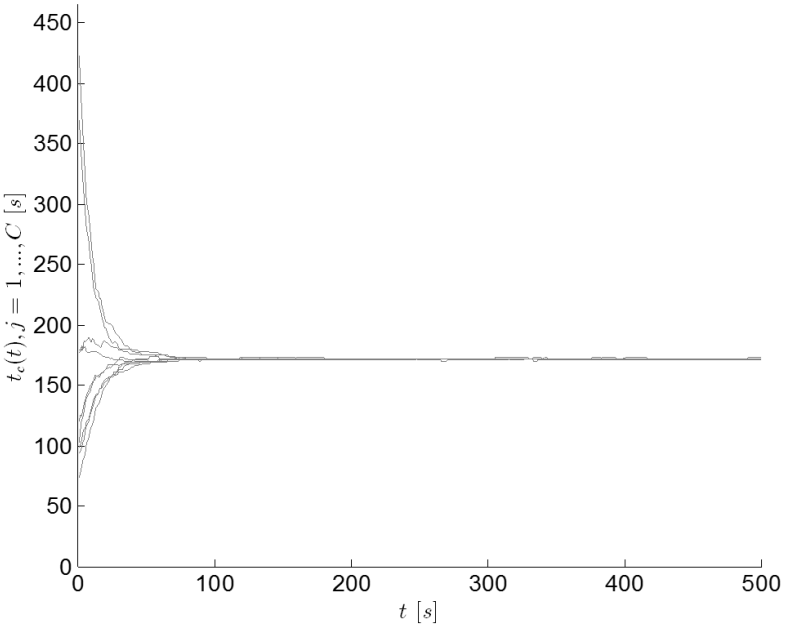

Figure 22 Latency values, i.e., controller response times

.

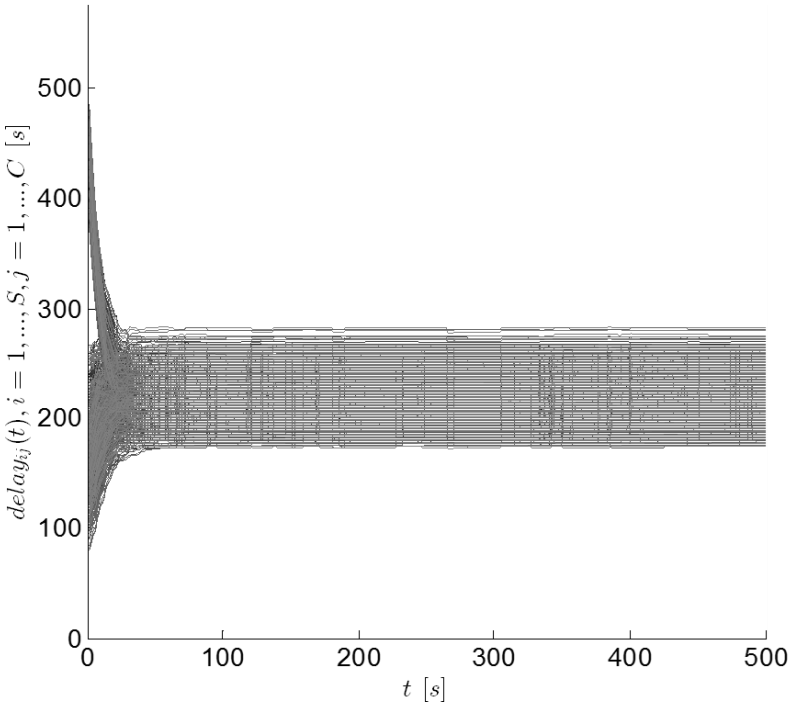

Figure 23 Switch-controller total delays

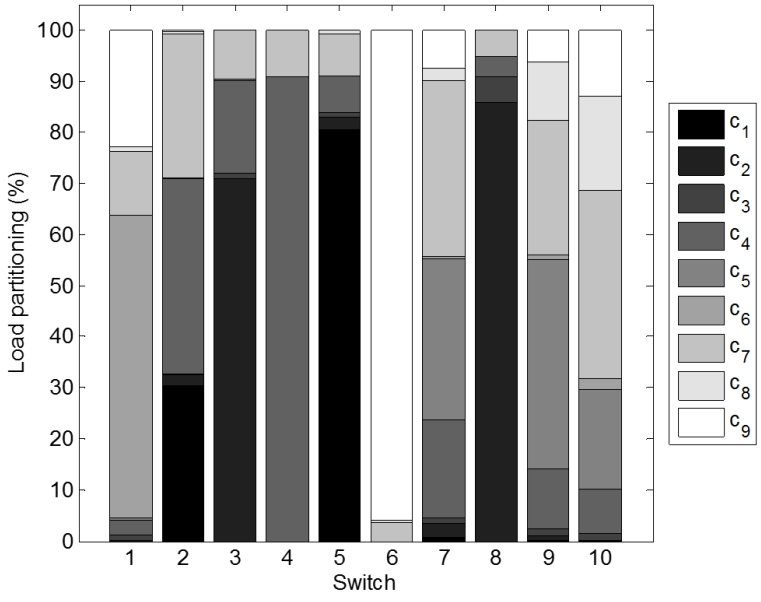

Figure 24 Wardrop Strategy for Switches S_i

3.4.3. Load Balancer Proof of Concept Implementation

The presented algorithm has been applied to enforce the discussed load balancing mechanisms onto the distributed SDN Control Plane developed within the T-NOVA project. The currently adopted solutions rely on a Control Plane adopting a logically centralized but physically distributed architectural model. This model considers the Control Plane as distributed across a cluster of multiple SDN Controllers.

Focusing on the emulation of the OpenFlow control traffic rather than on the emulation of the data plane traffic, we present the results obtained from a proof-of-concept experimental setup where the considered SDN works in Equal Interaction configuration across a cluster of three OpenDaylight SDN Controllers (Lithium release).

To validate the effectiveness of the load balancing algorithm, different tests with the generated OpenFlow control traffic have been performed relying on the WCBench (Wrapped Collective Benchmark) tool, which works as a generator of OpenFlow control traffic by emulating OpenFlow switches.

The requests were generated on the data plane and then sent to the SDN Proxies. Each implemented SDN Proxy, developed in Python, is a network proxy in charge of catching the generated OpenFlow traffic and embeds the proposed load balancing algorithm as well as two other algorithms for comparison purposes. Each SDN Proxy is connected to all the available SDN Controller instances and performs a per-request balancing policy.

The introduction of the SDN Proxies is transparent to the OpenFlow standard. Each SDN Proxy receives the requests of its switches and has the task of forwarding them to one of the available SDN Controllers, based on a load balancing algorithm as the one proposed. The algorithm considers each request from a switch as an agent (whose decision is to determine the SDN Controller such a request must be routed to) and is based on the measured response time of the SDN Controllers themselves: the algorithm is such that the agent decisions lead to an equilibrium where the values of the latency functions of the SDN Controllers are equalized within a tolerant parameter, as described in the previous sections.

In this case study, the latency associated with an SDN Controller is its average response time. The response time grows with the controller load and thus:

(i) It is a reliable indicator of the controller congestion status;

(ii) It is a nondecreasing function of the request rate and therefore a suitable latency function, and

(iii) It can be easily computed by the SDN Proxies, as explained below.

The time-scale is divided into rounds of duration δ. At every round, the latency is evaluated, and the control actions are implemented as described in Table 2 Algorithm Implementation.

To simulate the SDN Controller load, OpenFlow control traffic was generated on the data plane side at different request rates.

The three deployed SDN Controllers were assumed to have different processing capabilities. Two configurations are used for the SDN Controller resources, namely, the Small one with 2GB of RAM and 2 vCPUs, and the Big one with 4GB of RAM and 4 vCPUs.

Consequently, as shown in the following figure, the load-response time curves are different.

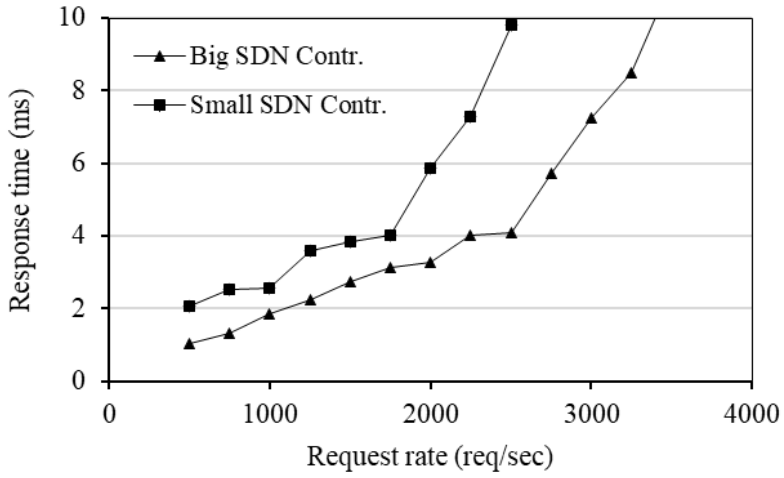

Figure 25 Load vs. response time curves of the two considered configurations for the SDN Controller resources

The curves were obtained by measuring the average response time of the providers, denoted with $\tau_p(x_p)$, for different load values $x_p[k] = x_p$.

In detail, a constant rate of x_p requests per seconds was sent for 180s to each controller $p \in V$. The values of the x_p's ranged from $\underline{x} = 500$ req/s to the maximum load value such that the response time exceeds $\bar{l}$, denoted with $\bar{x}_p$, with granularity $\Delta x = 250$ req/s.

The measured latency variations were then used to estimate the maximum slope as:

$$\bar{\eta} \approx \max_{p \in V, x_p \in \{\underline{x}, \underline{x}+\Delta x, \ldots, \bar{x}-\Delta x\}} \frac{l(x_p+\Delta x)-l(x_p)}{\Delta x}.$$

From Figure 25, we can empirically assess that the latency of the adopted SDN Controllers grows with the request rate, thus exhibiting a positive and non-decreasing behaviour that satisfies Assumption 2.

In the implemented Wardrop load balancing algorithm, the migration policy (8) is defined as:

$$\mu_{pq}^{i}(l_p, l_q) = \begin{cases} 0, & \text{if } l_p \leq l_q + \varepsilon, \\ \min\left(\frac{l_p - l_q}{\bar{l} - \underline{l}}, \bar{\mu}\right), & \text{otherwise,} \end{cases} \forall p, q \in V, \forall i \in C.$$

The scenario and algorithm parameters are reported in the below table.

Parameter	**Value**
Number of SDN Controllers	$\|V\| = 3$
Number of SDN Proxies	$\|C\| = 2$
Maximum load	$\lambda = 10\text{^}4 \left[\frac{req}{s}\right]$
Maximum latency value	$\bar{l} = 10\ [ms]$
Minimum latency value	$\underline{l} = 0\ [ms]$
Maximum Lipschitz constant of the latency functions	$\bar{\eta} = 4 * 10\text{^} - 5\ \left[\frac{ms^2}{req}\right]$
Latency tolerance	$\varepsilon = 0.3\ [ms]$
Maximum value of the migration policy	$\bar{\mu} = 1$
Sampling time	$\delta = 1\ [s]$
Averaging constant	$\alpha = 0.95$

Table 5 Scenario and Algorithm Parameters

3.4.3.1. Evaluation Results

Figure 26 - Figure 29 show the test results, in terms of throughput, defined as the rate at which the incoming requests are processed by the SDN Control Plane, and latency (response time).

We have compared the performance of the proposed per-request Wardrop load balancing algorithm with two approaches:

- a simple Least Latency load balancing algorithm, devised in a closed-loop fashion;
- an open-loop Round Robin load balancing algorithm, as adopted by HAProxy 1 (High Availability Proxy).

The Least Latency load balancing algorithm at time k distributes traffic to the SDN Controller currently exhibiting the lowest measured average latency. The Round Robin load balancing algorithm, given the list of available SDN Controllers, forwards the control traffic to each SDN Controller in turn (i.e., it migrates a predefined amount of traffic to the first listed SDN Controller at time k, then to the second listed SDN Controller at time $k + 1$, and so on).

The test results show that the Wardrop load balancer outperforms the other two approaches, by yielding a performance improvement evaluated in terms of total throughput and average latency of 13% and 18%, respectively, if compared with the Least Latency algorithm, and of 17% and 31%, if compared with the HAProxy Round Robin one.

The figures show how the proposed algorithm sends a larger request rate to the third SDN Controller which has more resources (Big configuration) with respect to the first and second ones (Small configurations), in such a way that the latencies are equalized.

The Least Latency algorithm operates in a similar way but is less effective, since it does not dynamically adjust the migration rates to obtain convergence; the Round Robin algorithm simply balances the request rate among the three SDN Controllers, which causes the latencies of the first and second SDN Controllers to grow above the latency of the third one. Figure 29 also shows that the ε-Wardrop equilibrium is practically reached, since the maximum difference between latencies is equal to 0.26ms; the difference grows with the Least Latency and Round Robin algorithms to 2.48ms and 0.96ms, respectively.

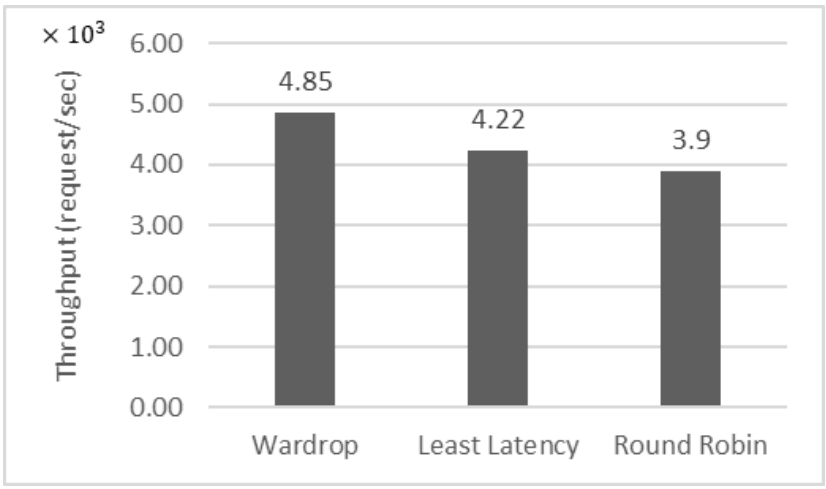

Figure 26 Average throughput comparison among Wardrop, Least Latency and HAProxy Round Robin load balancing algorithms.

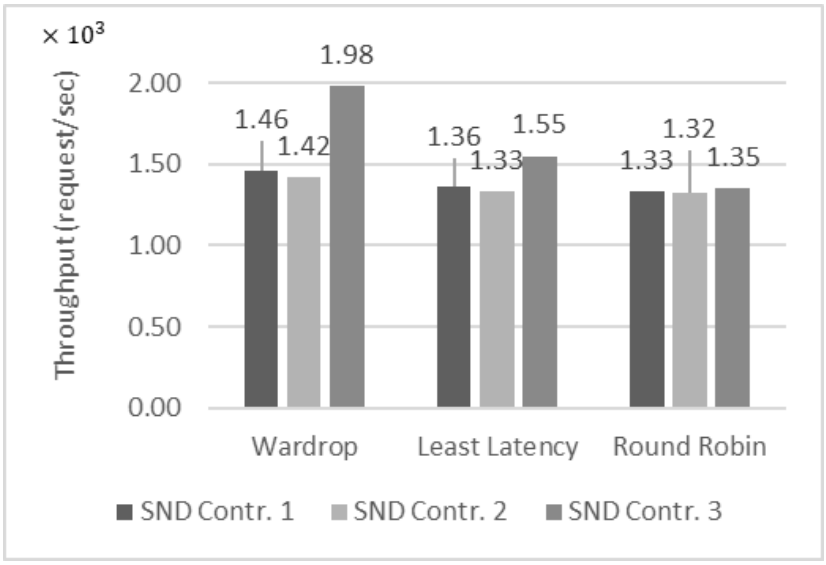

Figure 27 Per SDN Controller throughput comparison among Wardrop, Least Latency and HAProxy Round Robin load balancing algorithms.

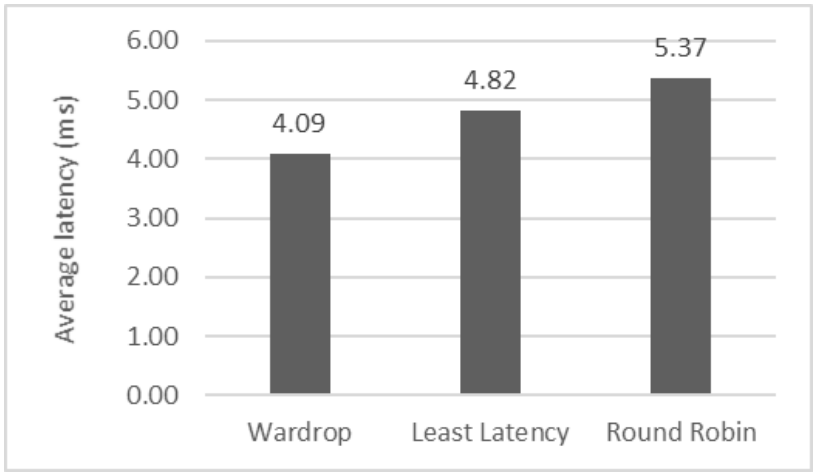

Figure 28 Response time (latency) comparison among Wardrop, Least Latency and HAProxy Round Robin load balancing algorithms.

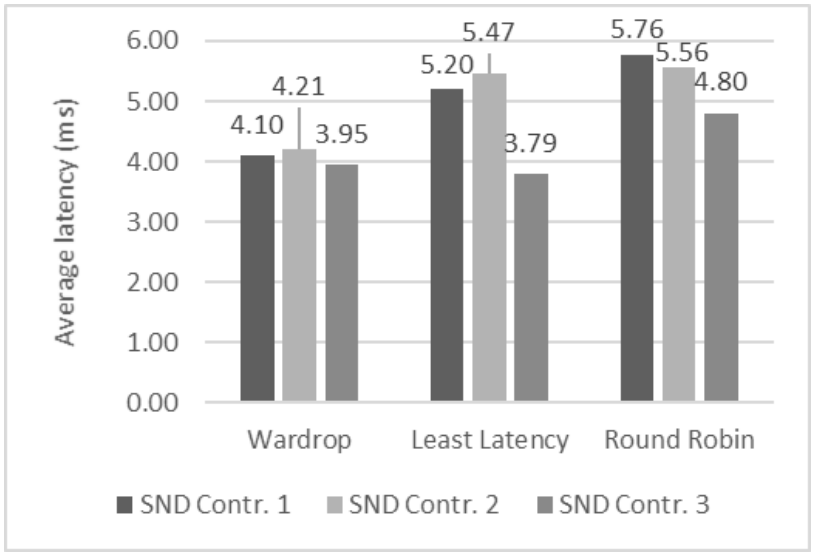

Figure 29 Per SDN Controller response time (latency) comparison among Wardrop, Least Latency and HAProxy Round Robin load balancing algorithms.

3.5. Conclusion

In order to address the performance issues related to the adoption of a distributed SDN Control Plane, a distributed load balancing algorithm with the aim of dynamically balancing the control traffic across a cluster of SDN Controllers is presented. In particular, the proposed algorithm is aimed at dynamically learning the most efficient combination of flow rates from each controller, thus (i) ensuring the convergence to stable policies and consequently balancing the OpenFlow control traffic efficiently, (ii) increasing the overall throughput, and (iii) minimizing the control connection latencies.

The proposed discrete-time, distributed, non-cooperative load balancing algorithm suitable for the and based on game theory and converged to a specific equilibrium known as Wardrop equilibrium. The Wardrop load balancing algorithm for SDN networks is proved to converge to an arbitrarily small neighbourhood of a Wardrop equilibrium. From an architectural point of view, SDN Proxies for the OpenFlow traffic are introduced to improve the scalability of SDN networks by dynamically dispatching the control workload across the available SDN Controllers. A proof-of-concept implementation on a real SDN network has been carried out and the related performance test results are reported. The proposed approach is scalable, since no communications among the switches is needed and no centralized load balancing algorithm must be executed by the SDN Controllers.

It represents a concrete result of the technology and knowledge transfer applied by the author in the context of EU funded H2020 FIWARE and T-NOVA research projects.

Appendix A

Geometrical Considerations Proving Inequality (12)

The quantity $(x_2 - x_1) \cdot l(x_2)$, represented by the area of the rectangle with bold lines in Figure 30, is larger than the integral $\int_{x_1}^{x_2} l(s)ds$, represented by the grey area.

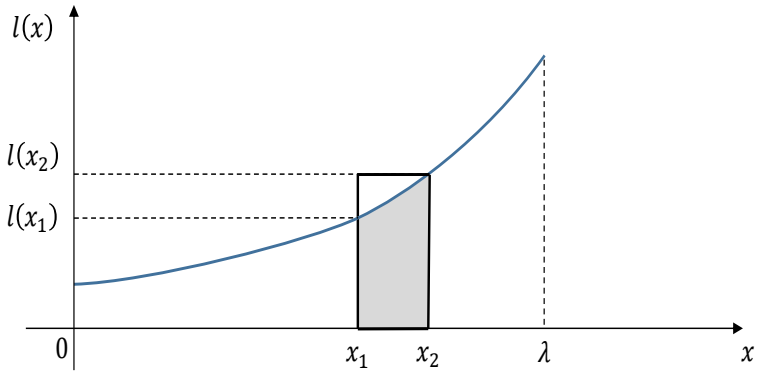

Figure 30 Geometrical considerations proving inequality

Appendix B

Geometrical Considerations Proving Argument B1 in Section 3.3.4.B

The Let $\boldsymbol{x}_W$ be the flow vector at the Wardrop equilibrium, i.e., $l_p(x_{p,W}) = l_W$, $\forall p \in V$, and let $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{W,\varepsilon}$ be defined by the vector $(x_p)_{p\in V} = (x_{p,W} + \Delta x_p)_{p\in V}$, where Δx_p is the difference (positive, negative or null) between x_p and $x_{p,W}$. The difference between the potential in $\boldsymbol{x}$ and the potential (4) at the equilibrium $\Phi_{min} = \Phi(\boldsymbol{x}_W)$ is then:

$$\Phi(\boldsymbol{x}) - \Phi(\boldsymbol{x}_W) = \left(\sum_{\substack{p\in V \\ \Delta x_p>0}} \int_0^{x_{p,W}+\Delta x_p} l_p(s)ds + \sum_{\substack{q\in V \\ \Delta x_q<0}} \int_0^{x_{q,W}+\Delta x_q} l_q(s)ds + \sum_{\substack{m\in V \\ \Delta x_m=0}} \int_0^{x_{m,W}} l_m(s)ds\right) - \sum_{n\in V} \int_0^{x_{n,W}} l_n(s)ds$$

$$= \sum_{\substack{p\in V \\ \Delta x_p>0}} \int_{x_{p,W}}^{x_{p,W}+\Delta x_p} l_p(s)ds - \sum_{\substack{q\in V \\ \Delta x_q<0}} \int_{x_{q,W}+\Delta x_q}^{x_{q,W}} l_q(s)ds.$$

Considering the areas $\mathcal{B}_q$ and $\mathcal{A}_q$ depicted in Figure 31, we can write:

$$\Phi(\boldsymbol{x}) - \Phi_{min} = \sum_{\substack{p\in V \\ \Delta x_p>0}} (\Delta x_p l_W + \mathcal{A}_p) - \sum_{\substack{q\in V \\ \Delta x_q<0}} (\Delta x_q l_W - \mathcal{B}_q) =$$

$$\sum_{\substack{p\in V \\ \Delta x_p>0}} \mathcal{A}_p + \sum_{\substack{q\in V \\ \Delta x_q<0}} \mathcal{B}_q \geq \sum_{\substack{p\in V \\ \Delta x_p>0}} \frac{\varepsilon_p^2}{2\bar{\eta}} + \sum_{\substack{q\in V \\ \Delta x_q<0}} \frac{\varepsilon_q^2}{2\bar{\eta}},$$

where the last equality holds since $\mathcal{A}_p \geq \frac{\varepsilon_p^2}{2\bar{\eta}}$ and $\mathcal{B}_q \geq \frac{\varepsilon_q^2}{2\bar{\eta}}$ and where $\varepsilon_p \coloneqq l_p(x_p) - l_W$ and $\varepsilon_q \coloneqq l_W - l_q(x_q)$.

Since $\boldsymbol{x} \in \mathcal{X} \setminus \mathcal{X}_{W,\varepsilon}$, there exists at least a couple $p', q' \in V$ such that $\Delta x_{p'} > 0$ $\Delta x_{q'} < 0$ and $\varepsilon_{p'} + \varepsilon_{q'} > \varepsilon$. It follows that:

$$\Phi(\boldsymbol{x}) - \Phi_{min} \geq \frac{\varepsilon_{p'}^2}{2\bar{\eta}} + \frac{\varepsilon_{q'}^2}{2\bar{\eta}} \geq \frac{\left(\varepsilon_{p'}+\varepsilon_{q'}\right)^2}{4\bar{\eta}} > \frac{\varepsilon^2}{4\bar{\eta}}.$$

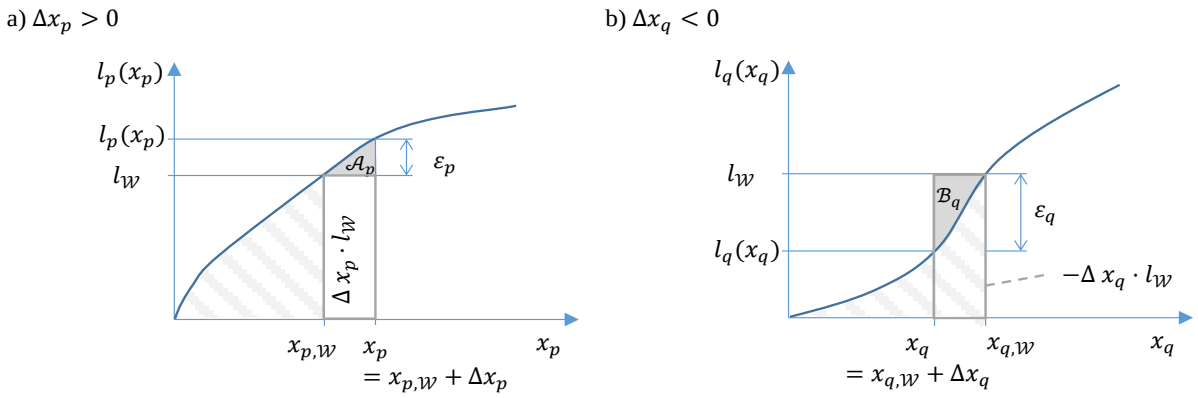

Figure 31 Geometrical considerations providing argume B1 in section 3.3.4.B

Chapter 4

User Quality of Experience Evaluation and Control

A key Future Internet initiatives objective is to design transparent, efficient, fair and flexible mechanisms for controlling all available resources so that each user perceives a Quality of Experience (QoE) as close as possible to his desired QoE. In this chapter, a cognitive architecture supporting QoE management is presented, within orchestration algorithms which take evaluation of the QoE value and control decisions and mechanisms to transparently enforce such control decisions into the underlying heterogeneous telecommunication networks and cloud infrastructures

4.1. Introduction

Network service providers are becoming increasingly aware of the importance of customer experience in a more and more competitive market, especially since service quality has started replacing tariffing as the key selling point. Hence, solutions that help network service providers gain a comprehensive view of the end-user experience, together with means and methods for improving it, are key for their business.

At this scope, the current research is focused on solutions where applications transparently, efficiently and flexibly exploit the available resources while satisfying the expectations of the involved actors.

Most of the actual limitations are related to the fact that most of the algorithms and procedures embedded in the telecommunication networks are open-loop. They are setup and configured on the base of off-line, reasonable estimations of network variables, rather than on real-time measurements of such variables.

This static approach does not fit well with the dynamicity of the Future Internet application and services and claims for an evolution towards closed-loop algorithms and procedures able to properly react on the base of appropriate real-time feedback and monitoring information. Therefore, a new kind of user-centric metric is needed, based on the QoE concept.

The ITU-T [6] organization defines the term Quality of Experience (QoE) as: “the overall acceptability of an application or service, as perceived subjectively by the end-user”.

Different categories of factors influencing the QoE:

- technological factors such as video resolution, framerate or decoding algorithm, bandwidth, delay, jitter etc;
- environmental factors such as user location and background, space, time of day, frequency of use etc;
- human/psychological factors such as visual and auditory acuity, gender, age, mood; but also, more high-level factors such as cognitive processes, social-cultural and economic background, expectations, needs and goals.

In this respect, some initial works mainly focused on the definition of QoE and on its relationship with Quality of Service (QoS), defined by the ITU-T as “the totality of characteristics of a telecommunications service that bear on its ability to satisfy stated and implied needs of the user of the service”, are [46] and [47]. In particular these studies focus on the mapping functions between QoE and QoS, for different services. These mapping functions [48] can be linear, cubic, exponential, logarithmic or logistic functions. Most of them are specific to a service type or to a protocol, but the logarithmic Weber-Fechner Law, the Stevens’ Power Law and the exponential IQX Hypothesis can be generalized to all the services.

Among them, the literature suggests that the IQX Hypothesis seems to be the most effective and reliable choice for most of the service scenarios. At the basis of this mapping, there is the fact that, given the same amount of change of the QoS value, the change of QoE depends on the current level of QoE [49].

Anyway, QoS metrics are traditionally used to assess the performances of on-line services and networked elements, but they are not sufficient to monitor and control the satisfaction of the users: user satisfaction is affected by end-to-end factors that include technological, environmental and human/psychological factors.

A large amount of research is on-going in the field of the identification of the personalized user expected QoE level in a given context for a given application (e.g. see [50] for voice and [51] for video applications, respectively), as well as of the functions for QoE computation, including passive and active monitorable feedback parameters which serve as independent variables for these functions. By passive and active parameters, we mean the ones which are independent of and dependent on the active involvement of the users in their computation, respectively [52].

The approach assumed in this work for the QoE control is based on the analysis not only of the static parameters characterizing the users and the applications, but also of properly selected dynamic, real-time, multi-layer, multi-network and monitorable parameters, hereafter referred to as Feedback Parameters.

In particular, we assume that the perceived QoE of a user is computed by means of properly selected functions – the structure of these functions depends on the user and application profiles – whose independent variables are properly selected Feedback Parameters which can span from "traditional" QoS parameters characterizing the application like bandwidth delay, throughput, packet loss, etc.), to security and mobility parameters (e.g., related to privacy, dependability, roaming, and handover performance), as well as explicit feedbacks directly provided by the users (e.g., through proper clicking mechanisms).

In the next section, the proposed framework for the QoE management is introduced and discussed. First the QoE Management Framework architecture is introduced and each of submodule of interest presented and discussed from a functional point of view. Then the QoE Evaluator within its algorithm is introduced, the same for the QoE Controller. Evaluation results both for the QoE Evaluator both for the QoE Controller are presented and discussed. During a service fruition such framework in able to asses a personalized level of QoE of a user, by evaluating his level of perceived QoE exploiting network information and implicit/explicit feedback and produce driving parameters aiming at reducing the difference between the target and perceived QoE, namely, the QoE Error. Such driving parameters are abstract parameters that should be translated in per-domain network configuration and enforced into the underlying infrastructure, in order to optimize the allocation of network resources while improving the user satisfaction.

From a technological point of view, the proposed solution is in line and compliant to the Software Defined Network (SDN) and Network Function Virtualization (NFV) trend. The exploitation of the SDN-enabled NFV infrastructure can provide monitoring and actuation mechanism for the network, compute and storage resources.

4.2. QoE Management Framework Architecture

The Orchestration Layer of the Future Internet platform (see section 2.2) is the most interesting one from a control perspective since it includes the cognitive features of the overall architecture.

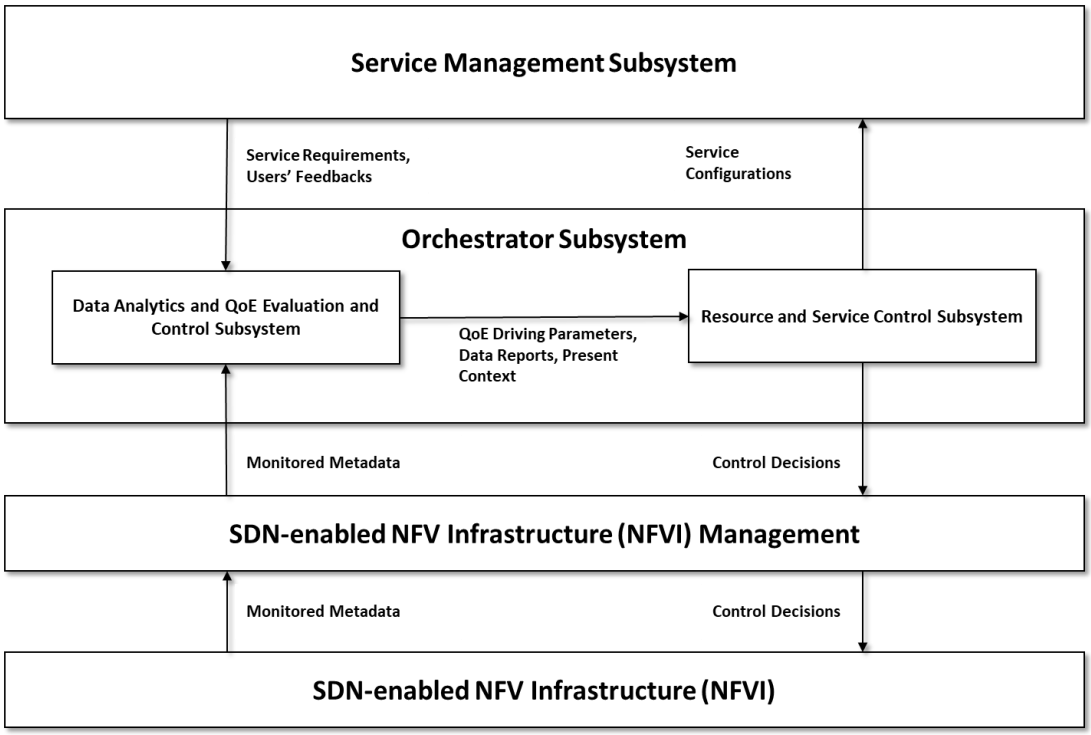

Figure 32 Proposed Future Internet Architecture

Here the *Data Analytics and QoE Evaluation/Control Subsystem* (see Figure 33), implementing the QoE Management framework, is presented and discussed.

The proposed approach for coping with QoE Management in this context is to implement a modular and cognitive architecture where the QoE Management functionalities consist of a QoE Evaluator and a QoE Controller, which can be designed independently from one another.

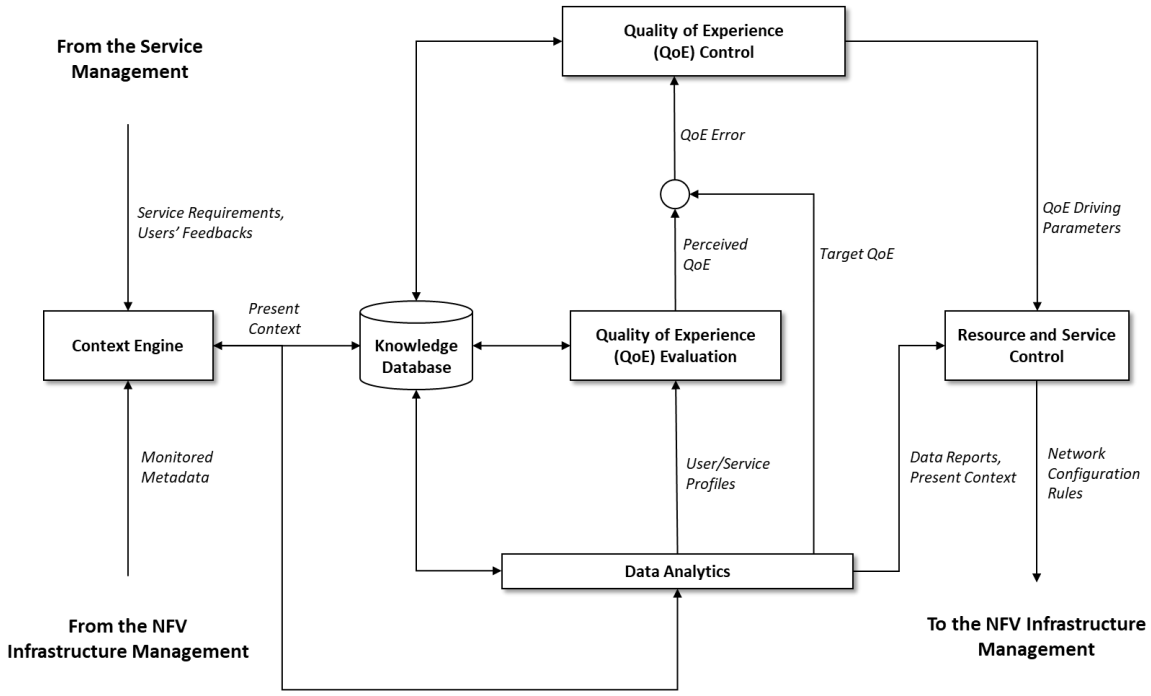

Figure 33 Architecture of the Data Analytics and QoE Evaluation/Control Subsystem

Such block is composed of the following subsystems:

- The *Context Engine*, which receives in real-time the Monitored Metadata from the NFV Infrastructure as well as the metadata relevant to the Service Parameters and the Users' Feedbacks (e.g., via social networks) from the Service Management Layer. The Context Engine is, therefore, in charge of the formal description, the appropriate aggregation and the semantic enrichment of all the received metadata, eventually producing the so-called Present Context, i.e., a real-time multi-layer, multi-network and technology-independent structured record of the present state, characterizing each service session enjoyed by a user over a certain network. The Context Engine is also in charge of continuously feeding a Knowledge Database which stores all the updates of the Present Context.
- The *Data Analytics Subsystem*, which is in charge of performing the analysis of the data stored in the Knowledge Database (which can be considered as "Big Data"). This subsystem includes properly designed pattern recognition techniques (e.g., Support Vector Machine algorithms), aimed at inferring the so-called Ad-Hoc Profiles (each profile corresponding to a suitable cluster of users presenting similar session records), as well as the personalized QoE desired by a single user while enjoying a given service (Target QoE).
- The *QoE Evaluator*, which is in charge of assessing, in real-time, for each user enjoying a given service, the so-called Perceived QoE, i.e., the QoE that is currently being perceived by the user. Such computations are to be performed based on a suitable set of personalized QoE Utility Functions depending on the Present Context and on the Ad-Hoc Profiles, as explained in the next section.

- The *QoE Controller* is aimed at the satisfaction of the personalized QoE requirements, namely at the minimization, for each running service and for each user, of the personalized QoE Error defined as the difference between the Perceived QoE (input from the QoE Evaluator) and the Target QoE (input from Data Analytics) of the considered service. The QoE Controller is in charge of the real-time computation of proper QoE Driving Parameters, namely personalized performance target values, which will then be exploited in order to ensure the desired minimization of the QoE Error associated with each service. As a result, the overall architecture will employ the dynamically deduced QoE Driving Parameters not only to drive the QoS performance of the underlying telecommunication networks, but also to ensure real-time control of the security and mobility performance of the network and/or cloud infrastructures, as well as the real-time control of the QoE Management and service delivery procedures.
- The *Resource and Service Control Subsystem* operates based on the Present Context (input from the Context Engine) and of the QoE Driving Parameters (input from the QoE Controller). It consists of a set of cooperative and technology-independent control functionalities in charge of making appropriate coordinated and technology-neutral Control Decisions which will then affect the underlying network infrastructures, as well as of producing automatic Service Notifications associated with the detection of network/service/computing anomalies due to security problems or faults. The Service Notifications will eventually trigger suitable adaptation or reconfiguration which will be properly enforced either by the SDN-enabled NFV Layer (if the detected anomalies are related to the underlying network and/or cloud infrastructures), or by the Service Management Layer (otherwise). The Control Decisions are responsible for driving the network resources of the underlying telecommunication networks and the computing/storage resources of the underlying cloud infrastructures to reach the performance target values provided by the QoE Driving Parameters, while simultaneously ensuring efficient resource exploitation. In this respect, such Control Decisions will specify what actions will have to be enforced by the Ad-Hoc Actuation functionalities in terms of scheduling, admission control, selection of the telecommunication domains which will have to support the admission of the requested service, traffic load balancing intra-domain routing, inter-domain handovers, etc.

From the above discussions, it should be clear that the Orchestrator plays the "key role" of control: The Data Analytics and QoE Evaluation/Control Subsystems "rearrange" the feedback variables (to provide the Feedback Parameters represented by the Present Context) and generate the target reference values (namely, the Target QoE). Furthermore, the Orchestrator, based on the Feedback Parameters, produces the control variables, namely the Control Decisions impacting on the SDN-enabled NFV Layer and the Service Notifications impacting on the Service Management Layer.

4.3. QoE Evaluation

The problem of definition of reliable measures for QoE has been faced in several practical applications due to the massive development of resource demanding multimedia services over Internet.

As in [53], suitable optimization strategies are made possible by an accurate evaluation of the users' QoE so to minimize the network resources usage and, at the same time, to satisfy the actual users expectations on the Quality of Service (QoS).

Several approaches show the dependency of QoE from QoS parameters according to different services [54]. The mapping functions between QoE and QoS functions [48] can be linear, cubic, exponential, logarithmic depending on the service type or protocol. Anyway, the logarithmic Weber-Fechner Law, the Stevens' Power Law and the exponential IQX Hypothesis can be generalized to all the services. Among them, the literature suggests that the IQX Hypothesis seems to be the most effective and reliable choice for most of the service scenarios. At the basis of this mapping, there is the fact that, given the same amount of change of the QoS value, the change of QoE depends on the current level of QoE [49].

In these approaches the role of explicit feedback from the user plays an important role, the fact that similar users can provide similar feedbacks about the perceived quality has not been explicitly considered so far. In this thesis, we consider a QoE measure system for a telecommunication service, where the QoE measure is usually strongly related to the QoS parameters [55] [56].

Though there is no doubt on the fact that QoS parameters (like bandwidth, delay, jitter, and packet loss) directly affect the user QoE, a reliable measurement system to provide a completely QoS based function representing the user QoE is still missing. Both subjective (e.g. the Mean Opinion Score) and objective measurement systems have been proposed in valuable and pioneering researches (as in [57] [58] [59]).

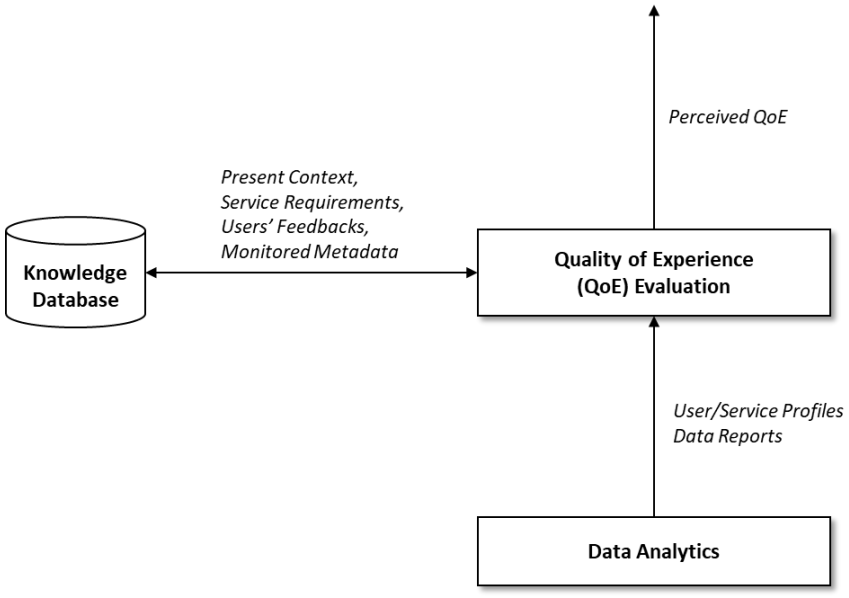

Figure 34 QoE Evaluator Interaction

The approach assumed in this work cope with the problem of automatically identifying distinct user profiles characterized by similar behaviour under the same context scenario. The proposed mechanism for the QoE evaluation operates by exploiting a QoE Utility Function able to asses in real-time the perceived QoE level by a user based on static parameters characterizing the users and the applications, but also on properly selected dynamic, real-time, multi-layer, multi-network and monitorable parameters, hereafter referred to as Feedback Parameters.

A data model that automatically extracts significant features from raw data time series is proposed; a clustering algorithm for the identification of the most significant user profiles is introduced. The results of a validation phase on a proof of concept implementation using data gathered from preliminary field trials are presented and extensively discussed.

Since the approach is based on personalized and being dynamic QoE Utility Functions, which are personalized on the specific service and user profiles, the proposed mechanism results high flexible by providing a personalized QoE metrics computation tool.

4.3.1. Procedure

The QoE Evaluator is in charge of evaluating, for each Application, the personalized QoE expected by the user (hereinafter referred to as Target QoE) and the actual, present QoE experienced by the user (hereinafter referred to as Perceived QoE). The QoE Evaluator should operate both on a real-time basis, using the User Feedbacks sent by the Applications, or in batch, using the Historical Data, stored in a database.

We assume that the quality experienced by the user is closely linked to the following two types of parameters:

1) the static parameters that remain unchanged during the session (e.g. service type, content, user ID, etc);
2) the dynamic parameters that, generally, change during the session; we will distinguish two sub-types of dynamic parameters, that is to say:
    a) the Quality of Service (QoS) parameters such as jitter, packet loss, delay, throughput, but also parameters related to safety and user's mobility;
    b) the feedback provided by the user to give his/her subjective measure of satisfaction with respect to the specific service. Even in this case, we refer to feedback to indicate both the explicit value returned by the user (for example using a MOS scale) and the implicit one (e.g. the frequency of the clicks, stop/pause events, etc).

These parameters are collected in a proper data structure hereafter referred to as Session Report, for the subsequent recognition of the behavioural profiles. At the end of the service fruition, the Session Report is saved into the database.

In order to specifying information from data to define their characteristics based on the identified different behavioural profiles. Such features are introduced to compare two distinct sessions. Given:

- a number M of session typologies;
- a set U_i of users where each user u_{ij} provides a feedback $\phi_{ij}(k)$ with respect to a specific j^{th} session at time k in $\{1,\dots, k_{ij}\}$ for some session typology $i \in \{1,.., M\}$;
- for each session typology $i \in \{1,.., M\}$, one function structure in a family Ψ mostly suitable for *QoE* measurement model, including functional relationship among dynamic parameters of *QoS*;

If the user returns a feedback during the interval $[k-1, k]$, the returned value is associated to the k-th tuple. Otherwise, the value associated to the (k-1)-th tuple is used, if the user doesn't give an explicit feedback, the previous one is still valid. All the QoS measurements received before the first explicit user's feedback are not considered, the other is stored in a database and then processed.

The QoE Evaluation includes two main steps:

Step 1	The first step has the purpose of profiling the users in U_i, for each session typology i, namely it returns a partitional clustering $P_i = \{p_1, \ldots, p_{q_i}\}$ of U_i. The partitional clustering P_i allows to associate in the same group the users providing similar feedbacks in similar situations and, on the contrary, users considered sufficiently dissimilar from the behavioral point of view are associated in different groups. In the following, we denote with $pr_i(v)$ the most representative element of each group p_v for $v \in \{1,..,q_i\}$ (e.g. medioid, centroid) and we assume that it represents the behavioral profile of cluster v.
Step 2	The second step aims at estimating for each session typology $i \in \{1,..,M\}$ and for each profile $pr_i(v)$ the optimal parameters that characterize the function structure Ψ_i. The optimal choice of these parameters is made by analyzing a suitable set of session for each profile in order to automatically infer the correlation between the feedback and the QoS parameters. A proper choice of the function parameters for each profile $pr_i(v)$ allows to increase the quality of the QoE measurements for all the users belonging to p_v.

Table 6 QoE Evaluation steps

Each Profiled QoE function represents a behavioural profile of a class of similar users and when one of these users starts a new session, his/her behavioural profile (previously stored in the historical data) is used by the QoE Evaluator to measure the current QoE.

For a given session typology $i \in \{1,\ldots,M\}$, the partition of the users (User Profiling step) and the definition of a suitable Profiled QoE function (Identification step) can be performed "offline", for example, each time a predetermined number of new sessions is added. Moreover, in order to follow the variation of the behavioural profile of each user, you may want to give greater weight to the most recent sessions and less to older.

Step 1: User Profiling

The first step, namely the User Profiling, is based on the execution of a clustering algorithm (e.g. *k*-means++ [60]), that requires to be effective a proper choice of the features used to describe the session. Aims of this step is to identify relevant behavioural patterns. This phase is executed "off-line" and is repeated every time a certain number of fruition sessions are added to the database.

By restricting our attention to time series of parameters describing the $QoS_{ij}(k)$ and the sequence of user feedback $\phi_{ij}(k)$, the clustering algorithm identifies q_i groups of users U_i such that the sequences are similar for two users in the same group and dissimilar for two users in distinct groups.

$$< QoS_{ij}(1,k), \Phi_{ij}(1,k) >, k \in \{2, \dots, k_{ij}\} \tag{1}$$

To define a similarity measure to compare two distinct sequences (1) we refer to the two families of features (formerly introduced in [61]). This two families of features describes the individual characteristics of each single parameter in $QoS_{ij}(k)$ and the combined effects of parameters $QoS_{ij}(k)$ on the user feedback $\phi_{ij}(1, k)$, respectively.

In the first family of features, we consider the individual characteristics which are measured by the finite derivative ($TQoS_{ij}(k)$) and the discrete integral ($IQoS_{ij}(k)$) functions of each single parameter in $QoS_{ij}(k)$.

$$TQoS_{ij}(k) = \frac{\left(QoS_{ij}(k)\right) - \left(QoS_{ij}(k-1)\right)}{\left(k - (k-1)\right)} = \left(QoS_{ij}(k)\right) - \left(QoS_{ij}(k-1)\right) \tag{2}$$
$$k \in \{1, \dots k_{ij}\}$$

$$IQoS_{ij}(k) = \sum_{s=1}^{k} QoS_{ij}(s), \qquad k \in \{1, \dots k_{ij}\} \tag{3}$$

The first component $TQoS_{ij}(1)$ of the finite derivative feature defined in (2) is evaluated by setting $QoS_{ij}(0) = QoS_0$, where QoS_0 is the mean value of the $QoS_{ij}(k)$ parameter in $\{1,\dots, k_{ij}\}$. $TQoS_{ij}(k)$ has the scope of catching the trend of the discrete time series of parameter $QoS_{ij}(k)$ in *k*.

$IQoS_{ij}(k)$ is considered in order to take into account not only the QoS observed in *k*, but also what happened before time *k*.

As concerns the second family of features, we consider the *sensitivity* of the user u_{ij} with respect to a given parameter of QoS_{ij}. To increase the accuracy of the sensitivity, we evaluate the proportional effect of $TQoS_{ij}(k)$ on the user feedback $\phi_{ij}(k)$ in *k* and we *weight* this effect considering the user reaction time. In fact, we assume that the longer the reaction time, the less is the sensitivity manifested by the user.

To formally define the reaction time at *k* we consider the last interval $k'(QoS_{ij})$ when a change in the parameter QoS_{ij} occurred before *k*:

$$k'(k, QoS_{ij}) := \max\{s \in \{1, \dots, k\} : TQoS_{ij}(s)! \neq 0\}$$

Hence, the user reaction time is given by:

$$\Delta k := k - k'(k, QoS_{ij})$$

The sensitivity $SQoS_{ij}(k)$ of the user u_{ij} with respect to a given parameter of QoS_{ij} is defined as follows:

$$SQoS_{ij}(k) = \frac{\Phi_{ij}(k) - \Phi_{ij}(k-1)}{TQoS_{ij}(k'(k,\ QoS_{ij}))}\ \Delta k, \qquad k \in \{1, \dots k_{ij}\} \tag{4}$$

Once the features $TQoS_{ij}(k)$, $IQoS_{ij}(k)$ and $SQoS_{ij}(k)$ describing the finite derivative and the discrete cumulative effect as well as the sensitivity of the user u_{ij} with respect to each single parameter $QoS_{ij}(k)$) have been defined, the clustering algorithm measures the similarity between two distinct sequences (1) by evaluating the distance between the feature vectors defined as:

$$[TQoS_{ij}(k), IQoS_{ij}(k), SQoS_{ij}(k)]_{k=1}^{k_{ij}}$$

Then the second step of the QoE Evaluator comes into action.

Step 2: Function Characterization

The aim of this second step is that of completely identifying a function characterized by a structure f_i in Ψ and a finite set of parameters $\Omega_{f_{iv}}$ such that:

$$\begin{gathered} QoE_{ijv}(k) = f_i(\Omega_{f_{iv}}; QoS_{ij}(1,k), \Phi_{ij}(1,k-1) \\ u_{ij} \in p_v,\ v = 1, \dots q_i,\ k = 2, \dots k_{ij} \end{gathered} \tag{5}$$

In (5) the unknown measure $QoE_{ijv}(k)$ providing the level of quality experienced by user u_{ij} at time k in session h_{ij} of typology i is estimated by a function f_i depending on suitable set of parameters $\Omega_{f_{iv}}$, of QoS parameters at time k $QoS_{ij}(1,k)$ and of previous feedbacks $\phi_{ij}(1,k\text{-}1)$ from user u_{ij}. An important family of functions that links the QoS parameters to QoE measurements is reported in [62].

From the algorithmic point of view, given a session of typology i and the partitional clustering of the users in a given set U_i in q_i groups $\{p_1, \dots, p_{q_i}\}$ of similar users, we consider a finite set of sequences $<QoS_{ij}(1,k), \phi_{ij}(1,k\text{-}1)>$ for u_{ij} in a specific group p_v and estimate the optimal values $\Omega_{f_{iv}}{}^*$ of parameter $\Omega_{f_{iv}}$ of structure f_i in Ψ such that:

$$\begin{gathered} \Phi_{ij}(k) = f_i(\Omega^*_{f_{iv}}; QoS_{ij}(1,k), \Phi_{ij}(1,k-1) \\ u_{ij} \in p_v,\ v = 1, \dots q_i,\ k = 2, \dots k_{ij} \end{gathered} \tag{6}$$

Among all possible feasible solutions to problem (6), we select the one maximizing the generalization capability of f_i.

Once the optimal values $\Omega_{f_{iv}}$* have been estimated by finding a feasible solution to problem (6), $f_i(\Omega_{f_{iv}}$* approximates user feedback ϕ_{ij} that we consider as the main indicator of the QoE. Therefore, we define the function of measurement of Profiled Quality of Experience (QoE^P) as follows:

$$QoE^P_{ijv}(k) \coloneqq f_i(\Omega^*_{f_{iv}}; QoS_{ij}(1,k), \Phi_{ij}(1,k-1) \quad u_{ij} \in p_v,\ v = 1, \dots q_i,\ k = 2, \dots k_{ij} \tag{7}$$

Each Profiled QoE function represents a behavioural profile of a class of similar users and when one of these users starts a new session, his/her behavioural profile (previously stored in the historical data) is used by the QoE Evaluator to measure the current QoE.

For a given session typology i $\in$ {1,..,M}, the partition of the users (User Profiling step) and the definition of a suitable Profiled QoE function (Identification step) can be performed "offline", for example, each time a predetermined number of new sessions is added. Moreover, in order to follow the variation of the behavioural profile of each user, you may want to give greater weight to the most recent sessions and less to older.

4.4. QoE Controller

As previously described, when a user starts a new service session, the QoE Evaluator is in charge of identifying the most suitable Profiled QoE Function for him/her. This function, together with the real time QoS measurements for the observed session, provides inputs for the QoE Controller; and as shown in the overall architecture (Figure 1), the *Target QoE* completes the inputs required by the QoE Controller. The QoE Controller must deduce the Driving Parameters aiming at the satisfaction of the Personalized QoE Application Requirements, namely at the minimization, for each in progress application of its QoE Error, defined as the difference between the Target QoE and the Perceived QoE of the application in question.

The aim of the QoE-based Controller is to define for each user, identified by his/her Profiled QoE Function and *Target QoE*, a set of driving parameters, to guarantee QoE measurements aligned with the commercial profile. We assume that the QoE Controller determines a driving parameter for each QoS parameter (e.g. throughput, bandwidth, packet loss, etc) used to identify the Profiled QoE function. In this respect, it should be clear that the Target QoE Area identifies the constraints which should be met by the driving parameters (i.e. by the corresponding QoS parameters) to assure a satisfactory QoE to the user.

These constraints (personalized for each active session), will be sent to the Cognitive Network Control Functionalities, which is in charge of triggering the appropriate Network Control procedures (e.g. routing, congestion control, dynamic bandwidth allocation, etc.) aiming at achieving QoS performance respecting the above-mentioned constraints. Finally, the decisions taken by the above-mentioned Network Control procedures are actuated on the networks.

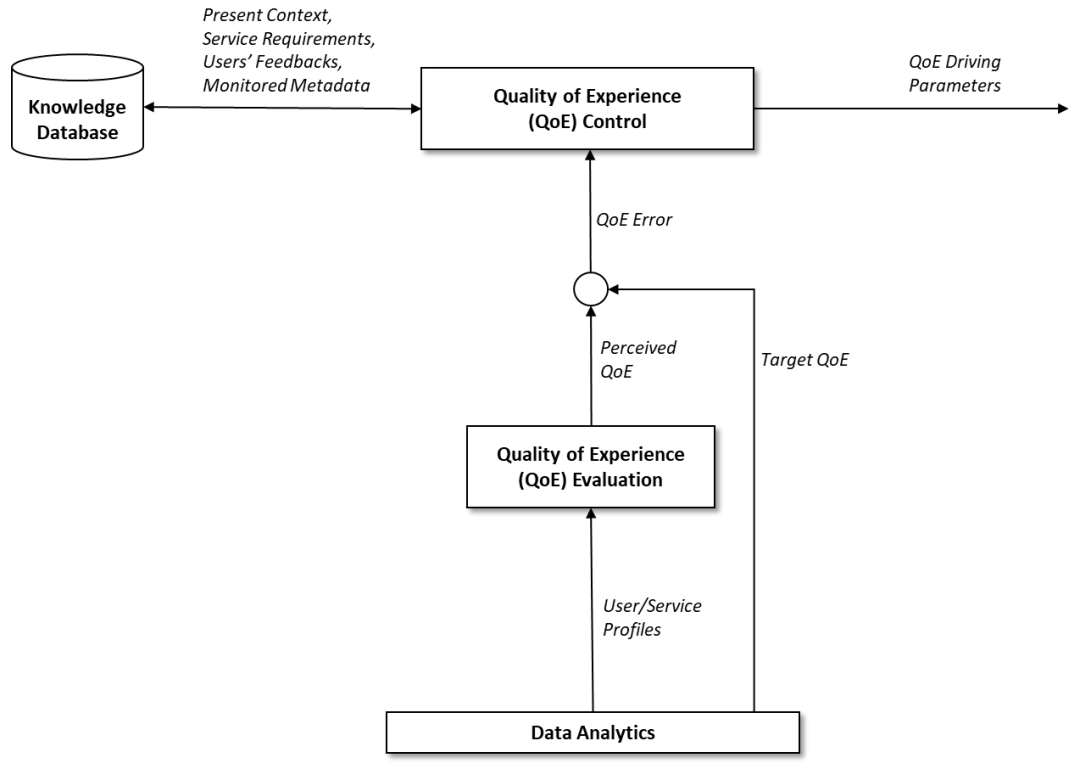

Figure 35 QoE Controller Interaction

As regards the QoE Driving Parameters, their nature depends on the considered service type and user profile. For instance, the QoE Driving Parameters may include, among others, QoS reference values (e.g., the tolerated transfer delay range, the minimum throughput to be guaranteed, the tolerated packet loss range, the tolerated dropping frequency range, etc.), as well as security and/or application-specific reference values (e.g., the desired encryption level, the allowed network nodes, the expected distribution of the offered traffic among the heterogeneous wireless access networks simultaneously covering a given user, etc.).

For the sake of simplicity, the case in which the QoE Driving Parameters are just QoS reference values is considered: hence, the QoE Controller has to dynamically decide, for each running application, the most appropriate QoS reference values which should actually drive the Perceived QoE as close as possible to the Target QoE.

However, since the control action has a large number of degrees of freedom, the exploration of the solution space may take a large amount of time, thus making the task of the QoE Controller excessively complex. A simpler (yet less fine-grained) control task arises if the management of the underlying networks is arranged into Classes of Service (CoS). Each CoS *c* is characterized by a set of QoS requirements which have to drive the network control functionalities to some desired performance results.

In this case, the role of the QoE Controller is to dynamically select, in real-time, the most appropriate CoS for the ongoing service sessions (i.e., the QoE Driving Parameters coincide with the Classes of Service associated to the running applications, as proposed in [64]), with the overall aim of reducing the personalized QoE Error.

To reach this goal, the QoE Controller should know – or, at least, estimate – the correlation between its decisions (the selected driving parameter) and the Perceived QoE in a given Present Context. In this respect, no model of the traffic flows in the network can be assumed, since the network behaviour depends on too many factors: traffic characteristics of the on-going applications, network topologies, network resource management algorithms, congestion control algorithms, and so on. The decision strategy must therefore be learned on-line by trial and errors. In this respect, this paper proposes Reinforcement Learning as the key technology to enable an organized on-line exploration of the possible decision strategies, named policies, and the exploitation of the best policy to be enforced.

The QoE Controller can be implemented by means of Agents (referred as QoE Agents) to be carefully embedded in properly selected network nodes (e.g., Base Stations and Mobile Terminals in a wireless environment). The proposed algorithm, referred to as single-agent learning, proposes that the decisions (i.e., the value of the Driving Parameters) are taken by each Agent based on its local knowledge of the Present Context and of the so-called Status Signal, which represents in a concise way the overall Network status, broadcast by a single centralized entity. the learning approach consists in a model-free adaptive feedback approach: the effects of the decisions are observed as a variation of the QoE Error, and the decisions are taken based on past-observations.

Below, the assumptions considered for the considered scenario:

- the telco operator assigns each user a range of expected *QoE* values which is referred to as *Target QoE*. Typically, this range of expected *QoE* is assigned considering the commercial profile of the user and other aspects not directly related to user feedbacks.
- the presence of a single centralized entity, named *Supervisor Agent* (SA) which sends the *Status Signal*, representing a concise description of the global network status which is broadcast by the SA at each discrete time instant. This paradigm perfectly fits the current trends in telecommunication networks, where a central orchestrator remotely manages the network entities (as in a Software Defined Network [63]).
- decisions (i.e., the choice of the values of the QoE Driving Parameters) are made by each UA at discrete time instants (periodically occurring and hereafter referred to as *time steps*), based on its local knowledge of the Present Context and on the so-called *Status Signal*.

The learning approach consists in a model-free adaptive feedback approach: the effect of the decisions are observed as a variation of the QoE Error, and the decisions are taken based on past-observations.

Reinforcement Learning (RL) is an interesting approach to solve the problem. RL focuses the attention on learning by the individual from directly interaction with its environment, without relying on complete model of the environment. The interaction, between agent and environment, is defined by formal framework (states, actions, rewards or costs) and the environment is typically formulated as a finite-state Markov Decision Process. The approach entails the presence of a centralized entity, which sends control signalling to the Agents. This approach is well-matched to the current trends in managing communication network, as with the Software Defined Network.

The problem is described by a Markov Decision Process, a tuple $\{X, A, pr, r\}$, where X is the finite state space, A is the finite set of agent actions, pr is the transition probability function, r is the one-step reward function. The state $x \in X$, that describes the environment, can be altered by the agent action $a \in A$. The environment changes state according to the state transition probabilities given by $pr(x, a, x')$. The reward evaluates the immediate effect of action a. The behavior of the agent is described by its policy π, which specifies how the agent chooses its actions given the state, it may be either stochastic, $\pi: X \times A \rightarrow [0,1]$, or deterministic, $\pi: X \rightarrow A$.

We consider a common reinforcement learning technique, known as Q-Learning, that works by learning the action-value function. It provides good policies before learning the optimal policy and is able to reacts to changes in the environment. The action-value function $Q^{\pi}(x, a)$ is the expected return starting from x taking action a, and thereafter following policy π; it satisfies the Bellman equation:

$$Q^{\pi}(x, a) = \sum_{x' \in X} pr(x, a, x') \left[r(x, a, x') + \gamma \max_{a' \in A} Q^{\pi}(x', a') \right]$$

where the discount factor $\gamma \in [0,1)$ weights immediate rewards versus delayed rewards.

Let $Q^{*}(x, a)$ be the optimal action-value function, defined as:

$$Q^{*}(x, a) = \max_{\pi} Q^{\pi}(x', a'), \forall x \in X, a \in A(x)$$

Then, the agent, computing $Q^{*}(x, a)$, can maximize its long-term performance, while only receiving feedback about its immediate, one-step performance. The greedy policy is deterministic and picks for every state the action with the highest Q-value:

$$\pi(x) = \arg \max_{a' \in A(x)} Q(x, a')$$

The Q-learning approach derives the policy on-line by estimating the (action, state)-values with the following update rules:

$$Q(x, a) \leftarrow \big(1 - \alpha(x)\big) Q(x, a) + \alpha(x) \left[r(x, a, x') + \gamma \max_{a' \in A} Q(x', a') \right]$$

where the learning rate $\alpha(x) \in [0,1]$ determine the convergence speed and accuracy.

4.4.1.1. Reinforcement Learning Based Control Rule

The personalized QoE Error, relevant to the user u and to the service f at time t, is defined as:

$$e^u(t,f) = QoE^u_{perc}(t,f) - QoE^u_{target}, \quad (1)$$

where $QoE^u_{perc}(t,f)$ represents the Perceived QoE at time t associated with user u when enjoying service f and forwarded by a QoE Evaluator instance; the QoE^u_{target} represents the Target QoE associated with user u.

If this error is positive, the running application is said to be overperforming, whereas, if the error is negative, the ongoing service is said to be underperforming. It is worthy of note that, should the error be positive, such a situation would be desirable only if the network were idle; conversely, should the network be congested, the fact that a given application is overperforming would not, in general, be desirable since it may happen that such an application is subtracting valuable resources to other applications which, in turn, are underperforming.

Therefore, the goal of the QoE Controller is to guarantee a nonnegative QoE Error for all users u, with the aim of avoiding the occurrence of underperforming applications. Furthermore, if it is not possible to guarantee a nonnegative QoE Error for all users (e.g., due to the lack of resources), the QoE Controller is nonetheless aimed at reducing, as far as possible, the QoE Errors of the various users by guaranteeing fairness among them.

Let us consider N UAs (i.e., end nodes), each requesting one specific application type and characterized by a personalized Target QoE level QoE^u_{target}, for $u = 1, \dots, N$.

As far as the computation of QoE^u_{target} is concerned, we now assume that, to each user u, the service operator assigns a reference level which is assessed by considering the commercial profile of the user. We also assume that the network supports C Classes of Service (CoS).

At each time step t, each UA u selects the most appropriate CoS to be associated with the requested application by running its own Q-Learning algorithm on the basis:

(i) of the Status Signal;

(ii) the measure of the Perceived QoE exploiting the local feedback on the service experienced in the currently selected CoS.

All such data comes from the Present Context.

In particular, the Status Signal includes which CoS each UA opted for at the preceding time step, whereas item measure of the Perceived QoE allows each UA to compute the expected QoE Error in the currently selected CoS. The control objective is to minimize the QoE Error (1) for each UA u.

The presented problem can be modelled by means of a Markov Decision Process (MDP) in the form $(\mathcal{S}, \mathcal{A}, \{\mathcal{P}^a_{ss'}\}, \gamma, \mathcal{R}_u)$, for each UA u:

1) The state space $\mathcal{S}$ describes the environment: assuming, without loss of generality, that all the considered UAs are active (i.e., they have an in-progress application instance) and that any application type can be assigned in a one-to-one fashion to each UA. The state $s(t)$ represents the vector of active nodes enjoying the service m, $m = 1, \dots, M$, using the class of service $c, c = 1, \dots, C$, at time t: $\mathrm{s}(t) = \left(n_{11}(t), \dots, n_{1C}(t), \dots, n_{M1}(t), \dots, n_{MC}(t)\right)$, where $n_{cm} = 0,1, \dots$; $c = 1, \dots, C$; $m = 1, \dots, M$. Thus, the finite state space is defined as $S = \{\mathrm{s} = (n_{mc}), c = 1, \dots, C; m = 1, \dots, M\}$

2) The action set represents, for each UA, the CoS selected for the transmission:

$$\mathcal{A}_u = \mathcal{A} = \{1, \dots, C\}, \quad u = 1, \dots, N.$$

3) The $\mathcal{P}^a_{ss'}$'s are the state transition probabilities, i.e., $\mathcal{P}^a_{ss'}$ is the probability of next state s' given state s and action a.

4) For each UA u, the reward r_{t+1}^{u} is defined by the opposite of the square of the QoE Error (1) of agent u, i.e.:

$$r_{t+1}^{u} = -\left(QoE_{meas}^{u}(t,f) - QoE_{target}^{u}\right)^{2}, \qquad u = 1, \dots, N.$$

In this case, each UA runs its own Q-Learning algorithm, i.e., particularizing the single-agent Q-Learning algorithm in [65] and [66] to the distributed case under scrutiny:

$$Q_u(s,a) \leftarrow (1-\alpha)Q_u(s,a) + \alpha\left[r_{t+1}^{u} + \gamma \max_{a' \in A} Q_u(s',a')\right] \tag{2}$$

for $u = 1, \dots, N$.

This recursive algorithm, for each UA u, can be shown to converge to the *optimal action-value function* Q_u^*, from which the *optimal policy* (i.e., π_u^* in the notation of [66]) can be easily determined.

According to [66], in order to address the *exploration* problem, an ε-greedy method is followed in the update rule (2):

- the UAs choose an action that has maximal estimated action value with probability $1-\varepsilon$ (where $\varepsilon \in (0,1)$ is the exploration rate), but with probability ε they select an action $a \in \mathcal{A}_u$ at random.

A large value of ε guarantees continual exploration of the (state-action) pairs, thus preventing each UA from remaining stuck in a local minimum; a small value of ε, on the other hand, lets each UA choose the best action based on the current estimates of the action-value function.

By varying the learning rate α, the exploration rate ε and the discount rate γ, the convergence velocity of the algorithm and the quality of the solution significantly change.

In the proposed solution, the continuous update of the Present Context (QoE Controller input) and the dynamic selection of the QoE Driving Parameters (namely, the CoSs for each UA) allow fast reaction to any kind of network impairment, thus producing improved performance results. Moreover, the fact that the Perceived QoE is a function of the Present Context entails network awareness, thus driving the network control design towards full cognition, as well as the fact that both the Present Context and the QoE Driving Parameters contain multi-network and multi-layer information drives the network control design towards *cross-layer optimization*.

4.5. Evaluation Results

The correctness of the proposed solution for the QoE Evaluator was validated with a proof of concept implementation against a real dataset from preliminary field trials within a video streaming service experiencing.

For what concerns the QoE Controller, a simple case study with the aim of providing a proof of concept of the QoE Control approach is introduced and discussed.

4.5.1. QoE Evaluator Validation and Results

A testbed with a video streaming service is considered. During the service fruition, the objective quality of the video (i.e. the video resolution and codec), vary in a random way; the user experiencing such video through the testbed can express a feedback using a MOS (Mean Opinion Scoree) scale value. Such data were collected, processed and stored in a data structure.

With reference to the video streaming service, there are different categories of factors influencing the QoE. There are technological factors (acting at network, device and application layer), environmental factors (e.g. user location and background), psychological factors (e.g. user expectations or mood) and technical factors (e.g. video resolution, framerate or decoding algorithm). The approaches in video quality assessment can be divided according to the characteristics of the QoE metrics [67], in:

- objective approaches: mathematical models based on the human visual system are created to assess the quality of videos without external interferences;
- subjective approaches: they are based on the perception of human observers. In a laboratory, a group of volunteers or selected people (crowd-sourcing) watch a video in different conditions and with different devices and give a score to the video. This kind of measure is affected by confounding factors. The most common metrics for service rating is the Mean Opinion Score (MOS) [68], a scale from 1 to 5: 1 (Bad), 2 (Poor), 3 (Fair), 4 (Good) and 5 (Excellent);
- hybrid approaches: it is a combination of the former approaches. A group of persons evaluates the degraded video and, based on the results of this evaluation, a prediction model is then formulated, using also other features, not strictly related to the video.

The subjective methods are expensive, time consuming, not repeatable and affected by confounding factors but manage to catch the human perception better than the objective approaches.

Considering the proposed solution provided by the QoE Evaluation and Control they form a suitable solution since they provide a hybrid approach, that estimates the subjective QoE metrics in an automatic, quantitative and repeatable manner, based on objective metrics and QoS parameters.

The quality of service (QoS) perceived by a generic user is subjective and depends on the user, but also by the service (e.g. a video streaming), the device (e.g. a tablet), the surrounding context (e.g. while on the move) and the emotional situation (e.g. relaxed). Even if there is no doubt on the fact that QoS parameters (like bandwidth, delay, jitter, and packet loss) directly affect the user QoE, anyway a reliable measurement system to provide a completely QoS based function representing the user QoE is still missing.

For these considerations, the role of explicit feedback from the user plays an important role. At this scope, the evaluation phase is based on a real dataset, collected during the field trial test, contains a collection of tuples composed by:

- the explicit user's feedback (expressed in MOS scale);
- the corresponding QoS configuration/measures;
- the service type;
- the user identifier.

The data have been processed to fit a proper data structure. The feedbacks have been aggregated on the basis of the user and service type.

Result of such case study experimenting is the automatic identification of distinct user profiles based on subjective user feedbacks while experiencing the service. Such significant features were automatically extracted from raw data time series by using a clustering algorithm for the identification of the most significant user profiles.

Based on a preliminary test in a video streaming scenario, we observed that all the users' behavioural profiles could be represented as a combination of three main elementary behaviours. We define as "elementary behaviour" the QoE function evaluated on the basis of the variation of a single QoS parameter. Without loss of generality, we assume that the QoE measurements and the QoS parameters are normalized in the range [0,1]. In Figure 36 the three identified elementary behaviours/profiles are shown.

The first one, namely Behaviour A, describes a user able to perceive each QoS mutation and to vary his/her perceived QoE accordingly. In the opposite way, Behaviour B describes a class of elementary behaviour where the QoE of the users is completely insensitive with respect to the specific QoS parameter variation. Finally, Behaviour C represents the class of user with a 'rigid' behaviour, or rather they are fully satisfied or not at all.

Obviously, all the previous defined behaviours represent a class containing more specific behaviours, each one with its own traits (e.g. slope, changing point, etc…). In this perspective, a Profiled QoE Function is a law that links and combines various elementary behaviours.

Figure 37 shows two Profiled QoE functions representing meaningful Behavioural Profiles. The first one (Behavioral Profile 1) represents a class of users having sensitive elementary behaviors with reference to both QoS parameters, even if with different intensity. The second profile represents the behavior of users that are totally insesitive to QoS Parameter 2, but they exhibit a 'rigid' attitude with respect to QoS Parameter 1.

Figure 38 shows an example of *Target QoE* reported over a Behavioral Profile: the colored area (hereafter referred to as *Target QoE Area*) identifies all values of the QoS parameters 1 and 2 corresponding to situations in which the user perceives an adequate QoE, accordingly with his/her commercial profile.

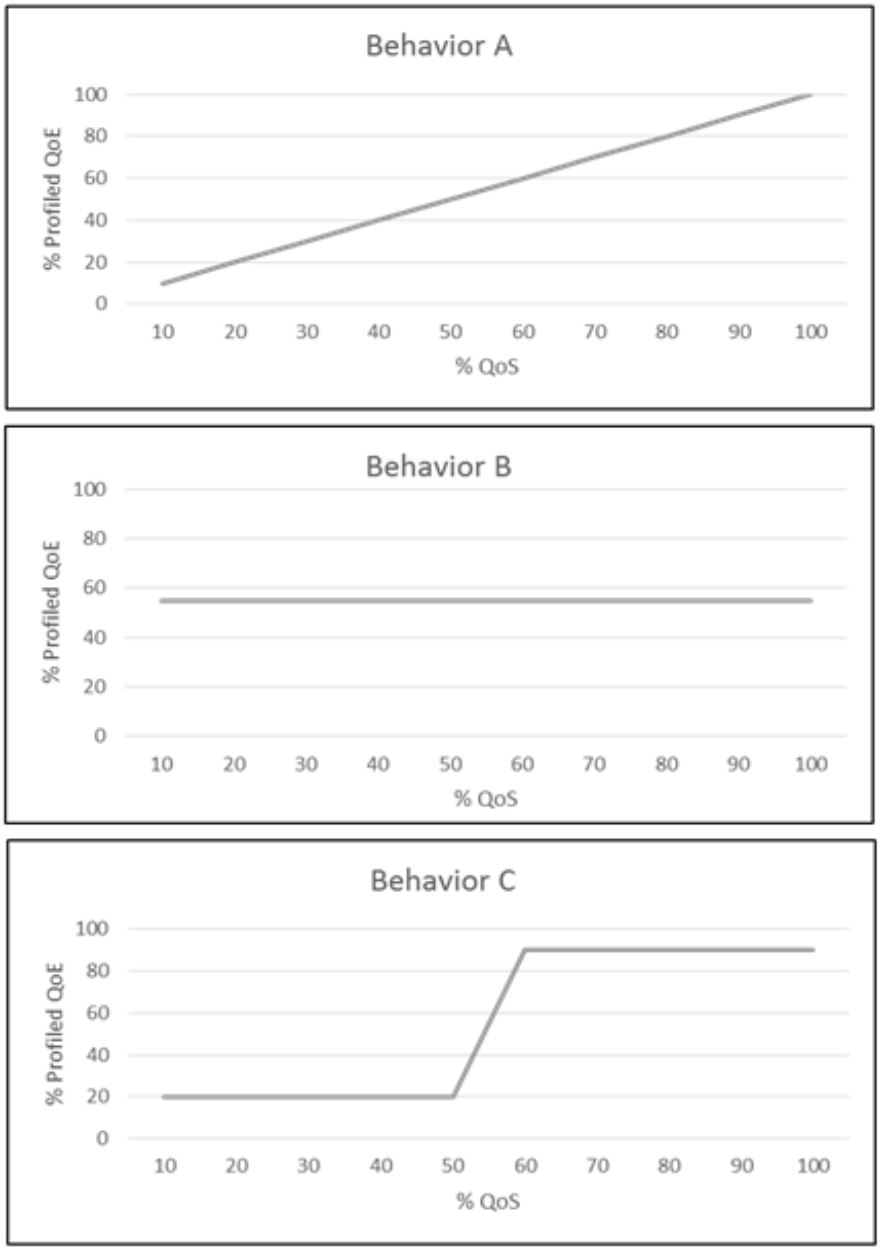

Figure 36 Basic User Behaviors/Profile Identified

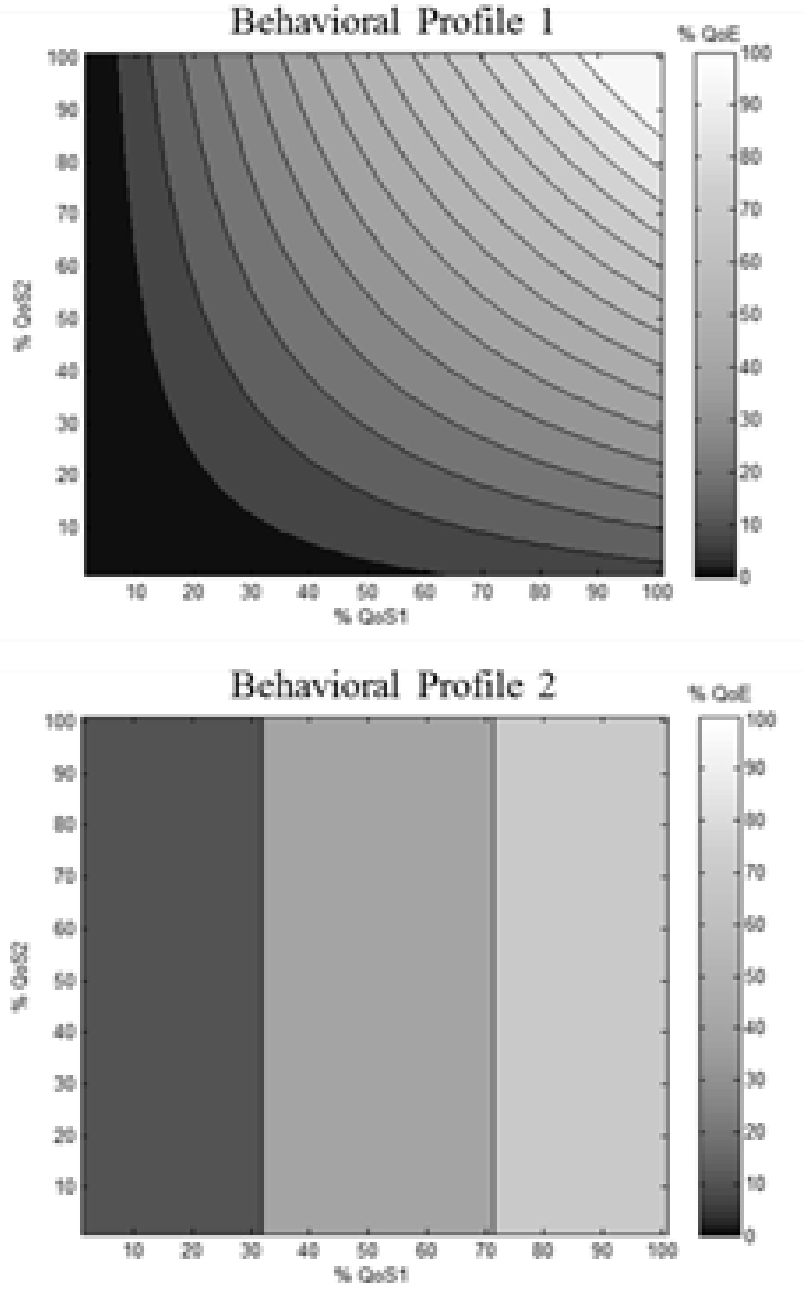

Figure 37 Identified Behavioural Profiles

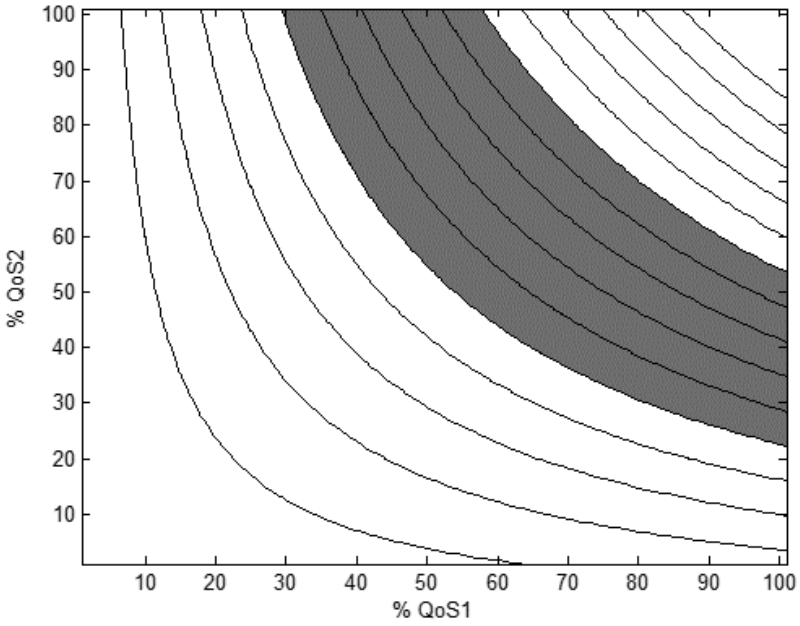

Figure 38 Example of Target QoE

4.5.2. QoE Controller Validation and Results

Regarding the QoE Control capabilities of the designed framework, numerical simulations were performed to evaluate the effectiveness of the proposed approach.

A simple case study was modelled to illustrate the potentials of the QoE Controller algorithms, it is assumed that the Driving Parameters are the CoSs, a small number of Agents, n = 8 and of CoSs, m = 3 is considered.

The data packets related to each application instance are transmitted to the first switch, which sends the packets to the second switch, which, in turn, sends them to the destinations (corresponding to the UAs).

Therefore, Router West multiplexes the traffic coming from the different sources over the bottleneck link, then Router East demultiplexes the aggregated traffic towards the corresponding destination links. The bottleneck link is the one between the two switches, and it is characterized by an available link capacity denoted by B_{link}.

The UAs and the SA are embedded in all of the destination nodes and in Router West, respectively.

The CoS is selected dynamically by the learning algorithms and the decisions occur at constant 1s time intervals. The network has a dumbbell (see Figure 39) configuration, *n* sources (the Agents) transmitting to a switch, which sends the packets to a second switch, which, in turn, send them to the destinations. The link between the switches is the bottleneck, characterized by the available link capacity B_{link}.

The network supports three different service classes (m = 3) and three different types of applications, each application is characterized by an average transmission bitrate.

Each Agent has its specific QoE function, which is computed based on the well-known IQX hypothesis:

$$f_{QoE} = \alpha \cdot e^{-\beta \cdot p_{QoS}} + \gamma$$

where α, β and γ are parameters to be tuned depending on the Application, in this case $\alpha = 1$, $\beta \in \{0.5, 0.7, 1\}$ depending on the application, and $\gamma = 0$. The IQX hypothesis is formulated with QoS as parameter (p_{QoS}), reflecting the objective service quality, in this case p_{QoS} is the error between requested and allocated bandwidth.

Different simulation environments were considered, in each one each end-node is characterized by a personalized target QoE level E_n, $n = 1, \dots, 8$, a random value taken in the set $\{0.7, 0.8, 0.9\}$.

During the simulation, each scenario is characterized by a given association between the Target QoE level and the type of application, characterized by the average transmission bitrate and denoted $b_n \in \{0.6, 1.2, 2\}$, $n = 1, \dots, 8$, furthermore the particular association between agent and type of application influences the scenario offered traffic load, computed as sum of average transmission bitrate of each end-node: $\eta_{off} = \sum_{n=1}^{N} b_n$.

To evaluate the overall algorithm performance in different traffic conditions, each scenario was simulated with three different value of available link capacity, in particular: $0.7\eta_{off}$, $0.8\eta_{off}$ and $0.9\eta_{off}$, denoted, respectively, *High*, *Medium* and *Low* traffic condition.

The CoS of an application is dynamically chosen according to its transmission rate requirement.

Each simulation run was executed two times: the first time, a heuristic static policy was implemented, referred to as *Greedy*, in which the CoS are permanently associated to the agent in a greedy fashion based on type of application; the second time, the policy computed by means of Single-agent RL algorithm (in particular the Q-learning approach) was implemented.

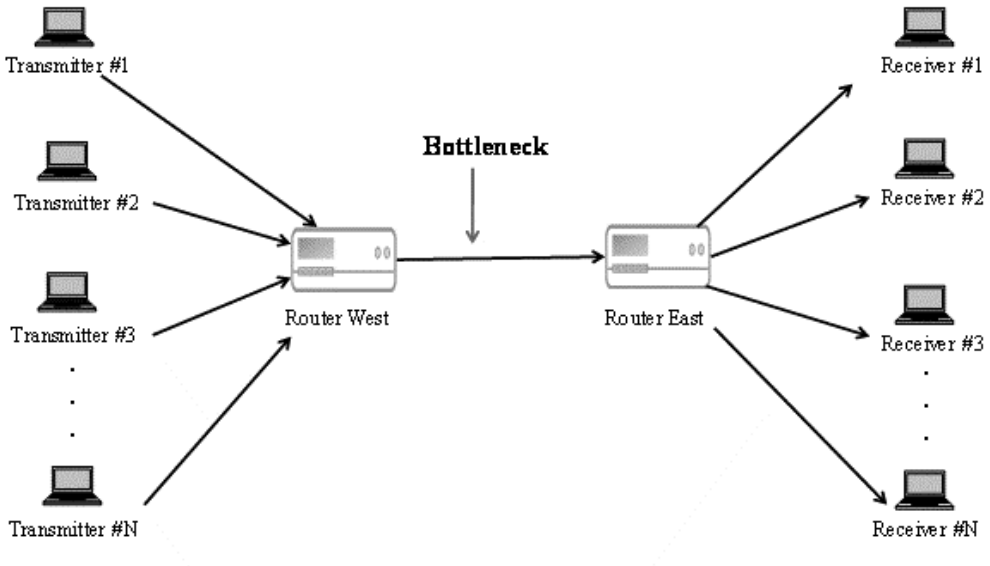

Figure 39 Dumbbell network topology as an application scenario.

For each scenario, ten simulation runs, each $2 \cdot 10^4$ steps long, were executed for each traffic condition. Each simulation carries out a comparison (in terms of the minimization of the performance) between the Q-Learning (Q-L) based approach and a heuristic static CoS assignment policy, in which CoSs are permanently associated to the users in a greedy fashion based on the application type.

Figure 40 shows the results of a single "sample" simulation (among all those that have been carried out and whose average results are shown in Figure 41) by plotting the Perceived QoE level for the eight considered UAs; the figure also shows the Target QoE values (in red). From the figure, it is clear that the Q-Learning based approach (in blue) yields better results (i.e., lower QoE Errors) than the static CoS assignment policy (in green).

Absolute QoE Error averaged over ten simulation runs and over ten different simulation scenarios per traffic load condition (namely, High, Medium and Low traffic conditions). The simulation results in terms of the absolute QoE Error averaged over the ten considered simulation scenarios per traffic load condition are shown in Figure 41. The obtained results show that the dynamic Q-Learning (Q-L) based approach outperforms the static policy. However, it should be noted that the amount of exchanged information is larger in the dynamic approach, whereas it is null, during the application lifetime, in the fixed policy case. In particular, the evaluation of the traffic overhead as well as the convergence time of the employed algorithms cannot be underestimated as they become crucial in larger scenarios. This may entail further research in the field of approximated Reinforcement Learning techniques.

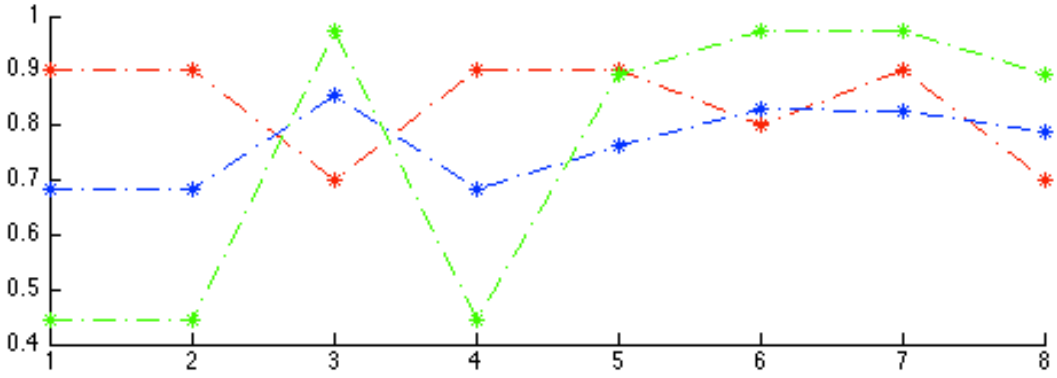

Figure 40 QoE Controller measures trends

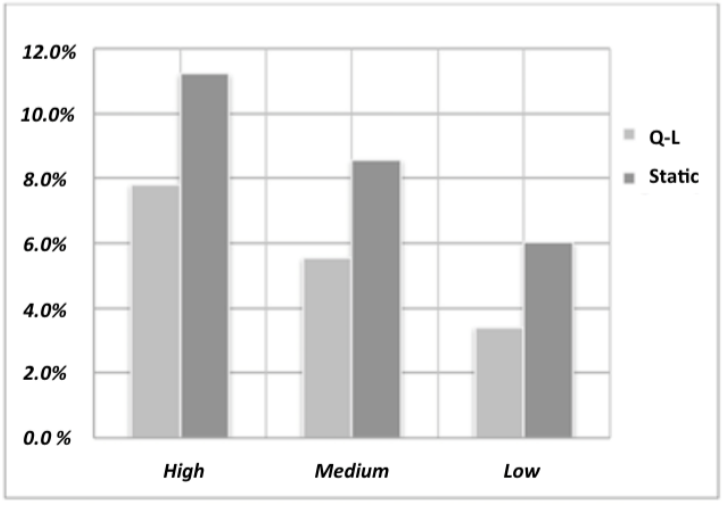

Figure 41 Absolute averaged QoE Error

4.6. Conclusion

A promising approach for coping with QoE Management in the Future Internet framework is presented. It defines a modular and cognitive architecture where the QoE Management functionalities consist of a QoE Evaluator and a QoE Controller, which can be designed independently from one another.

The potential impact of the presented approach is huge considering that it leads to a win-win strategy: the operators exploit their resources not to guarantee aseptic Service Level Agreements, but to optimize the customers' perception of the provided services, while the users enjoy satisfactory experiences in a seamless way with respect to QoS.

Regarding the QoE evaluation, it is able to identifies distinct user profiles on the base of subjective user feedbacks, gathered while they use generic services. The proposed solution extracts automatically suitable features, adopts off the shelf clustering algorithms and is independent of the specific service or QoS parameter. The proposed solution has been validated on the base of preliminary field trials.

Regarding the QoE Control framework, it enables a dynamical CoS selection aimed at reducing the error between the personalized Perceived QoE and the personalized Target QoE levels, by driving the underlying networks to efficiently exploit the available resources. This result has been obtained by embedding Reinforcement Learning algorithms in properly defined User Agents, which become thus able to make autonomous, consistent and effective decisions based on local and global feedback measurements.

The proposed approach presents several advantages: (i) it is fully compatible with the Future Internet architecture; (ii) it does not require any a-priori knowledge of the environment (i.e., it is model-free); (iii) QoE requirements can be personalized by properly selecting the functions for QoE computation and the corresponding Target QoE; (iv) it is scalable since communication and cooperation/consensus/negotiation procedures among agents are minimal.

It represents a concrete result of the technology and knowledge transfer applied by the author in the context of MIUR PLATINO research project.

Chapter 5
Conclusion and Future Works

The context of the next generation computer network, within the context the Future Internet initiatives is introduced, then, the main issues and limitation associated to the design and development of transparent, efficient, fair and flexible mechanisms for controlling network resources have been introduced and discussed, along with the reference architecture and requirements.

The mathematical formulation of both the problems (based on the game theory and on the concept of Wardrop equilibrium for the SDN Control Plane load balancer and on a mathematical model and on a Reinforcement Learning algorithm for the QoE Management) has been presented and discussed.

Model simulation and proof of concept implementation results have been then introduced and discussed. The results clearly show the validity of both the proposed approaches and their technological implementation feasibility.

At the network core, in order to address the performance issues related to the adoption of a distributed SDN Control Plane, a discrete-time, distributed, non-cooperative load balancing algorithm suitable for the and based on game theory and converged to a specific equilibrium known as Wardrop equilibrium with the aim of dynamically balancing the control traffic across a cluster of SDN Controllers is proposed. In particular, the proposed algorithm is aimed at dynamically learning the most efficient combination of flow rates from each controller, thus (i) ensuring the convergence to stable policies and consequently balancing the OpenFlow control traffic efficiently, (ii) increasing the overall throughput, and (iii) minimizing the control connection latencies.

The proposed Wardrop load balancing algorithm for SDN networks is proved to converge to an arbitrarily small neighbourhood of a Wardrop equilibrium. From an architectural point of view, SDN Proxies for the OpenFlow traffic are introduced to improve the scalability of SDN networks by dynamically dispatching the control workload across the available SDN Controllers. A proof-of-concept implementation on a real SDN network has been carried out and the related performance test results are reported. The Wardrop load balancing algorithm is proved to be able to improve the scalability of SDN networks by dynamically dispatching the control workload across the available SDN Controllers. The proposed approach results scalable, since no communications among the switches is needed and no centralized load balancing algorithm must be executed by the SDN Controllers.

On the other side, at the network edge, a modular and cognitive architecture where the QoE Management functionalities consist of a QoE Evaluator and a QoE Controller, which can be designed independently from one another cognitive framework in proposed; aimed at approaching the desired QoE level exploiting a mathematical model and methodology to identify a set of QoE profiles and an optimal and adaptive control strategy implemented by properly selected User Agents embedding a Reinforcement Learning based algorithm.

Regarding the QoE evaluation, it is able to identifies distinct user profiles on the base of subjective user feedbacks, gathered while they use generic services. The proposed solution extracts automatically suitable features, adopts off the shelf clustering algorithms and is independent of the specific service or QoS parameter. The proposed solution has been validated on the base of preliminary field trials.

Regarding the QoE Control framework, it enables a dynamical CoS selection aimed at reducing the error between the personalized Perceived QoE and the personalized Target QoE levels, by driving the underlying networks to efficiently exploit the available resources. This result has been obtained by embedding Reinforcement Learning algorithms in properly defined User Agents, which become thus able to make autonomous, consistent and effective decisions based on local and global feedback measurements.

The proposed approach presents several advantages: (i) it is fully compatible with the Future Internet architecture; (ii) it does not require any a-priori knowledge of the environment (i.e., it is model-free); (iii) QoE requirements can be personalized by properly selecting the functions for QoE computation and the corresponding Target QoE; (iv) it is scalable since communication and cooperation/consensus/negotiation procedures among agents are minimal. The potential of the proposed approaches is high considering that it provides advantages both to the service provider (exploit their resources not to guarantee aseptic Service Level Agreements, but to optimize the customers' perception of the provided services) both the end-user (users enjoy satisfactory experiences in a seamless way with respect to QoS).

Many interesting future works could depart from this thesis, both of theoretical and practical nature:

- improve the convergence properties of the Wardrop algorithm and validate the algorithm on larger use cases;
- improve multi-agent QoE Management coordination with minimal information exchange.

Glossary of Main Terms

Acronym	Description
5G	5th (Fifth) Generation
ACL	Access Control List
API	Application Programming Interface
BSS	Base Station Subsystem
CAPEX	Capital Expenditure
CoS	Class of Service
CP	Control Plane
DP	Data Plane
ETSI	European Telecommunication Standards Institute
HIDS	Host-based Intrusion Detection System
ICT	Information Computer Technologiy
IoT	Internet of Things
ISG NFV	Industry Specification Group for NFV
IT	Information Technology
ITU-T	International Telecommunication Union – Telecommunication Standardization Bureau
L2	Layer 2
L3	Layer 3
LTE	Long Term Evolution
MAC	Media Access Control
MOS	Mean Opinion Score
NF	Network Function
NFV	Network Function Virtualization

NFVI	Network Function Virtualization Infrastructure
NS	Network Service
ODL	OpenDaylight
OF	OpenFlow
ONF	Open Network Foundation
OPEX	Operating Expense
OSS	Operational Support System
OVS	Open Virtual Switch
OVSDB	Open vSwitch Database Management Protocol
QoE	Quality of Experience
QoS	Quality of Service
REST	Representational State Transfer
RL	Reinforcement Learning
SA	Supervisor Agent
SDN	Software Defined Network
SNMP	Simple Network Management Protocol
SSH	Secure SHell
TLS	Transport Layer Security
UA	User Agent
vCPU	Virtual CPU

References

[1] Platform for Innovative Services in Future Internet," Italian Ministry of University and Research (MIUR) PLATINO project, (Grant Agreement no. PON01_01007), http://www.progettoplatino.it/ [Online].

[2] "FI-WARE (Future Internet Ware), EU FP7-ICT Large-scale Integrating Project (IP), 2011-2014, grant agreement no. 312826, http://www.fi-ware.eu/.," [Online].

[3] "T-NOVA EU FP7-ICT Project. Grant agreement no 619520," [Online]. Available: www.t-nova.eu.

[4] "FI-PPP - Future Internet Public Private Partnership," [Online]. Available: https://www.fi-ppp.eu.

[5] "EU Horizon 2020 Research and Innovation programme," [Online]. Available: https://ec.europa.eu/programmes/horizon2020/.

[6] ITU-T, "Recommendation P.10/G.100 New Appendix I - Definition of quality of Experience (QoE)," 2006.

[7] Open Network Foundation, "Software-defined networking: The new norm for networks," 2012. [Online]. Available: https://www.opennetworking.org.

[8] OpenDaylight - Linux Foundation, "OpenDaylight SDN Platform," [Online]. Available: https://www.opendaylight.org/.

[9] Open Network Foundation, "OpenFlow Switch Specification, Version 1.5.0 (Protocol version 0x06)," 2014.

[10] ETSI, "Network Functions Virtualisation. An Introduction, Benefits, Enablers, Challenges & Call for Action," in *SDN and OpenFlow World Congress* , Darmstadt-Germany, 2012.

[11] ETSI, "Network Functions Virtualisation (NFV); Management and Orchestration," 2014.

[12] K. e. al., "The nature of data center traffic: measurements & analysis," *Proceedings of IMC ACM,* pp. 202-208, 2009.

[13] A. a. C. M. a. K. T. a. S. S. Tavakoli, "Applying NOX to Datacenter," *Proc. ACM SIGCOMM HotNets,* 2009.

[14] W.-D. W. X.-Y. G. X.-R. Q. a. S.-D. C. Y.-N. Hu, "On the placement of controllers in software-defined networks,"," *J. China Univ. Posts Telecommun,* vol. 19, 2012.

[15] S. L. e. al., "Heuristic Approaches to the Controller Placement Problem in Large Scale SDN Networks," *IEEE Transactions on Network and Service Management,* vol. 12, 2015.

[16] M. S.-H. A. Sallahi, "Optimal Model for the Controller Placement Problem in Software Defined Networks," *IEEE Communications letters,* vol. 19, 2014.

[17] B. a. S. R. a. M. N. Heller, "The controller placement problem," *In Proc. ACM SIGCOMM, HotSDN,* 2012.

[18] D. Erickson, "The beacon OpenFlow Controller," *In Proc. ACM SIGCOMM, HotSDN,* 2013.

[19] M. C. N. G. J. S. L. P. M. Z. R. R. Y. I. H. I. T. H. a. S. S. T. Koponen, "Onix: A distributed control platform for large-scale production networks," *OSDI,,* 2010.

[20] M. G. J. H. Y. H. M. K. T. K. B. L. B. O. P. R. W. S. a. G. P. P. Berde, “ONOS: towards an open, distributed SDN OS,",” *Proceedings of the third workshop on Hot topics in software defined networking*, 2014.

[21] F. H. S. M. T. V. L. a. R. R. K. Advait Abhay Dixit, “Towards an elastic distributed SDN controller,” *HotSDN,*, 2013..

[22] Mininet, “Mininet: An instant virtual network on your laptop,” 2014. [Online]. Available: http://mininet.org/..

[23] openvswitch, “Production quality, multilayer open virtual switch,” 2014. [Online]. Available: http://openvswitch.org/.

[24] B. P. a. B. Davie, “The open vswitch database management protocol. RFC 7047,” 2013.

[25] “A Dixit, F Hao, S Mukherjee, T.V. Lakshman, R Kompella. Towards an Elastic Distributed SDN Controller. acm sigcomm 2014”.

[26] “D. Grosu, A. T. Chronopoulos, and M. Y. Leung, “Cooperative load balancing in distributed systems,” Concurr. Comput. Pract. Exp., vol. 20, no. 16, pp. 1953–1976, Nov. 2008.”.

[27] “S. U. Khan and I. Ahmad, “A Cooperative Game Theoretical Technique for Joint Optimization of Energy Consumption and Response Time in Computational Grids,” IEEE Trans. Parallel Distrib. Syst., vol. 20, no. 3, pp. 346–360, Mar. 2009.”.

[28] “R. Subrata, A. Y. Zomaya, and B. Landfeldt, “A Cooperative Game Framework for QoS Guided Job Allocation Schemes in Grids,” IEEE Trans. Comput., vol. 57, no. 10, pp. 1413–1422, Oct. 2008.”.

[29] “D. Grosu and A. T. Chronopoulos, “Noncooperative load balancing in distributed systems,” J. Parallel Distrib. Comput., vol. 65, no. 9, pp. 1022–1034, Sep. 2005.”.

[30] “R. Subrata, A. Y. Zomaya, and B. Landfeldt, “Game-Theoretic Approach for Load Balancing in Computational Grids,” IEEE Trans. Parallel Distrib. Syst., vol. 19, no. 1, pp. 66–76, Jan. 2008.”.

[31] “E. Even-Dar, A. Kesselman, and Y. Mansour, Convergence time to Nash equilibria. Springer-Verlag, Berlin Heidelberg, 2003.”.

[32] “E. Even-Dar and Y. Mansour, “Fast convergence of selfish rerouting,” in 16th annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics, 2005, pp. 772–781.”.

[33] “S. Shah and R. Kothari, “Convergence of the dynamic load balancing problem to Nash equilibrium using distributed local interactions,” Inf. Sci. (Ny)., vol. 221, pp. 297–305, Feb. 2013.”.

[34] “H. Ackermann, S. Fischer, M. Hoefer, and M. Schöngens, “Distributed algorithms for QoS load balancing,” Distrib. Comput., vol. 23, no. 5–6, pp. 321–330, Dec. 2010.”.

[35] “J. G. Wardrop, “Some Theoretical Aspects of Road Traffic Research,” ICE Proc. Eng. Div., vol. 1, no. 3, pp. 325–362, Jan. 1952.”.

[36] “H. Kameda, J. Li, C. Kim, and Y. Zhang, Optimal Load Balancing in Distributed Computer Systems. London: Springer London, 1997.”.

[37] “S. Fischer, H. Räcke, and B. Vöcking, “Fast Convergence to Wardrop Equilibria by Adaptive Sampling Methods,” SIAM J. Comput., vol. 39, no. 8, pp. 3700–3735, Jan. 2010.”.

[38] “D. Barth, O. Bournez, O. Boussaton, and J. Cohen, “Distributed learning of Wardrop equilibria,” in Lecture Notes in Computer Science, vol. 5204, 2008, pp. 19–32.”.

[39] “F. Cimorelli, F. Delli Priscoli, A. Pietrabissa, L. Ricciardi Celsi, V. Suraci, and L. Zuccaro, “A Distributed Load Balancing Algorithm for the Control Plane in Software Defined Networking,” in Proceedings of the 24th Mediterranean Conference on Control a”.

[40] “M. Beckmann, C.B. McGuire, C.B. Winsten, Studies in the Economics of Transportation. Yale University Press, New Haven, CT, 1956.”.

[41] “T. Roughgarden and E. Tardos, “How bad is selfish routing?,” J. ACM, 49(2): 236–259, 2002.”.

[42] “S. Fischer, B. Vöcking, “Adaptive routing with stale information,” Theoretical Computer Science, vol. 410, no. 36, 2009, pp. 3357-3371.”.

[43] “G. A. Shanholt, “Set stability for difference equations,” International Journal of Control, vol. 19, no. 2, pp. 309-314, 1974.”.

[44] “S. Fischer, B. Vöcking, “Adaptive routing with stale information,” Theoretical Computer Science, vol. 410, no. 36, 2009, pp. 3357-3371.”.

[45] “wmod17”.

[46] “S. Khirman and P. Henriksen, “Relationship between quality-of-service and quality-of-experience for public internet service,” in Proc. 3rd Workshop on Passive and Active Measurement, Fort Collins, Colorado, USA, March 2002.”.

[47] “P. Brooks and B. Hestnes, “User measures of quality of experience: why being objective and quantitative is important,” IEEE Network, vol. 24, no. 2, pp. 8-13, 2010.”.

[48] J. W. M Alreshoodi, “Survey on QoE\QoS correlation models for multimedia services,” *International Journal of Distributed and Parallel Systems (IJDPS),* 2013.

[49] “M. Fiedler, T. Hossfeld, and P. Tran-Gia, “A generic quantitative relationship between quality of experience and quality of service,” IEEE Network, vol. 24, no. 2, pp. 36-41, 2010.”.

[50] “S. Singh, J.G. Andrews, G. de Veciana, “Interference Shaping for Improved Quality of Experience for Real-Time Video Streaming”, IEEE Journal on Selected Areas of Communications, Vol. 30, Issue 7, Pg. 1259 – 1269, 2012.”.

[51] “S. Jelassi, G. Rubino, H. Melvin, H. Youssef, G. Pujolle, “Quality of Experience of VoIP Service: A Survey of Assessment Approaches and Open Issues”, IEEE Communications Surveys & Tutorials, Vol. 14, Issue 2, 2012.”.

[52] M. I. A. P. V. S. F. Delli Priscoli, “Modelling Quality of Experience in Future Internet Networks,” *Future Network & Mobile Summit 2012, Berlin, July 2012.*.

[53] “D. Kotsakos, G. Trajcevski, D. Gunopulos, C. Aggarwal, “Time-series Data Clustering”, in Data Clustering: Algorithms and Application, CRC Press, 2013.”.

[54] “H. A. Tran and A. Mellouk, “Qoe model driven for network services”, Wired/Wireless Internet Communications, pp. 264–277, 2010.”.

[55] “M.S. Musthaq, B Augustin, and A.Mellouk, “Empirical Study based on Machine Learning Approach to Asses the Qos/qoe Correlation”, Proc. 17th European Conf. Networks and Optimal Comm.,pp. 1-7, 2012.”.

[56] “Oliver Hohlfeld, Rüdiger Geib, Gerhard Haßlinger, “Packet loss in real-time services: Markovian models generating QoE impairments”, Proc. of the 16th International Workshop on Quality of Service, IWQoS, June 2008, pp. 239–248.”.

[57] “F. Delli Priscoli, M. Iannone, A. Pietrabissa, V. Suraci, “Modelling Quality of Experience in Future Internet Networks”, Future Network & Mobile Summit 2012, Berlin, pp. 1-9, July 2012.”.

[58] “M. Fiedler, T. Hossfeld, P. Tran-Gia. “A generic quantitative relationship between Quality of Experience and Quality of Service”, IEEE Network, vol. 24, no. 2, Mar./Apr. 2010.”.

[59] “T. Hoßfeld, D. Hock, P. Tran-Gia, K. Tutschku, M. Fiedler, “Testing the IQX Hypothesis for Exponential Interdependency between QoS and QoE of Voice Codecs iLBC and G.711”, 18th ITC Specialist Seminar on Quality of Experience, Karlskrona, Sweden, May 2008.”.

[60] “Arthur D., Vassilvitskii S., k-means++: the advantages of carefull seeding, in Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, 1027-1035, 2007.”.

[61] “S. Canale, F. Facchinei, R. Gambuti, L. Palagi, and V. Suraci, “User profile based Quality of Experience”, 18th International Conference on Circuits, Systems, Communications and Computers (CSCC 2014), Santorini Island, Greece, July 17-21, 2014.”.

[62] “T. Hoßfeld, D. Hock, P. Tran-Gia, K. Tutschku, M. Fiedler, “Testing the IQX Hypothesis for Exponential Interdependency between QoS and QoE of Voice Codecs iLBC and G.711”, 18th ITC Specialist Seminar on Quality of Experience, Karlskrona, Sweden, May 2008.”.

[63] “A. Palo, L. Zuccaro, A. Simeoni, V. Suraci, L. Musto, and P. Garino, “A Common Open Interface to Programmatically Control and Supervise Open Networks in the Future Internet,” Future Networks & Mobile Summit 2013, Lisbon, July 2013.”.

[64] “P. Giaccone, B. Prabhakar, and D. Shah, “Randomized scheduling algorithms for high-aggregate bandwidth switches,” IEEE Journal on Selected Areas in Communications, vol. 21, no. 4, pp. 546-559, 2003.”.

[65] “C.J.C.H. Watkins and P. Dayan, “Q-Learning,” Machine Learning, vol. 8, no. 3, pp. 279-292, 1992.”.

[66] “R.S. Sutton and A.G. Barto, Reinforcement Learning: An Introduction. MIT Press, Cambridge, Massachusetts, 1998.”.

[67] “Bentes Maia O., Yehia H. C., De Errico L., “A concise review of the quality of experience assessment for video streaming”, Computer Communications , Volume 57, 15 February 2015, Pages 1–12, DOI: 10.1016/j.comcom.2014.11.005”.

[68] “ITU-T Recommendation BT 500-13, “ Methodology for the Subjective Assessment of the Quality of Television Pictures”, 2012.”.

[69] “N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: enabling innovation in campus networks,” SIGCOMM Comput. Commun. Rev., vol. 38, pp. 69-74, Mar. 2008.”.

[70] “EU 5G Public Private Partnership 5G-PPP. https://5g-ppp.eu/ [Online].,” [Online].