Efficient and Reliable Management of IoT-Based Services and Big Data in SDN-Based Smart Environments

by

Yosra NJAH

MANUSCRIPT-BASED THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
Ph.D.

MONTREAL, DECEMBER 10, 2021

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

ACKNOWLEDGEMENTS

First and foremost, I would like to praise and thank God, the almighty, who has granted me countless blessings, genius, perseverance, and patience so that I have been finally able to accomplish my Ph.D degree.

Moreover, this thesis becomes a reality with the valuable supports of many people in Tunisia, Canada, and Switzerland. Thus, I take this opportunity to express my sincere gratitude to all of them.

My first loving thanks go to my wonderful parents, Jamel and Hassiba, without whom I would not have dare thinking of achieving any of my dreams. I am forever indebted to you for giving me the opportunities and experiences that have made me who I am. I am infinitely thankful for your tireless sacrifices, your incomparable supports, your limitless listening, your precious advice, your infinite patience, and unparalleled love. I am so blessed and proud to be your daughter! I am also blessed to have caring and delightful four sisters, Bouthaina, Rahma, Khouloud, and Yasmine: I love you! Special thanks to my three brothers-in-law for their kind support during these years of my Ph.D. program. Kisses to my nephews and nieces, Farah, Mohamed-Jamel, Areej, Massarah, and Wouroud, who made this journey full of laughter and happiness. I also reserve special thanks to my entire family, particularly my grandparents, who were always supporting and encouraging me with their love and best wishes.

A very special gratitude goes out to my advisor Prof. Mohamed Cheriet for the continuous support of my Ph.D. study and related research. My sincere appreciation for all his distinctive acts in good and hard times. I also wish to express my gratitude to my thesis committee, Prof. Abdelouahed Gherbi, Prof. Aris Leivadeas, and Prof. Soumaya Cherkaoui, who agreed to review and evaluate this research. A special mention of thanks to my former professor Jean Hennebert, from the Haute École d'Ingénierie et d'Architecture de Fribourg (HEIA-Fr) in Switzerland, who supported collaborations with his master students and encouraged me to perform the co-supervision process. I would like to thank as well all my colleagues and friends who were always willing to help and give their best suggestions. My gratitude extends to the Natural Sciences and Engineering Research Council (NSERC) of Canada and the Canada Research Chair, Tier 1, held by Prof. Mohamed Cheriet, for the funding opportunity to undertake my studies at the department of systems engineering, École de Technologie Supérieure (ÉTS). Finally, I'm grateful to all the ÉTS staff and people of Canada, who, as a university and a country, have generously offered me all the necessary rights, respect, peace, and safe environment to accomplish my Ph.D. degree.

Yosra NJAH

Gestion efficace et fiable des services basés sur l'IdO et du big data dans des environnements intelligents basés sur SDN

Yosra NJAH

RÉSUMÉ

Récemment, des organisations du monde entier ont adopté des technologies numériques intelligentes afin d'améliorer la qualité de vie dans des environnements intelligents et durables (ex., maisons, campus, usines, hôpitaux, véhicules, villes, etc.). L'Internet des Objets (IdO) est l'une des technologies habilitantes les plus prometteuses pour le déploiement des environnements intelligents en créant un réseau mondial composé d'une pléthore d'objets physiques interconnectés et intégrés aux électroniques, aux logiciels, aux capteurs, et aux connectivité des réseaux, offrant ainsi de grandes opportunités et avantages, tel que la commodité, l'automatisation, la flexibilité, et l'intelligence. Cependant, cette expansion explosive des appareils mobiles, des capteurs, des services infonuagiques, et du trafic vidéo a soulevé des défis inouïs pour les administrateurs de réseaux au niveau du développement des solutions avancées assurant une gestion efficace et fiable de l'infrastructure sous-jacente limitée et du big data correspondants, sous forme d'innombrables services IoT et non IoT hétérogènes. En second lieu, le réseau traditionnel a une visibilité globale limitée de l'architecture tout entière ainsi que les ressources correspondantes disponibles en raison du paradigme d'accouplement entre les plans de contrôle et de données. Les réseaux définis par logiciels (réseaux SDN) est une technologie prometteuse qui fournit un modèle centralisé purement logiciel où le plan de contrôle est totalement découplé du plan de données permettant ainsi le contrôle à distance et la configuration dynamique de toutes les ressources et services réseau hétérogènes.

Pour concevoir une gestion efficace et fiable des environnements intelligents et durables, qui est le but de cette thèse, nous étudions l'hypothèse de développement de mécanismes génériques basées sur la technologie SDN permettant de contrôler et optimiser les ressources réseau de l'infrastructure sous-jacente et les flux de données massifs, simultanés, et hétérogènes. Pour atteindre cet objectif, quatre questions clés sont abordées dans le système que nous proposons et sont résumées comme suit: (i) Quelles sont les particularités essentielles des différents environnements intelligents pour concevoir des engins génériques basés sur la technologie SDN? (ii) Comment concevoir des mécanismes de routage supportant le provisionnement de la Qualité de Service (QdS) dans les réseaux intelligents? (iii) Comment concevoir des mécanismes de routage supportant l'optimisation de l'allocation des ressources réseau limitées? Et (iv) Pourquoi/Comment/Où l'analyse et la caractérisation du trafic (ex., en termes de QdS) doivent-elles être installées dans des architectures entièrement programmables?

Pour aborder le but global de cette thèse et les questions de recherche mentionnées ci-dessus, nous fixons cinq objectifs qui nous permettent de concevoir et de valider expérimentalement de nouvelles fonctionnalités réseau liées à l'analyse du trafic, au provisionnement de la QdS, à l'allocation des ressources, et à la consommation d'énergie. En outre, différents cas d'utilisation d'environnements intelligents (ex., industriel et campus éducatif) ont été considérés comme cas d'études afin d'examiner leurs caractéristiques et exigences conformément à plusieurs perspectives, validant ainsi la communité des fonctionnalités réseau étudiés. Avec chaque environnement, on commence par la revue des services hétérogènes et l'architecture réseau sous-jacente, où on se concentre particulièrement sur les aspects de "softwarization" et de programmabilité. Ensuite, en fonction de l'objectif de chaque cas d'étude, nous concevons avec des systèmes de contrôle centralisés (réactifs et proactifs) des modèles mathématiques et des algorithmes utilisant la théorie de la relaxation lagrangienne et permettant de gérer un ensemble de milliers de flux hétérogènes tout en assurant la QdS requise, en optimisant les ressources disponibles, en réduisant la consommation d'énergie, et en minimisant le coût de routage. En outre, nous démontrons comment les fonctionnalités d'analyse du trafic et de classification de services jouent un rôle important dans la conception d'algorithme de routage approprié ainsi que la gestion de la performance du réseau. Par conséquent, on se concentre sur l'analyse du trafic hétérogène dans les environnements intelligents, en particulier la différenciation des flux IdO et non-IdO. On propose un cadre d'évaluation extensif basé sur des mécanismes d'apprentissage automatique et en profondeur pour une identification fine des services tout en préservant la confidentialité des utilisateurs et en supportant la nature évolutive du trafic réseau, la haute précision de classification et la faible complexité de calcul. Les résultats de notre étude montrent que le déploiement de mécanismes génériques pour les différents environnements intelligents est possible avec certains modules (ex., récupération de l'infrastructure, analyse et caractérisation du trafic, etc.), tandis que d'autres modules (ex., provisionnement de la QdS, optimisation des ressources, etc.) doivent être ajustés en fonction de l'objectif de l'application. De plus, les mécanismes proposés basés sur SDN considérablement améliorent le provisionnement de la QdS, optimisent l'allocation des ressources et réduisent la consommation d'énergie par rapport aux travaux existants.

Mots-clés: environnements intelligents, Internet des Objets (IdO), réseau définie par logiciel (SDN), multi-programmabilité, flux simultanés, big data, analyse du trafic, caractérisation des services, apprentissage automatique et en profondeur, allocation distribuée des débits, Qualité de Service (QdS), optimisation des ressources, sensibilisation à l'énergie.

Efficient and Reliable Management of IoT-Based Services and Big Data in SDN-Based Smart Environments

Yosra NJAH

ABSTRACT

Recently, organizations worldwide have been embracing intelligent digital technologies in order to boost the quality of living within smart-sustainable environments (e.g., homes, campuses, factories, healthcare, vehicles, cities, etc.). The Internet of Things (IoT) is one of the most promising enabling technologies for deploying these environments by creating a worldwide network of a plethora of interconnected physical objects embedded with electronics, software, sensors, and network connectivity, bringing thereby tremendous opportunities and benefits, including convenience, automation, flexibility, and intelligence. However, this explosive expansion of mobile and sensing devices, cloud services, and video traffic has raised unprecedented challenges for network administrators in advanced solutions development to ensure efficient and reliable management of the underlying constrained infrastructure and the corresponding big data, as innumerable heterogeneous IoT and non-IoT services. On the other hand, the traditional network has limited global visibility of the overall architecture and the corresponding available resources because of the coupled control and data planes paradigm. Software-Defined Networking (SDN) is a promising technology that provides a centralized model with pure software for remote control and dynamic configuration of all heterogeneous network resources and services.

For designing efficient and reliable management of smart-sustainable environments, which is the aim of this thesis, we study the hypothesis of developing generic SDN-based engines for monitoring and optimizing the network underlying infrastructure resources and the massive concurrent heterogeneous flows. To meet this goal, four key issues are required to be addressed in our framework and are summarized as follows: (i) What are the essential particularities of the various smart environments to design generic SDN-based engines? (ii) How to design QoS provisioning aware routing mechanisms? (iii) How to design resource optimization aware routing mechanisms? And (iv) Why/How/Where traffic analysis and characterization (e.g., in terms of QoS) should be performed in fully programmable architectures?

To address the main goal of this thesis and the above research questions, we fix five objectives that enable us to design and experimentally validate new network engines related to traffic analysis, QoS provisioning, resource allocation, and energy consumption. Furthermore, different smart environment use cases (e.g., smart industry and education) have been considered in order to review their requirements and characteristics from various perspectives, thereby validating the proposed engines' commonness. With each environment, we start by reviewing heterogeneous services and the underlying network architecture, where we particularly focus on the softwarization and programmability technologies. Then, based on each network's purpose, we design, over centralized control systems, reactive and proactive mathematical models and algorithms using the Lagrangian relaxation theory to manage a set of thousands of heterogeneous flows while fulfilling the required QoS, optimizing available resources, reducing energy consumption, and minimizing the routing cost. On the other hand, we demonstrate how traffic analysis and service classification aspects play a crucial role in appropriate routing algorithm design and network performance. Hence, we focus on network traffic heterogeneity analysis, particularly IoT and non-IoT traffic flow identification. We propose an extensive evaluation framework based on machine and deep learning mechanisms for fine-grained service differentiation while preserving users' privacy and supporting the evolving nature of network traffic, the high classification accuracy, and the low computational complexity. The results of our study show that the deployment of generic engines over the different smart environments is possible with certain modules (e.g., infrastructure recovery, traffic analysis and characterization, etc.), while some other modules (e.g., QoS provisioning, resource optimization, etc.) need to be adjusted based on the application purpose. Furthermore, the proposed SDN-based engines within each environment significantly enhance QoS assurance, optimize resource allocation, and save energy consumption compared to existing works.

Keywords: smart environments, Internet of Things (IoT), Software Defined Networking (SDN), multi-programmability, concurrent flows, big data, traffic analysis, service characterization, machine and deep learning, distributed rate allocation, Quality of Services (QoS), resource optimization, energy awareness.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS

API	Application Programming Interface
AI	Artificial Intelligence
ACK	Acknowledgment field
BLE	Bluetooth Low Energy
CAGR	Compound Annual Growth Rate
CNN	Convolutional Neural Network
CR	Classified Ratio
CSP	Constrained Shortest Path
CPU	central Processing Unit
DL/ML	Deep/Machine Learning
DCLC	Delay-Constrained Least-Cost
DPI	Deep Packet Inspection
DSCP	Differentiated Services Code Point
DNN	Deep Neural Network
eMBB	enhanced Mobile Broadband
FIN	Finish - No more data from sender
FTP	File Transfer Protocol
FP	False Positive
FN	False Negative

Gbps	Gbps Giga Bit Per Second
GRU	Gated Recurrent Unit
GUI	Graphical User Interface
5G	Fifth generation (Cellular networks)
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
IDC	International Data Corporation
IoT	Internet of Things
IIoT	Industrial Internet of Things
IoE	Internet of Energy
IP	Internet Protocol
IMAP	Internet Message Access Protocol
IDS	Intrusion Detection System
JuMP	Julia for Mathematical Programming
k-NN	k-Nearest Neighbor
LARAC	Lagrange Relaxation based Aggregated Cost
LAN	Local Area Network
LDD	Lagrangian Dual Decomposition
LLDP	Link Layer Discovery Protocol
LoRaWAN	Long Range Wide Area Network

LPWAN	Low-Power Wide-Area Networks
LSTM	Long Short-Term Memory
LTE	Long Term Evolution
MSEs	Medium-Sized Enterprises
M2M	Machine-to-Machine
mMTC	massive Machine Type Communications
NB	Naive Bayes
NFC	Near-Field Communication
NFV	Network Function Virtualization
POP3	Post Office Protocol-3
P2P	Peer-to-Peer
PSH	Push function
QoS	Quality of Service
RFID	Radio-Frequency Identification
ROSA	Route Optimization and Service Assurance
RNN	Recurrent Neural Network
RF	Random Forest
RQ	Research Question
RST	Reset the connection field
RBF	Radial Basis Function kernel

ReLU	Rectified Linear Unit
SMTP	Simple Mail Transfer Protocol
SDN	Software-Defined Networking
SDR	Software-Defined Radio
SPD	Shortest Path Delay
SRAFM	Service and Resource Aware Flow Management
SVM	Support Vector Machine
SYN	Synchronize sequence numbers
TCP	Transmission Control Protocol
TP	True Positive
TN	True Negative
UDP	User Datagram Protocol
UPC	Universitat Politécnica de Catalunya
URG	Urgent pointer field
URLLC	Ultra Reliable Low Latency Communications
Wi-Fi	Wireless Fidelity
WAN	Worldwide Area Network
WLANs	Wireless Local Area networks
WPANs	Wireless Personal Area Networks
WHART	Wireless Highway Addressable Remote Transducer
WSN	Wireless Sensor Network

INTRODUCTION

0.1 Thesis scope

Recently, organizations worldwide have been embracing massive usage of digital technology, such as 5G, cloud/fog/edge computing, data analytics, and artificial intelligence, to boost the quality of living within smart-sustainable environments (e.g., city, industry, education, and healthcare). The Internet of Things (IoT) paradigm refers to the tens of billions of heterogeneous devices autonomously communicating with each other and with remote servers on the Internet while embracing a vast number of diverse applications. Moreover, within the same environment, a massive deal of data is continuously generated from various non-IoT devices, such as smartphones, PCs, cameras, social networking platforms, commercial transactions, and online games. Though these increasingly open, connected, and intelligent environments offer enormous potential in terms of convenience and business value in various application scenarios, they present the largest sources for generating big heterogeneous data, leading thereby to unprecedented challenges for network administrators who are not fully aware of the network content and imperatively need the development of advanced engines to ensure efficient and reliable management of the underlying infrastructure and the corresponding smart applications.

Indeed, the network performance depends mainly on the effective management of the massive heterogeneous data and resources. Nonetheless, intelligent environment services are characterized by various stringent QoS requirements (e.g., transmission rate, packet loss, and latency) whose simultaneous fulfillment represents one of the major challenges facing routing mechanisms design and development. Furthermore, the optimization of the constrained network resources (e.g., limited link capacity) introduces many problems in software development and network protocol structure, particularly if the resource energy consumption is also considered within the engineering engine design.

This thesis begins by surveying different smart environments in terms of IoT and non-IoT-based services, enabling technologies, protocols, and challenges while highlighting potentials for convenience and business value in various fields. On the other hand, we focus on softwarization and programmability standards in order to bring more automation, flexibility, and dynamism into network infrastructure monitoring and control. Thus, we study the hypothesis of designing generic engines for efficient and reliable management of massive smart environments using the Software-Defined Networking (SDN) paradigm. Hence, we address numerous issues related to network architecture design, big data processing, traffic analysis, service characterization, quality of service (QoS) provisioning, resource optimization, and energy consumption awareness. In this context, we design and experimentally validate new SDN-based engines that optimally manage massive intelligent environments. The main contributions of this thesis are summarized as follows.

0.2 Thesis contributions

Previous researchers have focused on a specific type of smart environment (e.g., smart home). Meanwhile, this thesis aims to study various environments to conclude the commonalities and differences to be considered in the design of network management engines and thereby avoid unnecessary duplication. Our research focuses on different networking management aspects within each environment, including the design of fully programmable architecture, traffic analysis, service characterization, QoS provisioning, resource optimization, and energy awareness. It is noteworthy to mention that, to tackle this thesis, we have faced considerable challenges due to different reasons: firstly, the persistent emergence of the IoT-based applications, technologies, and related protocols; secondly, the unavailability of analytical solutions and resources related to specific IoT applications, such as industrial IoT services and digital learning data; thirdly, the non-extensive existing models that may be imperfect to encompass various related aspects, such as heterogeneous data to be characterized, heterogeneous QoS to be assured, heterogeneous infrastructures to be optimized, in addition to the necessity for addressing network sustainability and energy consumption reduction. The major contributions of this thesis are summarized as follows.

Initially, within each smart environment, we focus on the design of the unified fully-programmable architecture using the software-defined networking paradigm to ensure remote monitoring across heterogeneous infrastructures (e.g., wired and wireless networks). Many technologies have been used, such as the SDN and SDR paradigms and the controller orchestrator. Furthermore, we consider the essentials of the different network engines to ensure optimized decision-making. For example, while devising data analytics models, end-user privacy as well as non-intelligent forwarding devices lead us to design a distributed end-host-based network-level for service identification and characterization. This distributed network-level paradigm enables not only to preserve end-user privacy while analyzing network flows but also to optimize the control plane efficiency, the processing time, and the network bandwidth consumption.

Another main contribution is related to the routing awareness functionalities. Specific contributions will be made to the proposition of new strategies that implement QoS provisioning, resource allocation optimization, and energy consumption reduction. Thus, first of all, we design a *parallel routing approach* that is contrary to the existing approaches, takes advantage of the SDN multi-programmability feature, and deploy various algorithms to ensure QoS requirements for a set of hundreds, even thousands of concurrent heterogeneous network services. Furthermore, we propose a *decomposed routing approach* using the Lagrangian method, which finds an optimal solution not only for each flow independently but for the whole set of flows simultaneously while performing coordination between their various QoS requirements and the available resources. With these proposed approaches, we support the introduction of good trade-offs between network service assurance, resource optimization, and energy consumption reduction in order to enhance the overall network performance. Thus, based on the type of traffic, these mechanisms distribute rates across activated resources having low bandwidth utilization to optimize network resources, prevent congestion, and reduce energy consumption.

Furthermore, we focus on network data analytic functionalities to provide a deep understanding of the traffic generated from IoT and non-IoT devices. These engines are crucial for all network management operations, e.g., resource allocation, service assurance, and intrusion detection. Therefore, we propose a comprehensive up-to-date technical survey that facilitates the selection and further development of lightweight fine-grained service differentiation engines. First, we provide a review about network flow features, while highlighting important features and removing irrelevant ones according to the requirements of the classification problem. Then, we deploy the most significant modules in the state-of-the-art related to traffic analysis and service differentiation, including machine and deep learning classifiers, while conducting rigorous evaluations between the different engines. Then, we recommend the deployment of a multi-stage engine for service differentiation composed of CNN and random forest models over a distributed end-host-based architecture to ensure automatic feature engineering process with the evolving nature of network traffic and high classification accuracy.

These major contributions are detailed in the following scientific papers:

- Yosra Njah and Mohamed Cheriet. "Parallel route optimization and service assurance in energy-efficient software-defined industrial IoT networks." IEEE Access 9 (2021): 24682-24696.
- Yosra Njah, Chuan Pham, and Mohamed Cheriet. "Service and resource aware flow management scheme for an SDN-based smart digital campus environment." IEEE Access 8 (2020): 119635-119653.
- Yosra Njah, Misha Cattaneo, Jean Hennebert, and Mohamed Cheriet. "Machine learning and deep learning-based evaluation framework for IoT and non-IoT smart network traffic analytics." submitted to IEEE Internet of Things Journal.

0.3 Thesis organization

The rest of this thesis is organized as follows. Chapter 1 provides the research problem. It presents the general context of the thesis and claims the problem statement, including the specific research questions.

Chapter 2 discusses the relevant literature related to the deployment of SDN-based smart environments. Then, we review various functionalities subject to this thesis (e.g., QoS provisioning, resource optimization, and service identification), followed by a brief discussion emphasizing the gaps in the state-of-the-art and the originality of our research.

Chapter 3 resumes the general methodology of our work and gives a brief overview of the deployed techniques. This chapter also defines our objectives more precisely and explains our motivation behind developing each methodology.

Since this document is a papers-based thesis, Chapters 4, 5, and 6 present the main results of the work in the form of published papers. For the sake of fidelity of the published and submitted versions, these articles are presented without modifications. Although each of the articles addresses different aspects, yet they are all closely interrelated. While a separate section is devoted to the literature review of this thesis, in each article, a specialized state-of-the-art review is presented and analyzed based on the research problems and contributions subject to that particular work.

Hence, Chapters 4 and 5 propose new solutions for managing industrial and campus IoT-based networks, respectively. The philosophy of the proposed methodologies is to develop reactive and proactive engines, including service assurance, resource optimization, and energy consumption reduction based on each network's purpose in order to carry out efficient and reliable smart sustainable environments.

Chapter 6 presents an extensive evaluation framework based on machine and deep learning mechanisms, serving as a map for designing lightweight fine-grained service differentiation engines without accessing packets' contents to preserve users' privacy.

Finally, we conclude the thesis, where firstly, we summarize the work accomplished, while providing discussion on the strengths and limitations of the proposed methods. Then, we provide our recommendations and perspectives.

Figure 0.1 depicts the roadmap of the thesis, the relationship between the motivation, research questions, objectives, proposed approaches, and chapters, to facilitate the reading of this thesis.

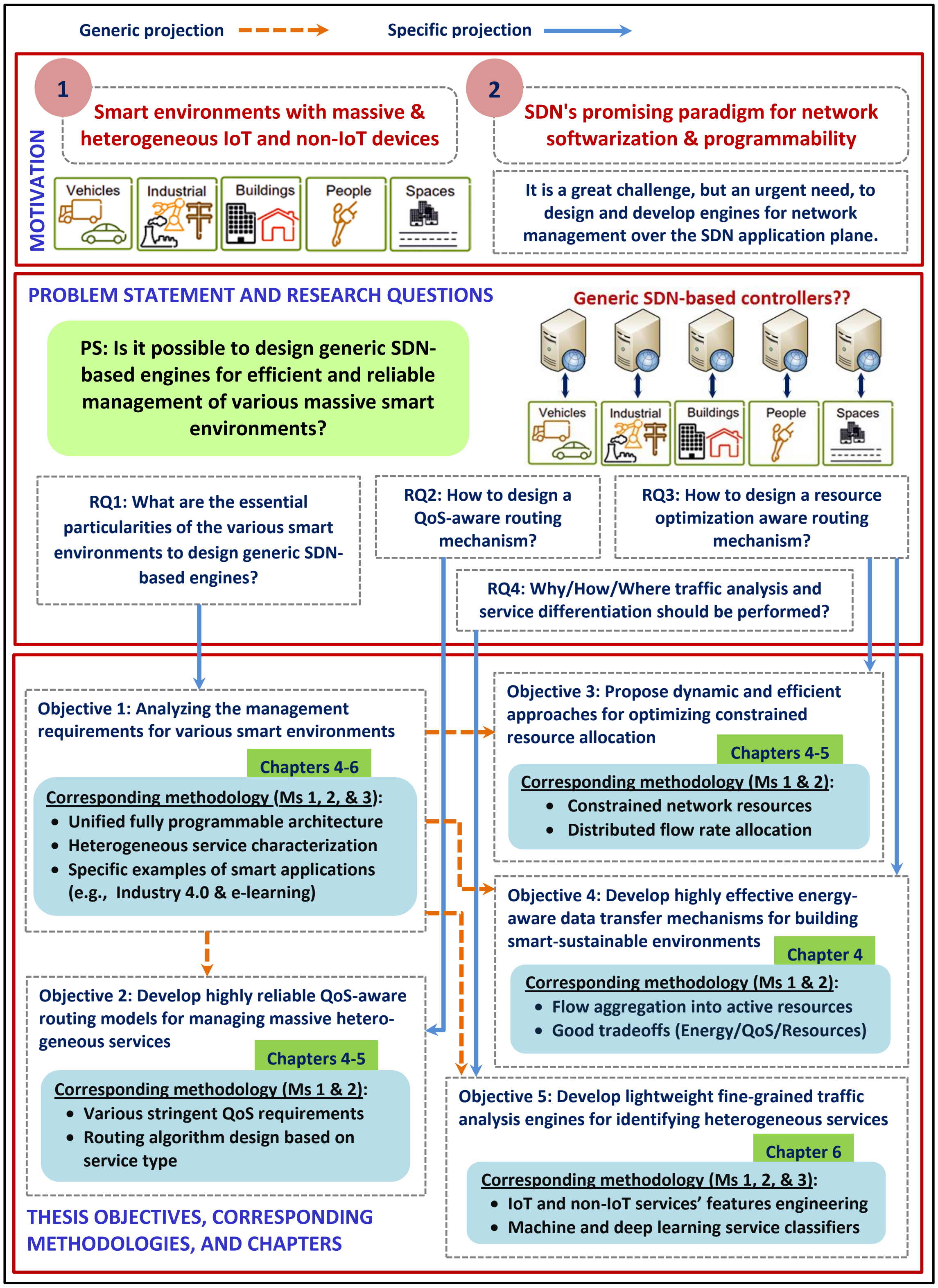

Generic projection
Specific projection
MOTIVATION
1
Smart environments with massive & heterogeneous IoT and non-IoT devices
Vehicles
Industrial
Buildings
People
Spaces
2
SDN's promising paradigm for network softwarization & programmability
It is a great challenge, but an urgent need, to design and develop engines for network management over the SDN application plane.
PROBLEM STATEMENT AND RESEARCH QUESTIONS
PS: Is it possible to design generic SDN-based engines for efficient and reliable management of various massive smart environments?
Generic SDN-based controllers??
RQ1: What are the essential particularities of the various smart environments to design generic SDN-based engines?
RQ2: How to design a QoS-aware routing mechanism?
RQ3: How to design a resource optimization aware routing mechanism?
RQ4: Why/How/Where traffic analysis and service differentiation should be performed?
Objective 1: Analyzing the management requirements for various smart environments
Chapters 4-6
Corresponding methodology (Ms 1, 2, & 3):
Unified fully programmable architecture
Heterogeneous service characterization
Specific examples of smart applications (e.g., Industry 4.0 & e-learning)
Objective 2: Develop highly reliable QoS-aware routing models for managing massive heterogeneous services
Chapters 4-5
Corresponding methodology (Ms 1 & 2):
Various stringent QoS requirements
Routing algorithm design based on service type
Objective 3: Propose dynamic and efficient approaches for optimizing constrained resource allocation
Chapters 4-5
Corresponding methodology (Ms 1 & 2):
Constrained network resources
Distributed flow rate allocation
Objective 4: Develop highly effective energy-aware data transfer mechanisms for building smart-sustainable environments
Chapter 4
Corresponding methodology (Ms 1 & 2):
Flow aggregation into active resources
Good tradeoffs (Energy/QoS/Resources)
Objective 5: Develop lightweight fine-grained traffic analysis engines for identifying heterogeneous services
Chapter 6
Corresponding methodology (Ms 1, 2, & 3):
IoT and non-IoT services' features engineering
Machine and deep learning service classifiers
THESIS OBJECTIVES, CORRESPONDING METHODOLOGIES, AND CHAPTERS

Figure 0.1 The roadmap of the thesis

CHAPTER 1

RESEARCH PROBLEM

In this chapter, we briefly introduce some of smart networks with the corresponding IoT application areas. Furthermore, we point out the main enabling technologies and deployment challenges. Then, we claim the problem statement and the related specific research questions.

1.1 General context and definitions

The Internet of Things (IoT), which is also interchangeably referred to as Machine-to-Machine (M2M) connections, presents an emerging technology that enables an enormous number of heterogeneous objects to be interconnected with each other without human intervention via the Internet. Many organizations across the globe are interested in developing IoT standards to bring tremendous opportunities and benefits, including convenience, flexibility, intelligence, and automation to business and individual life in different domains, such as healthcare, education, government, transportation, agriculture, etc. Figure 1.1 depicts examples of IoT-based intelligent environments. Within each smart environment, a large number of intelligent devices (e.g., sensors), a set of enabling technologies, and controlling engines (e.g., real-time remote monitoring, data transmission, on-demand services, and cyber-security strategies) are used to track and recognize changes in the environment.

1.1.1 Smart environments and IoT-based application domains

In this section, we introduce a set of smart environments with different criteria in terms of network size, architecture, protocols, and data transfer (Ahmed *et al.*, 2016; Mocnej *et al.*, 2018; Al-Fuqaha *et al.*, 2015; Da Xu *et al.*, 2014).

Smart city: The smart city network manages multiple public sustainable services that target a broad spectrum of purposes, improving and optimizing the utilization of public resources (i.e., the core infrastructure components of a city), such as decreasing the everyday running costs (e.g.,

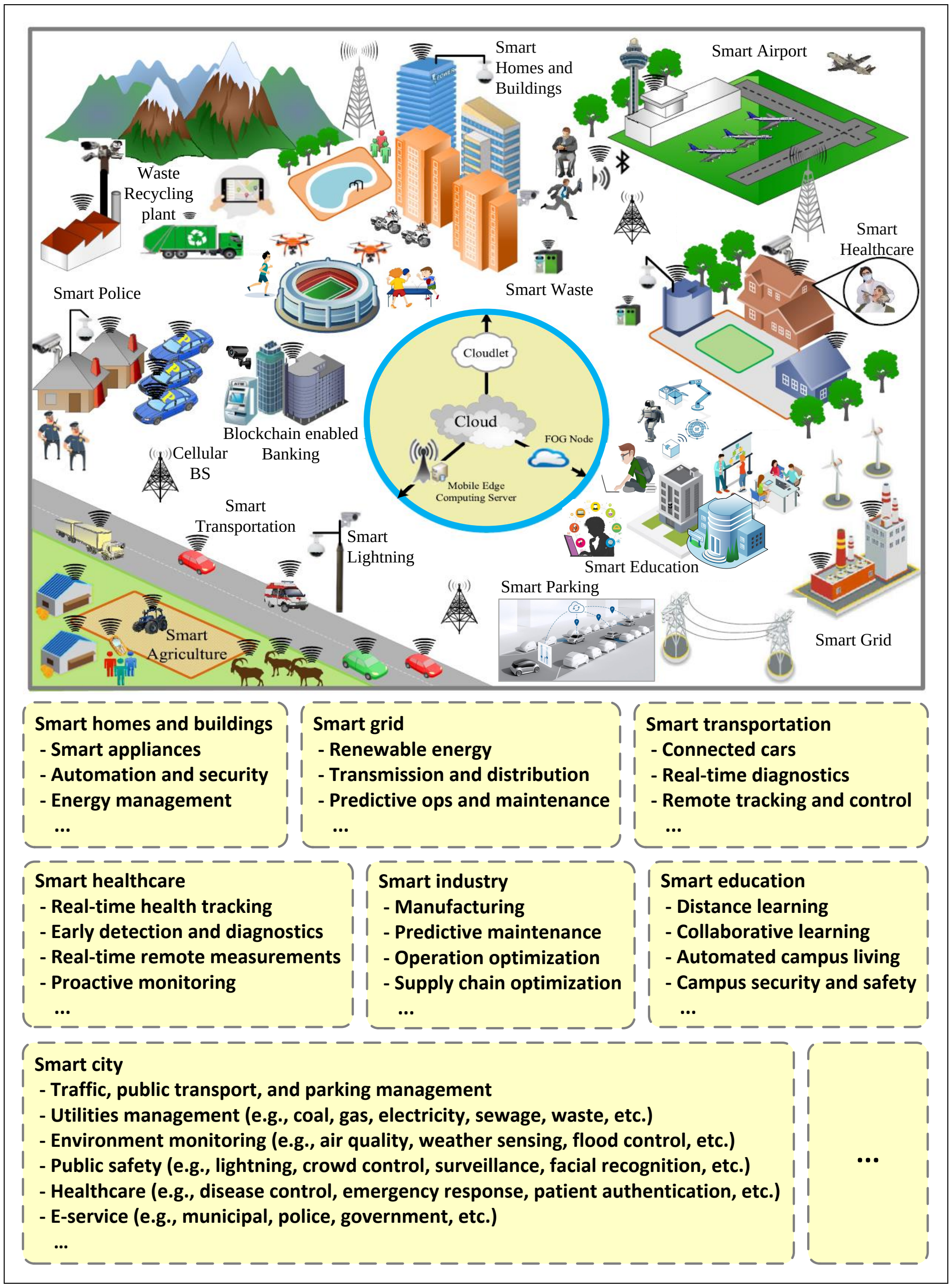

Smart Homes and Buildings
Smart Airport
Waste Recycling plant
Smart Healthcare
Smart Waste
Smart Police
Cloudlet
Cloud
FOG Node
Mobile Edge Computing Server
Blockchain enabled Banking
Cellular BS
Smart Transportation
Smart Lightning
Smart Education
Smart Parking
Smart Agriculture
Smart Grid
Smart homes and buildings
- Smart appliances
- Automation and security
- Energy management
...
Smart grid
- Renewable energy
- Transmission and distribution
- Predictive ops and maintenance
...
Smart transportation
- Connected cars
- Real-time diagnostics
- Remote tracking and control
...
Smart healthcare
- Real-time health tracking
- Early detection and diagnostics
- Real-time remote measurements
- Proactive monitoring
...
Smart industry
- Manufacturing
- Predictive maintenance
- Operation optimization
- Supply chain optimization
...
Smart education
- Distance learning
- Collaborative learning
- Automated campus living
- Campus security and safety
...
Smart city
- Traffic, public transport, and parking management
- Utilities management (e.g., coal, gas, electricity, sewage, waste, etc.)
- Environment monitoring (e.g., air quality, weather sensing, flood control, etc.)
- Public safety (e.g., lightning, crowd control, surveillance, facial recognition, etc.)
- Healthcare (e.g., disease control, emergency response, patient authentication, etc.)
- E-service (e.g., municipal, police, government, etc.)
...
...

Figure 1.1 IoT-based application domains and relevant major scenarios

smart lighting and waste management), improving the available interconnected information (e.g., traffic congestion and parking situation), and boosting the environmental performance (e.g., heavy infrastructure, air pollution monitoring, and flood control). Hence, with the increasing density of the urban population around the world, the intelligent city network size is typically medium up to large by connecting up to thousands of heterogeneous devices.

Smart home and building: The smart home and building-related networks typically incorporate living automation with efficient energy consumption through accurate occupancy detection and different intelligent services, such as lighting adjustment, heating adaptation, water-usage conservation, and appliances control (e.g., battery chargers, refrigerators, and ovens). Furthermore, for increased security insurance, smart home and building environments monitor incidents and anomaly detection through appropriate service-based access control and protection functionalities (e.g., authentication and authorization mechanisms). Hence, the corresponding network size is typically small to medium, connecting tens up to hundreds of heterogeneous devices.

Smart transport, mobility, and logistics: This kind of intelligent network requires that every transport element (e.g., advanced cars, trains, buses, and bicycles) as well as roads and rails to be equipped with sensors and actuators in order to collect and analyze network data related to transportation information (e.g., passenger counting, geo services, ticketing, and communication). These data present a key-enabling technology for deploying various intelligent functionalities, such as vehicle localization and tracking, quality of shipment monitoring, dynamic traffic lights control, and traffic road state monitoring, etc.; consequently improving the current traffic situation in cities (e.g., smoothing transportation, reducing collisions and accidents, reducing energy consumption, etc.). Hence, the corresponding fully smart network typically manages thousands of heterogeneous connected devices.

Smart healthcare: This IoT-based healthcare network enables to improve the quality of medical care while reducing costs through different applications. All healthcare facilities (e.g., medical equipment, patients, staff, assets, and buildings) are instrumented with sensors and actuators to collect information about healthcare environments. This paradigm provides many benefits to the healthcare domain, mainly the remote monitoring of the patients' health (e.g., heart rate, diabetes, and vital body signs) without the patients' physical presence. It reduces patients' safety incidents (e.g., wrong dose/drug/time/procedure), prevents misidentification errors for newborns, and meets security requirements (e.g., identity and access management, real-time materials inventory tracking, etc.). Thus, depending on the tracking application scenario (e.g., staff and patients tracking in a hospital), the network size is typically small to medium, managing the interaction between hundreds of healthcare nodes.

Smart grid: The IoT-based smart grid network controls the power systems in real-time while making them more secure, reliable, efficient, flexible, and sustainable to enhance the generation, transmission, and distribution of electrical energy. This network category is also referred to as the Internet of Energy (IoE), where all power grid equipment are instrumented with different information and communication technologies (e.g., sensors, actuators, and smart meters). These appliances monitor and collect the environment's electrical information (e.g., security system, temperature monitoring, fire detection, and lighting control). Then, they generate power utilization predicting patterns through intelligent algorithms for more accurate power grid infrastructure reconfigurations, efficient power usage, and low overall cost. The smart grid network size is medium to large, particularly within big cities that create a significant demand for efficient network power utilization.

In the rest of this thesis, we will consider in details two different IoT-based application domains: smart industry and smart e-learning campus.

1.1.2 Big data generated within smart environments

Smart environments present the largest sources of collecting large amounts of data (i.e., big data). In fact, as a result of the plethora of IoT devices among each field, which interact autonomously with each other and with remote servers on the Internet (e.g., fog/edge and cloud data-centers), as well as the various general Internet applications developments generated by non-IoT devices, such as smartphones, PCs, cameras, social networking platforms, commercial transactions, online video games, etc., the network traffic data is exploding at an unprecedented rate, leading to big data production (Talebkhah *et al.*, 2021). According to Cisco, the fastest growing mobile device category is the IoT (i.e., M2M connections), followed by smartphones. The M2M is projected to grow at a 30% compound annual growth rate (CAGR) from 2018 to 2023 (i.e., from 6.1 billion in 2018 to 14.7 billion by 2023). Smartphones will grow at a 7% CAGR within the same period (Cisco, Accessed: September 12, 2021). The International Data Corporation (IDC) estimates that 6 billion users, or 75% of the world's population, will be interacting with online data every day by 2025. Another study produced by vXchnge company states that there will be 41 billion IoT devices by 2027, while 70% of automobiles will be connected to the Internet by 2023 (vXchnge, Accessed: September 12, 2021). More precisely, connected home applications (e.g., home automation, home security and video surveillance, connected white goods, and tracking applications) will represent 48%, or nearly half, of the total M2M connections by 2023 (Cisco, Accessed: September 12, 2021).

While the dynamic smart environments, including tens of billions of heterogeneous devices, offer enormous potential in terms of convenience and business value in various fields (e.g., healthcare, manufacturing, education, transportation, and financial services), however, the increasing volume, variety, and velocity of the produced data raise unprecedented challenges for network administrators, since the performance of smart environment's services depends on the data management functionalities.

1.1.3 Enabling network technologies and protocols

Different enabling technologies, including communication (e.g., sensing, 5G, LTE, IP, wired and wireless networks, etc.), computing, softwarization, and data analytic present key factors for enabling the IoT promising paradigm (Macedo *et al.*, 2015; Mocnej *et al.*, 2018; Foubert & Mitton, 2020). In this section, we define some examples as follows:

1.1.3.1 5G and communication technologies

5G technology: 5G is the fifth generation of wireless communications and the latest innovation in mobile network technologies. It is designed for the Internet of Things (IoT) deployment. It is consolidated with all existing technologies while allowing billions of new heterogeneous devices to be connected to the Internet and to each other. The 5G technology mainly addresses the 4G insufficient capacity problems by ensuring proficient connectivity, higher performance, and improved efficiency through massive network capacity with increased availability, more reliability, and ultra-low latency. The 5G technology provides three major new communication scenarios: the enhanced mobile broadband (eMBB) communications that involve capacity enhancements for high data rate demands, the massive machine-type communications (mMTC) that enable interconnection and orchestration between heterogeneous devices while transmitting tremendous amounts of small traffic flows generated by all things, and the ultra-reliable low-latency communications (URLLC) that are characterized by stringent requirements on latency and reliability.

Wireless sensor network (WSN): One of the basic IoT building blocks is a Wireless Sensor Network (WSN) that incorporates a vast number of low-cost, low-power, and multifunctional sensor nodes in a dynamically changing environment. These intelligent devices communicate through wireless mediums under stringent constraints (e.g., energy and computational resources) while interacting with the corresponding communication infrastructure to carry out end-to-end networking operations. Hence, they maintain continuous network connectivity while tracking and recognizing real-time changes, acquiring data from various locations, and transmitting it to the corresponding infrastructure through communication technologies.

Short-range wireless communication technologies: They connect devices in small areas, in the ranges of a few centimeters up to several meters. They include networks, such as contactless data transmission, wireless personal area networks (WPANs), and wireless local area networks (WLANs). Following are examples of low-power, low-bandwidth, short-range wireless technologies applicable in the IoT-based networks: RFID, NFC, BLE, Ant, EnOcean, Z-Wave, Insteon, ZigBee, MiWi, DigiMesh, WirelessHART, Thread, 6LowPAN, and Wi-Fi HaLow.

Long-range wireless communication technologies: They cover large areas up to tens of kilometers. The related technologies focus not only on the long-range coverage but also the energy efficiency, thereby forming Low-Power Wide-Area Networks (LPWAN). Many LPWAN technologies are available and categorized as licensed (e.g., NB-IoT, and EC-GSM-IoT) or unlicensed (e.g., LoRaWAN, Symphony Link, Weightless, SIGFOX, and D7AP) networks based on the used frequency bands to carry communications.

Wireline communication technologies: They include all communication systems for which data is sent through a wire-based technology. They comprise various types of connections, such as twisted pairs, coaxial cables, and fiber optical cables, that are highly resistant to outside interference and noise. Furthermore, many networks today rely on fiber optic communication technology, wherein the supported data rates are incredibly fast. On the other hand, while wireless technologies have higher latency and lower reliability, they are cheaper and more flexible than wired technologies, particularly when there is no need to connect wires to all the points that need coverage.

1.1.3.2 Network cloud, fog, and edge computing

Cloud computing technologies deploy data processing and storage services through data centers accessed over the Internet. However, the centralized cloud paradigm is not appropriate for IoT applications that require real-time and reliability. Therefore, new technologies called fog and edge computing have been emerged to act as bridges between IoT devices and large-scale cloud computing. They are both extensions of cloud networks implemented as congregations of servers in a distributed infrastructure to bring intelligence and processing closer to where the data originated from. Edge and fog computing infrastructures seem similar; while, the key difference between them lies in where intelligence is placed in the network. Edge-computing pushes computational functions close to the data's source using, for example, embedded automation controllers in the users' devices. This paradigm enables real-time data processing, improves security, and reduces network bottleneck since data are not sent across the public Internet. However, by considering the limited processing capacity at the edge infrastructure, fog computing arises to mix between edge and cloud solutions while placing intelligence into the local area network (LAN) over equipment such as routers, switches, IoT gateways, and sensors. This paradigm enables low latency with flexible decentralized storage systems compared to the centralized cloud approach (Atlam *et al.*, 2018).

Hence, edge, fog, and cloud computing layers differ by their design and purpose but complement each other to address different issues such as security, efficiency, resource allocation, latency, reliability, and user experience optimization.

1.1.3.3 Network softwarization and programmability

Network management operations must meet various concerns (e.g., availability, reliability, flexibility, efficiency, and fault tolerance). Meanwhile, the traditional networks lack global timing information, synchronization, and effective management in network elements. Therefore, the network softwarization has become a strong worldwide interest as one of the key enabling technologies to introduce programmability and flexibility into heterogeneous network infrastructures. Network softwarization is enabled by different standards, such as SDN (Software Defined Networking) and NFV (Network Function Virtualization) (Macedo *et al.*, 2015; Alam *et al.*, 2019).

***Software-Defined Networking*:** SDN is an emerging paradigm that enables administrators to remotely and continuously control the underlying network infrastructures. The control intelligence is separated from the data plane and implemented further away in a logically centralized control plane (i.e., the brain of the network). Thus, it automatically configures engine rules (i.e., network policies and forwarding functions) into the data plane devices (simple high-speed forwarding elements) based on the state of the network and using software programs over the application plane (e.g., discovery, monitoring, and optimization) through standard interfaces (e.g., southbound and northbound APIs). SDN has also reached wireless networks with concepts including Software Defined Radio (SDR) (Macedo *et al.*, 2015).

***Network Function Virtualization*:** NFV runs on virtual machines networking processes (e.g., routing, firewall computing, and load balancing) that traditionally run on physical devices. Consequently, NFV improves agility by allowing service providers to deliver new network functions and applications dynamically on-demand without installing new commodity hardware. Furthermore, contrary to the traditional paradigm where each device is dedicated to a specific network function, with NFV, a single server can run different virtual network functions simultaneously. Thus, scaling the network architecture with virtual machines is faster and easier, and it does not require purchasing additional hardware.

1.2 Problem statement and research questions

Over the different intelligent environments, such as industry, education, healthcare, and transportation, precision and reliability are critical objectives that should be committed. Accordingly, numerous challenges for efficient engineering functionality need to be targeted, such as the computation of reliable routing paths through end-to-end connections between large numbers of stakeholders, in addition to the support for a wide range of heterogeneous services with precise QoS requirements.

The traditional network architecture has limited global visibility of the overall network architecture and the corresponding available resources since each router performs a hop-by-hop routing using its coupled control and data planes (Xia *et al.*, 2015). Furthermore, this network paradigm is complex and time-consuming for resource configuration and traffic flow management; for example, if we need to provide a new engineering mechanism, all the network devices' communication protocols should be updated statically one by one, which leads to low efficiency, scalability, and interoperability. In this regard, pure software methods have been required to enable remote control and dynamic configuration of all heterogeneous network resources and services. SDN has emerged as a centralized management paradigm that continuously and remotely controls heterogeneous network resources, demands, and end-to-end connections by means of programmability, automation, and flexibility, as shown in Figure 1.2 and described in Section 1.1.3.

Nevertheless, despite the promising paradigm of SDN, it is still a great challenge (but an urgent need) to design many engines and functions over its application plane (e.g., data analysis,

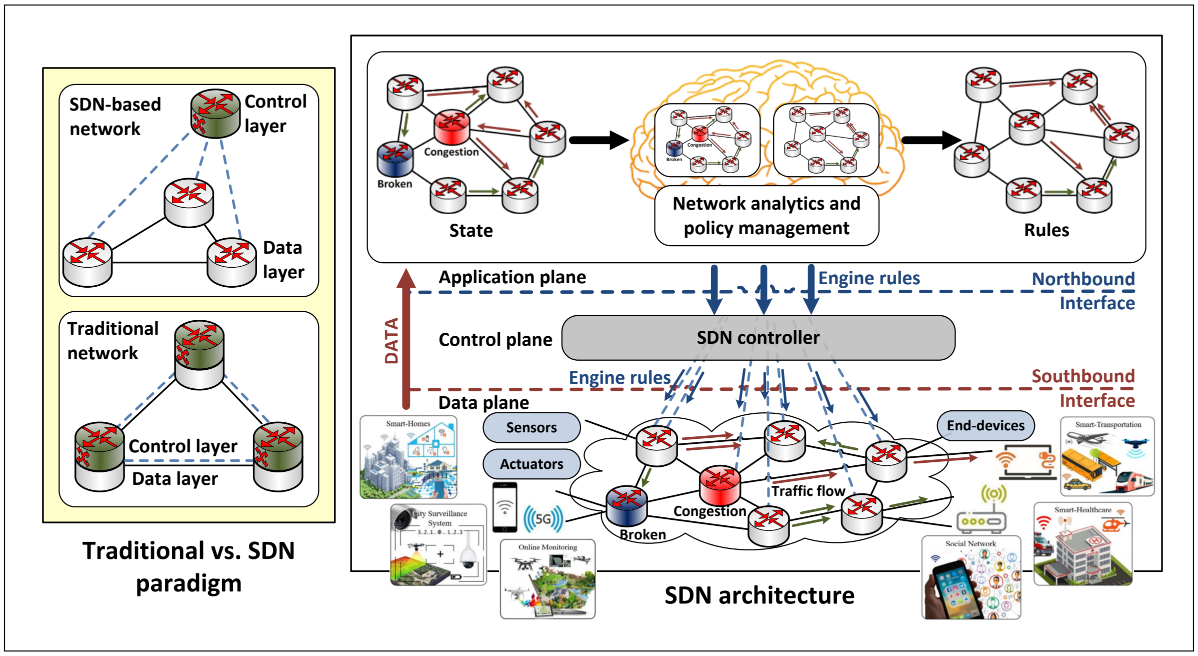


Figure 1.2 The SDN layered architecture

network security, service assurance, resource optimization, and failure prediction) for reliable and efficient management of the different network resources and demands. On the other hand, despite recent advancements in IoT-based intelligent environments, scientific and engineering challenging issues are yet to be solved by ingenious research efforts from academia and industry, particularly regarding the development of efficient and reliable network management solutions.

1.2.1 Problem statement: Generic SDN-based engines for efficient and reliable management of massive smart environments

In response to the above concerns, in this thesis, we address new architectures, protocols, and mechanisms for controlling various IoT-based smart environments. As a technology service provider, we study the hypothesis of designing and implementing generic centralized solutions for monitoring, managing, and optimizing intelligent environments and their corresponding massive concurrent heterogeneous flows. Thus, the research problem addressed in this thesis is stated as follows: *How to ensure end-to-end network efficiency and reliability through various SDN-based* *control systems, such as service characterization, QoS provisioning, traffic engineering, and resource optimization for different smart-sustainable environments?*

1.2.2 Research questions (RQs) and related main challenges

To address the above problem statement and drive our work methodology that will be discussed in Chapter 3, we further detail the problem statement into four research questions (RQs) as follows:

1.2.2.1 RQ1: What are the essential particularities of the various smart environments to design generic SDN-based engines?

While smart environments are different in the application purpose, they are homogeneous in terms of deployed technologies (e.g., wired/wireless communications systems and sensor networks). In addition, they have in common several use cases, such as real-time environment monitoring, diagnostics, and emergency detection, that run concurrently on the same infrastructure with the main application (e.g., manufacturing, healthcare, and education). On the other hand, from the network management side, different functionalities (e.g., infrastructure discovery and link failure detection) could also be considered as mutual between the different IoT-based smart environments. Therefore, it is rational to study the design of generic SDN-based engines for managing intelligent environments while considering the essential particularities of the various use cases. In other words, within each environment, we need to study the fundamentals related to a smart application purpose (e.g., smart industry) in addition to the following research question: What are the network policies that proactively automate and enhance the management process of a set of hundreds, even thousands of heterogeneous flows while fulfilling the required QoS, optimizing available resources, and minimizing the routing cost? We will address this concern with more specific questions in the remainder of this section.

1.2.2.2 RQ2: How to design a QoS-aware routing mechanism?

A wide range of heterogeneous IoT and non-IoT flows generated in smart environments necessitate various stringent QoS requirements (e.g., transmission rate, packet loss, and latency). For instance, certain safety-critical flows generated during some emergency/urgent periods, such as gas, smoke, and disaster monitors, have strict timing requirements with extremely low packet loss to conduct accurate and real-time control decisions. Certain types of mission-critical services, such as e-learning for smart education and streaming for remote medical and healthcare interventions, require reliable communication with ultra-low latency. Some other applications (e.g., peer-to-peer applications and cloud-based file storage systems) are not time-sensitive but very bandwidth-hungry applications that might occupy all available bandwidth and violate safety- and mission-critical flows while sharing the same resources through inefficient routing mechanisms. The fulfillment of services' characteristics (e.g., the maximum end-to-end latency and the packet loss probability requirements) represents one of the major challenges facing routing mechanism design. In other words, designing QoS-aware routing mechanisms that guarantee data delivery, availability, quality, and accuracy for each network service within a set of hundreds, even thousands of concurrent heterogeneous flows, is one of the main challenges in the deployment of efficient and reliable smart environments that will be addressed in the remainder of this thesis.

1.2.2.3 RQ3: How to design a resource optimization aware routing mechanism?

First, while considering the QoS provisioning exigency, the network resource optimization problem is basically complex since it is NP-hard by nature. Furthermore, the design of a routing model for managing massive heterogeneous data stream while optimizing the underlying infrastructure allocation is highly challenging, especially with the constrained resource issues (e.g., limited link capacity, constrained device with limited CPU, memory, and power) that introduce many problems in software development and network protocol structure. Moreover, the routing functionality needs to load balance traffic flows in the network equipment to effectively reduce the routing path end-to-end delays and perform end-to-end congestion control. In fact, even though a network has been designed with highly adequate available resources, but without efficient engineering mechanisms for service management and resource allocation, critical flows may compete with all kinds of traffic, including bandwidth-hungry flows, which causes serious service violations in the network. Accordingly, designing path planning to support large amounts of data and control the constrained network resources with effective resource allocation and minimum-cost strategy is one of the dire necessities for efficient network performance. On the other hand, joining the energy consumption concern to the design of the resource optimization model raises another level of complexity. In fact, a smart environment requires the setup of multiple heterogeneous devices with high density for sensing, collecting, and transferring large amounts of data. Consequently, it consumes a significant volume of energy. Therefore, an energy-efficient data transfer mechanism that includes reducing power emission and pollution is crucial for the deployment of a green sustainable environment.

1.2.2.4 RQ4: Why, How, and Where traffic analysis and service differentiation should be performed over an SDN-based programmable architecture?

To enhance the automation of network management and optimization activities (e.g., traffic engineering, resources allocation, QoS provisioning, and intrusion detection), a deep understanding of the network traffic nature and characteristics is fundamental for operators, since it enables them to be fully aware of the network content. Thus, all massive amounts of exchanged data need to be accessed, analyzed, categorized into traffic classes, and then managed in order to achieve efficient and reliable network operations. In this thesis, we consider traffic analysis as one of the leading engines that need to be studied, devised, and implemented in the SDN-based platform for smart environments. Particularly that the design of service characterization models presents a key-enabling technology for a concise specification of service requirements, and that a joining system of traffic identification and management is mandatory to devise appropriate routing solutions for supporting network optimization and QoS provisioning. Hence, to understand network traffic and design analytics engines, we will focus on the following related issues: (1) What are network services' characteristics in each massive environment? (2) How to perform timely and accurate traffic analysis while considering the evolving nature of heterogeneous network flows? (3) Where should traffic analysis engines be deployed within SDN-based network architectures in order to protect users' privacy and optimize the management process?

CHAPTER 2

LITERATURE REVIEW

This chapter reviews the state-of-the-art approaches relevant to various aspects subject to this thesis. Thus, we start with a review of the literature related to the deployment of smart environments. Then, we review various SDN-based network functionalities subject to this research work (e.g., QoS provisioning, resource optimization, energy awareness, and data analysis). Finally, we conclude the chapter with a brief discussion emphasizing the gaps in the state-of-the-art and the originality of our research. Yet, we leave a deep analysis of the reviewed mechanisms within each technical chapter separately, where we compare our propositions with existing works relevant to each contribution.

2.1 Smart environment deployment over softwarized infrastructures

Significant literature has been devoted to describe smart environment deployment and traffic management over programmable and softwarized infrastructures. For example, Hajjaji *et al.* (2021) reviewed big data and IoT-based applications in smart environments. They identified key areas of applications, current trends, architectures, and challenges while highlighting how the integration of big data and IoT technologies creates exciting opportunities for monitoring, protecting, and improving real-world natural resources. Cai *et al.* (2016) provided a roadmap for smart environments deployment while focusing on the foundations and principles of big data platforms, architectures, and storage systems. Talebkhah *et al.* (2021) reviewed the main characteristics of IoT-based smart cities in terms of architecture, challenges, and opportunities for big data management systems in large-scale intelligent cities networks. Furthermore, a hybrid architecture to optimize the storage of big data coming from smart environment monitoring activities into the cloud is presented in (Fazio *et al.*, 2015). Finally, Alberti *et al.* (2019) surveyed how SDN and IoT platforms and frameworks open the doors for more flexibility, automation, and scalability in the deployment of network functionalities.

All the above mentioned surveys review the life-cycle processing of IoT applications and big data from the IoT sensors to the cloud platform. In the rest of this section, we will focus on the fundamental role of SDN to manage IoT application and big data, while taking into consideration specific network functionalities, such as QoS provisioning, resource optimization, and service identification to build various smart environments.

2.2 SDN-based network functionalities

2.2.1 Service assurance and resource optimization-aware routing engines

The literature on service assurance and resource optimization-aware routing problems is vast. Thus, in this section, we consider relevant works over software-defined generic networks, IoT, Internet of (Campus) Things, grid, healthcare, and industrial networks, which are tackled in this thesis since they are attracting increasing attention from both academia and industry.

Egilmez *et al.* (2013) presented a generic SDN-based analytical framework to optimize the QoS-aware routing for the video streaming service. They assigned the highest routing priority to the video streaming service, while the rest of flows were managed with best-effort delivery. Then, they transformed the QoS-aware routing into a Constrained Shortest Path (CSP) problem solved using the LARAC algorithm (Juttner *et al.*, 2001). Despite the fact that Egilmez *et al.* (2013) covered several critical issues in terms of multi-level QoS, however they did not consider heterogeneous types of network services.

Saha *et al.* (2018) proposed a parallel QoS routing scheme running two separate single-metric models over SD-IoT networks. The first model was devised for delay-sensitive traffic, and the second one was designed for loss-sensitive traffic while using the Yen's-K shortest path algorithm (Yen, Accessed: October 20, 2020) as the basis for both of them. However, considering the delay and the loss sensitivities as the two main characteristics for classifying all SD-IoT network traffic is inefficient, since, for example, ultra-reliable low-latency communications (URLLC) are characterized equally with these two parameters (i.e., delay and loss). Thus, to which class would such services be assigned, i.e., according to (Saha *et al.*, 2018), should URLLC services be assigned to the loss-sensitive class or the delay-sensitive class?

Rezaee & Moghaddam (2019) presented a wide area measurement system using the SDN paradigm for service assurance and resource optimization in the intelligent grid network. They considered two different classes of traffic, where available bandwidth resources were assigned to high-priority packets that contained new data, whereas less-important packets that contain older data were dropped using a queue management approach. The LARAC algorithm (Juttner *et al.*, 2001) was deployed to solve the DCLC problem (applied to the bandwidth) and find the best path. However, in (Rezaee & Moghaddam, 2019), besides the single-metric proposed approach, which does not fit a large number of heterogeneous services in smart networks, another disadvantage in this work resides in dropping packets containing older data, which significantly affects the network services' reliability.

Professor Wolfgang Kellerer with his research team at the Technical University of Munich, who are the authors of (Guck *et al.*, 2016) and (Zoppi *et al.*, 2018), are pioneering in the field of QoS-aware routing for SDN-based industrial networks. Guck *et al.* (2016) have proposed a centralized QoS control framework in SDN-based industrial networks. The proposed solution manages QoS through a function split between DCLC routing and resource allocation using network calculus. Zoppi *et al.* (2018) devised a reliability-aware dynamic scheduler for industrial networks where the LARAC scheme was chosen to perform QoS-aware routing and to guarantee the delay bound of each application. Nevertheless, the proposed frameworks are designed to ensure a strict end-to-end delay parameter, typically required for industrial systems, but not enough to fulfill heterogeneous QoS industrial communications. Furthermore, these works do not address the energy awareness problem, which is a crucial aspect in the design of an IoT-based platform since industrial networks are characterized by high energy consumption.

Naeem *et al.* (2020) proposed an SDN-based energy-efficient and QoS-aware parallel routing scheme for IIoT-based intelligent healthcare networks. The authors considered a max-flow-min-cost optimization problem with multi-constrained QoS parameters while characterizing medical services as jitter-sensitive, loss-sensitive, and delay-sensitive flows. The objective was to maximize flow gathering over the active resources while minimizing the bandwidth costs and fulfilling the QoS requirements. However, Naeem et al., in the SDN-based smart healthcare network, adopted the Yen's-K shortest path algorithm (Yen, Accessed: October 20, 2020) that was taken into consideration in (Guck *et al.*, 2018), where Guck *et al.* (2018) proved the outperformance of the LARAC algorithm (Juttner *et al.*, 2001) as compared to 26 SDN-based routing algorithms.

Long *et al.* (2018) proposed a routing scheme to enhance communication latency performance and energy consumption in IIoT networks. Within each path selection, they aggregated network data through different clusters across a hierarchical framework to reduce energy while estimating the latency by using hop count. However, while Long *et al.* (2018) considered reducing energy as an essential factor, they did not consider heterogeneous application-dependent requirements, such as loss-sensitive, delay-sensitive, or a combination of any QoS traffic types. Considering only hop count is not enough to validate low-latency transmission. Furthermore, aggregating network flows without considering prioritization mechanisms for efficient bandwidth allocation leads to network congestion and service performance degradation. Therefore, the energy and hop count-based proposed model is insufficient for handling heterogeneous IIoT services, especially that the requirements of industrial systems in terms of delay and reliability provisioning are more critical than energy saving.

Finally, regarding smart digital campus networks, it is noteworthy that many research studies have been proposed to design and build intelligent campus networks with the appropriate technologies (e.g., traffic profiling, prediction of student attendance, etc.) to move towards digital education (Sutjarittham *et al.*, 2019; Uskov *et al.*, 2015). However, few research studies deal with QoS provisioning and resource optimization over SDN-based intelligent campus networks. Meanwhile, industrial companies, such as Cisco (Cisco, Accessed: July 12, 2019), Campus Management Corp (Campus-Management, Accessed: July 7, 2019), Ruckus (Ruckus, Accessed: July 16, 2019), and Huawei (Huawei, Accessed: December 19, 2019) have revolutionized how to design, build, and manage smart digital campus networks.

2.2.2 Heterogeneous service analysis and identification approaches

We survey traffic analytics and service differentiation techniques, particularly machine and deep learning models, which play a central role in the management and optimization (e.g., service assurance, resource optimization, and intrusion detection blocking) of softwarized networks.

Various machine-learning-based approaches have been proposed in the literature for traffic classification. Sivanathan *et al.* (2018) proposed a 2-stage machine-learning model using random forest and naive Bayes algorithms for real-time IoT devices classification. They considered different flow features, such as port numbers, domain names, cipher suites, and flow volumes. The proposed model achieved over 99% accuracy in an environment of 28 IoT devices. However, the authors focused mainly on distinguishing between IoT and non-IoT traces without considering the fine-grained service recognition that is the workhorse of many network management and prioritization operations.

Yamansavascilar *et al.* (2017) evaluated four machine-learning methods (J48, random forest, k-NN, and Bayesian network) to identify 14 different non-IoT services. They selected 12 attributes, including the flow duration, the minimum, maximum, and average statistics of the packet, in addition to the window size, from a set of 111 flow-based features. They focused on the techniques' accuracy performance and showed that the k-NN (k = 1) algorithm provided the most accurate result with 93.94%, while the second most accurate algorithm was the random forest technique with 90.87% accuracy. This work presents an important basis; nevertheless, the authors addressed only general non-IoT applications. Furthermore, by considering the flow duration attribute as one of the critical features, they did not address the early-stage processing to ensure real-time classification and reduce resource consumption. Considering only the accuracy metric is not enough to evaluate the performance of different machine learning classification techniques.

Fan & Liu (2017) addressed the general non-IoT application identification using machine learning techniques while focusing mainly on supervised Support Vector Machine (SVM) and unsupervised k-means clustering algorithms. They investigated how model adjustment and feature selection affect the performance of the classification techniques. Experimental results indicated that an overall accuracy of over 95% could be achieved with SVM, even with a small portion of the dataset for training. Although the accuracy of the K-means classifier is highly affected by the size of the dataset, it enables characterizing new or unknown application types. Besides, Sun *et al.* (2018) adopted SVM as the baseline method in traffic classification. While supporting the high accuracy of this technique, they addressed two main limitations, including the high training cost in terms of memory and CPU and the inability to support continuous learning. The experimental results proved the effectiveness of the proposed models in traffic classification.

Moore & Zuev (2005) used the naive Bayes technique for service identification with ten general non-IoT applications, e.g., bulk data transfer, database, mail, services, www, P2P, attack, games, and multimedia. Then, they extended the work with the application of the Bayesian neural network approach in (Auld *et al.*, 2007). They proved the accuracy enhancement compared to the naive Bayes technique and showed the high accuracy potential of the proposed model up to 98%. Furthermore, they used a list of 20 features from a set of 246 full-flow-based statistical features with their descriptions and importance ranking. However, many of these selected features, such as the effective bandwidth based upon entropy and the Fourier transform of packet inter-arrival times, are computationally challenging versus the accuracy detriment.

Lopez-Martin *et al.* (2017) used deep learning to conduct the IoT traffic classification. They explored the recurrent neural network (RNN), the convolutional neural network (CNN), and a combination of CNN and RNN models. They considered six features, including source and destination port numbers, number of bytes in the packet payload, TCP window size, inter-arrival time, and packet direction extracted from the first 20 packets of a flow. The designed combined model demonstrates good performance, attaining an accuracy of 96.32%. However, Lopez-Martin *et al.* (2017) provided non-exhaustive lists of services and corresponding features for IoT traffic.

Table 2.1 Synthesis of related works

Heterogeneous service assurance and resource optimization approaches						
Related work	**Heterogeneous services**	**Service assurance**	**Resource optimization**	**Network**	**Algorithm (per-flow routing)**	**Limits**
Egilmez et al. (2013)	No Only delay sensitive	Yes Video streaming	No	SD-generic network	DCLC with LARAC	1) Gap of heterogeneous services 2) Best effort for all the rest of traffic
Saha et al. (2018)	Yes Delay and loss sensitive	Yes	Yes	SD-IoT	Two separate single metric models-based on Yens algo	1) Yens algorithm (Guck *et al.*, 2018) 2) To which class URLLC are assigned?
Rezaee & Moghaddam (2019)	No Prioritizing new packets	Yes	Yes	SD-Grid	DCLC with LARAC (applied to the required bandwidth)	Drop less critical packets containing older data: degrade services' reliability
Guck et al. (2016)	No Delay sensitive	Yes	Yes	SD-Industrial networks	Network calculus-based approach	1) Delay metric is not enough to fulfill heterogeneous industrial services 2) Not considering energy awareness
Zoppi et al. (2018)	No Delay sensitive	Yes	Yes	SD-Industrial networks	DCLC with LARAC Bandwidth scheduler	Energy awareness is an important metric for industrial networks
Naeem et al. (2020)	Yes Delay/jitter/loss sensitive Energy awareness	Yes	Yes	SD-IIoT Healthcare	Three separate single metric models-based on Yens algo Max flow/Min cost for energy	1) Some services are sensitive to the delay, jitter, and loss simultaneously 2) Yens algorithm (Guck *et al.*, 2018)
Long et al. (2018)	No Energy and latency (using hop count)	No	No Energy saving	SD-IIoT	A cluster-based algorithm for flow aggregation	1) Estimating the latency by hop count 2) Aggregating flows for energy saving without considering service prioritization 3) Delay and reliability are more critical than energy saving

Heterogeneous service analysis and identification approaches					
Related work	**IoT services**	**Non-IoT services**	**ML/DL model**	**Automatic feature engineering**	**Limits**
Sivanathan et al. (2018)	Yes	Yes (But not fine-grained)	2-stage machine-learning model (random forest and naive Bayes)	No	1) No analysis for non-IoT classes 2) Non-automatic feature engineering
Yamansavascilar et al. (2017)	No	Yes	J48, random forest, k-NN, and Bayesian network	No	1) Only general non-IoT services 2) Non-automatic feature engineering 3) Non-convergent deployed approaches 4) Inconsistent features to online analysis 5) Computationally challenging features
Fan & Liu (2017)	No	Yes	SVM and Kmeans	No	
Sun et al. (2018)	No	Yes	SVM	No	
Moore & Zuev (2005)	No	Yes	Naive Bayes	No	
Auld et al. (2007)	No	Yes	Bayesian neural network	No	
Lopez-Martin et al. (2017)	Yes	Yes	CNN/RNN/Combination of CNN and RNN models	Yes	Non-exhaustive lists of services and corresponding features for IoT traffic

2.3 Conclusion and originality of the research

Table 2.1 synthesizes some related works in terms of deployed mechanisms and limits. In conclusion, regarding service assurance and resource optimization, most proposed approaches are designed to ensure a strict end-to-end delay parameter, typically required by time-sensitive services, but not enough to fit a large number of network services and fulfill heterogeneous QoS requirements. While this approach enables a delay-constrained transmission, nevertheless, it does not meet the QoS requirements of multiple heterogeneous services in smart environments. For instance, mission-critical flows generated during emergency/urgent periods, such as gas monitors, smoke sensors, and disaster sensors, require real-time data transference. Some other applications, such as peer-to-peer file sharing, software updates, and cloud-based file storage systems, are not time-sensitive but very bandwidth-hungry applications that might occupy all available bandwidth in the case of inefficient management. Furthermore, certain types of services, such as online learning and video streaming, require bandwidth and real-time guarantees to run without degraded performance. Thus, without efficient engineering mechanisms for controlling all network flows, critical services may compete with all kinds of traffic, including bandwidth-hungry flows that may cause serious service QoS violations in the network. In this thesis, we propose a global approach that addresses not only one type of service (e.g., a delay-constrained least-cost problem), but multiple heterogeneous services characterized by various specific QoS requirements (e.g., delay, loss, and bandwidth).

Moreover, to the best of our knowledge, most of the existing works are based on a per-flow approach, i.e., using the current state of the network, the optimal routing solution is selected independently for each flow and not for all concurrent flows in the system. Though this flow-by-flow technique can reach a fast routing decision, it cannot provide an optimal global solution for a whole set of concurrent flows; and in some cases, it violates service requirements and network performance. Therefore, a solution that simultaneously addresses an entire massive set of heterogeneous flows in the network is of fundamental importance for service assurance and resource optimization. On the other hand, numerous research studies address the QoS provisioning problem in different SDN-based intelligent environments. However, only a few research studies deal with the joining problem between heterogeneous service assurance, resource optimization, and energy awareness aspects, which, to the best of our knowledge, has not yet been sufficiently explored and is still an open issue with many challenges enabling to build efficient and reliable smart-sustainable networks.

Regarding heterogeneous service analysis and identification functionalities, current works focus either on a global differentiation between IoT and non-IoT traffic or on identifying the general non-IoT demands. Meanwhile, it is fundamental to enable fine-grained recognition for all network services since they coexist and share the same network resources in intelligent environments, thereby enabling the deployment of different network control functionalities (service assurance and resource allocation). In addition, all the existing works discussed in the above section, show the non-convergent approaches deployed for addressing the same issue, i.e., traffic analysis and service differentiation, particularly with the deep learning deployment that is still thorny and less approved in network traffic classification. Therefore, a comprehensive up-to-date survey with rigorous evaluations and comparisons of the well-known machine and deep learning-based classifiers with different sets of features and from various aspects (e.g., the classification accuracy and the time and space complexity) is required to facilitate the selection and further contribution to the design of more appropriate, lightweight, and fine-grained service differentiation engines for smart environments.

CHAPTER 3

GENERAL METHODOLOGY

In this chapter, we expose the main purpose and methodology of this thesis in accordance with the abovementioned research problem, the related research questions, and the limitation of prior works. In addition, we further detail the general methodology through five specific objectives that explain the rationale of the different proposed solutions.

3.1 Research objectives

The main objective of this thesis is to study the hypothesis of designing and implementing generic SDN-based solutions for monitoring, managing, and optimizing massive heterogeneous network resources and concurrent flows to build efficient and reliable smart-sustainable environments. It will be achieved with five specific objectives that are summarized as follows:

3.1.1 Objective 1: Analyzing the generic management requirements for various smart environments (e.g., industrial and e-learning applications)

Numerous existing works in the literature, such as (Egilmez *et al.*, 2013), (Guck *et al.*, 2018), and (Okay & Ozdemir, 2018), have deployed generic solutions for network management and optimization using the SDN paradigm. We examine in our thesis the design of SDN-based generic functionalities for various smart environments that are different in the applications' purpose but analogous in deployed technologies. Thereby, in addition to the commonly deployed services (e.g., connected lighting and parking, building automation, smoke alarms, and security monitoring), we examine two specific use cases: the industrial and e-learning intelligent applications. Furthermore, we study the design and deployment of different SDN-based network management engines for each environment, such as QoS provisioning, resource allocation, energy consumption, and service differentiation, which will be addressed in the following objectives, as shown in Figure 3.1. The corresponding proposed solutions are presented in Chapters 4, 5, and 6.

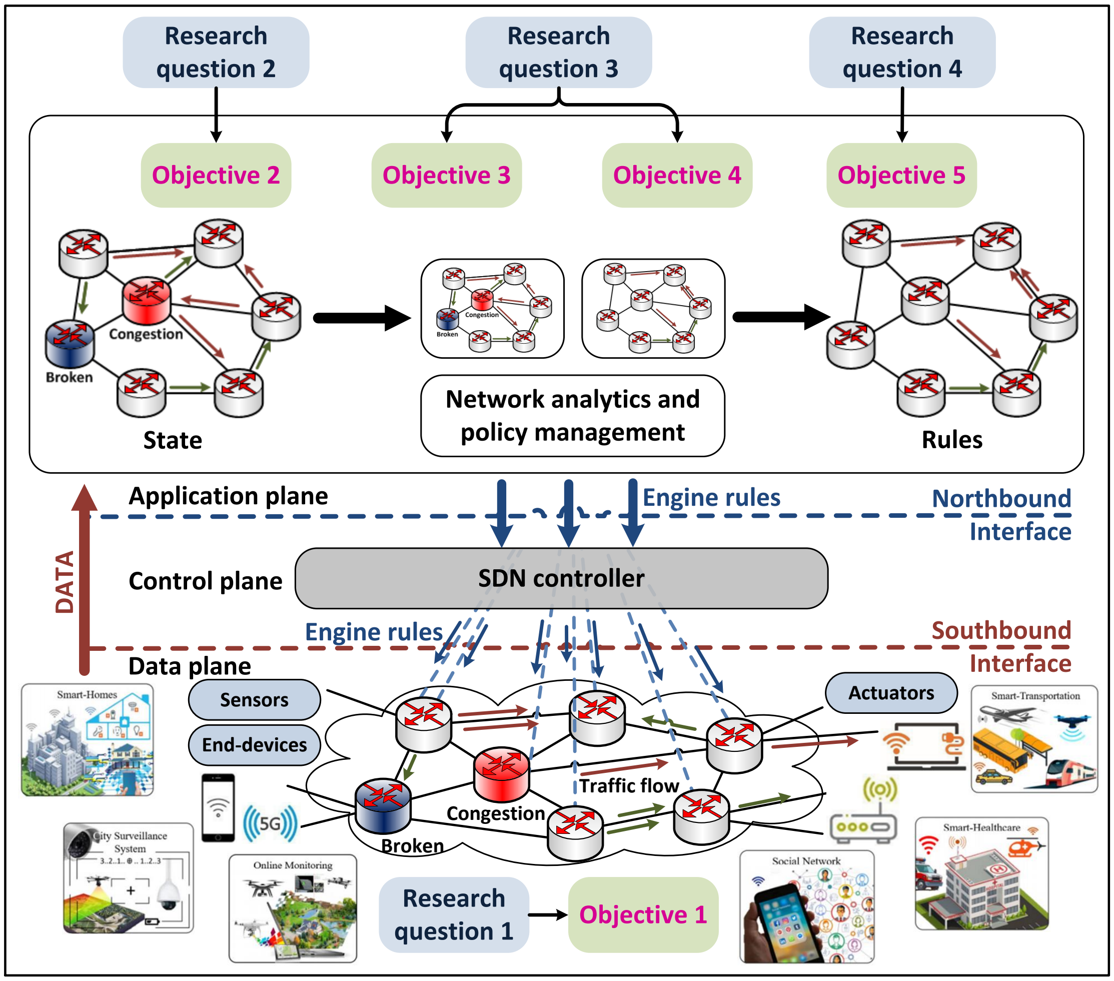


Figure 3.1 Research questions vs. specific objectives

3.1.2 Objective 2: Develop highly reliable QoS-aware routing models for managing massive, concurrent, and heterogeneous services

The current works are designed to ensure a strict end-to-end delay parameter, which is typically required for real time-sensitive services but not enough to fulfill the requirements of a large number of heterogeneous communications in smart environments. In our work, we propose global multi-metric QoS-aware routing approaches. The presented models target not only one type of service (e.g., delay-sensitive demands) but various services, including bandwidth-hungry flows that should be controlled by specific engineering policies; otherwise, they create congestion and degrade the performance of mission-critical applications when the same network resources are shared. We describe the proposed approaches in Chapters 4 and 5.

3.1.3 Objective 3: Propose highly dynamic and efficient approaches for optimizing resource allocation while considering constrained infrastructures and flows heterogeneity

Traditional resource allocation-aware routing approaches are mainly based on a per-flow mechanism; i.e., using the current state of the network, the optimal routing solution is selected independently for each demand. Though this flow-by-flow technique can reach a fast routing decision, it cannot provide an optimal global solution for a whole set of concurrent streams and thereby degrading the overall network performance. We propose centralized optimization approaches that manage flows based on SDN proactive and reactive strategies. Furthermore, we adopt a distributed rate allocation strategy that calculates the optimal routing paths for a whole set of concurrent flows while coordinating between available network resources and services' requirements. These proposed methods are adopted in Chapter 4 and profoundly detailed in Chapter 5, where we consider not only a small set of flows but huge sets with thousands of flows.

3.1.4 Objective 4: Develop highly effective energy-aware data transfer mechanisms for building smart-sustainable environments

The intensive setup of multiple IoT-based devices leads to tremendous growth in the amount of generated electric data. Therefore, the energy-awareness aspect presents a fundamental concern that should be addressed in the design of sustainable IoT-based environments. The existing works either focus on fulfilling heterogeneous QoS requirements or on reducing the energy consumption problem. Using SDN's flexibility and programmability features, our proposed approach efficiently reduces the infrastructure energy while establishing dynamic routes for heterogeneous services and aggregating corresponding flows across activated network resources. The proposed method and detailed experiments are presented in Chapter 4.

3.1.5 Objective 5: Develop lightweight fine-grained traffic analysis engines for identifying heterogeneous smart environment services

Despite the worldwide technological advancement, there is always a gap of a deep understanding for all network traffic patterns, particularly with the evolving nature of heterogeneous network traffic, technologies, and related protocols. Furthermore, the existing mechanisms for traffic identification (e.g., port and payload-based inspection approaches) are either unreliable or complex and resource-hungry functionalities. We propose an up-to-date technical survey based on machine and deep learning mechanisms, in addition to a detailed review of the critical features for IoT and non-IoT service identification. This comprehensive framework, which will be described in Chapter 6, facilitates selecting, designing, and deploying lightweight fine-grained service differentiation mechanisms while protecting user privacy.

3.2 General methodology

The above discussed research questions and objectives are tackled through three consecutive methodologies, wherein new strategies and protocols have been proposed and deployed. The presented approaches are consistent for enhancing the overall network performance, including service assurance, resource optimization, and energy reduction. Different smart environments use cases (e.g., smart industry and education) have been considered to review their requirements and characteristics from various perspectives, thereby validating the proposed network functionalities' commonness over different SDN-based intelligent environments. The three methodologies are resumed as follows:

3.2.1 Methodology 1: Service assurance and resource optimization over Industrial Internet of Things (IIoT)

This methodology aims to support the digital transformation of the industrial environments from traditional forms to intelligent, connected, and low-carbon and sustainable industrial strategies. We propose a Route Optimization and Service Assurance scheme, named ROSA, over energy-efficient Software-Defined Industrial-IoT networks. First, as part of objective 1, we present an adaptive software-defined based IIoT architecture that provides flexible, dynamic and real-time monitoring and configuration of the underlying industrial infrastructure. We also review the main characteristics of industrial communications, including eMBB, mMTC, and URLLC services, where we categorize them using delay, packet loss, and bandwidth requirements into two main traffic classes, named *DetNet* and *Scheduled* flows. The former class requires "determinism", i.e., it includes network flows requiring deterministic forwarding delay value with extremely low packet loss (e.g., URLLC and delay sensitive eMBB). The *Scheduled* class includes the enormous amounts of traffic injected into the network due to the large entities getting access to the Internet (e.g., mMTC) and traditional traffic demands that are not sensitive to delay but are characterized by high-rate requirements (e.g., eMBB). These two classes play a crucial role in appropriate algorithm design for service assurance and resource optimization. Hence, as part of objectives 2 and 3, we formulate the routing problem through multi-constrained shortest path models that take into account the type of traffic, the associated QoS requirements, the constrained network resources, and the network energy reduction. Because of the NP-hard complexity of the proposed models, we introduce a centralized platform using the SDN multi-programmability feature and the GEN-LARAC mechanism, which is based on the Lagrangian relaxation theory. The proposed scheme is composed of triplet algorithms. One algorithm is devised to conduct a parallel routing mechanism, while a pair of routing algorithms is designed to manage the *DetNet* and *Scheduled* classes of traffic. Typically, the non-controlled management of the different traffic classes, particularly the Scheduled flows, creates network congestion and subsequently leads to performance degradation of the mission-critical industrial services. Therefore, assuring each flow's QoS requirements (objective 2) while optimizing network resources with efficient rate allocation mechanisms (objective 3) is the pole of our ROSA scheme. Furthermore, to build sustainable industrial networks, we address the joining problem between the heterogeneous service assurance, resource optimization, and energy awareness aspects, which is still an open issue with many challenges. Thus, using the flexibility and programmability aspects of SDN, we introduce an energy-aware data transfer mechanism (objective 4). The proposed approach efficiently reduces the infrastructure energy while establishing dynamic routes by aggregating *Scheduled* flows across activated network resources. With this energy awareness strategy, our

ROSA scheme supports the necessity of performing good tradeoffs based on the flow type between service assurance, resource allocation optimization, and energy consumption reduction to enhance the overall network performance. The obtained results confirm that the proposed ROSA scheme with large numbers of heterogeneous flows has significantly improved the total performance of the IIoT network in terms of service assurance and reliability, resource utilization optimization, and energy consumption reduction.

3.2.2 Methodology 2: Service assurance and resource optimization over Internet of (Campus) Things

In this methodology, we tackle the intelligent education application that boosts the efficiency of learning and creativity through smart educational environments. We manage massive heterogeneous flows generated simultaneously from thousands of interconnected devices to optimize constrained network resource allocation and ensure the heterogeneous required QoSs. Thus, we propose a multi-flow management scheme over a software-defined smart digital campus network named Service and Resource Aware Flow Management (SRAFM). Thus, as part of objective 1, we discuss the design challenges to deploy a unified fully-programmable architecture that controls wired and wireless software defined-based smart campus networks. Furthermore, since traffic characterization presents a fundamental network functionality to support service assurance and resource optimization, we provide an overview of the different types of services and their QoS requirements on the basis of delay and bandwidth characterization. Accordingly, we identify the design challenges confronting the characterization of a large number of heterogeneous flows in a fully-programmable architecture, and we propose a distributed end-host-based plane for flow characterization based on a combined approach of device and service identification. While designing this specific network level in the architecture, we take into consideration the end-user's privacy, the processing time, the controller overhead, and the network bandwidth consumption. SRAFM controls not only delay-sensitive flows, as do prior works, but all smart campus network flows to ensure service QoS requirements (objective 2) and to optimize network resource allocation (objective 3). The proposed centralized optimization framework manages flows based on SDN proactive and reactive strategies and optimizes the system cost in terms of joining resource cost and path loss. Network functionalities, including QoS provisioning and resource optimization-aware routing, are formulated as mixed-integer linear programming problems. Due to its NP-hard complexity, we propose an approximation algorithm based on the Log-det approximation function to relax the problem. Then, by using the Lagrangian Dual Decomposition (LDD) and Subgradient methods, we decompose the relaxed problem into per-flow sub-problems that can be solved simultaneously in a decomposed fashion to calculate the optimal routing paths for the whole set of concurrent flows in the network on the basis of a distributed rate allocation design. This strategy is more challenging than most prior schemes that often address this issue with a flow-per-flow routing strategy. In fact, though the flow-by-flow technique can reach a fast routing decision, it cannot provide an optimal global allocation for the whole set in the network; in addition in some special cases, it may reduce the network performance by wrong routing decisions in advance. Meanwhile, the SRAFM routing strategy finds an optimal solution not only for each flow independently but for the whole set of flows while performing coordination between their various requirements and the available resources. Compared to the well-known benchmarks in QoS provisioning and resource optimization aware routing problem, our experimental results show promising performance in terms of reduction in system cost, end-to-end delay, average rate allocation, and rejected flow percentages, from different aspects (e.g., the number of simultaneous heterogeneous flows, QoS provisioning, characterization impacts, and network scalability).

3.2.3 Methodology 3: Flow-based analysis for IoT and non-IoT lightweight fine-grained service identification

In this methodology, we tackle smart environment traffic analysis and service identification functionalities to enable administrators to be fully aware of the network content, and thereby proactively optimize service assurance and resource allocation. Hence, it focuses on objective 1 and objective 5 (Chapter 6). First, since the performance of traditional classification approaches (e.g., port and deep packet inspection-based mechanisms) has become inefficient due to the rise of different emerging technologies, such as the appearance of new applications using non-standardized port numbers and traffic encryption mechanisms, we adopt the flow-based inspection approach. In the literature, multiple sophisticated machine learning-based mechanisms have been proposed to increase classification efficiency, wherein the performance of each traffic classifier in terms of accuracy, speed, and resource consumption, depends not only on the differences among algorithms and their specific configuration but also on the selection of the important features for classification. Therefore, this methodology provides a technical survey (e.g., (Finsterbusch *et al.*, 2013; Guck *et al.*, 2018)) that offers the research and development community a machine and deep learning-based comprehensive evaluation framework with regard to a various set of flow-based features distinguishing heterogeneous services. This platform serves as a map for the design of lightweight fine-grained service differentiation engines in smart environments while supporting the evolving nature of network traffic, users' privacy, real-time operations, and high classification accuracy. First, we present a feature importance review that determines the best minimal set of attributes characterizing IoT and non-IoT network services. The produced sets of features consider the type of network application, early-stage identification, and the correlation between features in order to filter irrelevant and redundant attributes, consequently preserving the classification accuracy and decreasing the computational complexity. Second, to review traffic analysis and classification, we carry out rigorous evaluations and comparisons between different machine learning and deep learning-based techniques, which are the most deployed modules in the state-of-the-art for traffic analysis and service differentiation. In particular, while deep learning deployment is still thorny and presently less approved in network traffic classification, we justify the deep learning high complexity and training requirements in heterogeneous intelligent environments. Our proposed framework differs from the current published works for different reasons. First, we do not focus either on identifying IoT-based demands or on identifying the general-based demands, but on both, since they coexist and share the same network resources in smart environments. In addition, through these two main classes, we fill the gap in the literature, and we perform fine-grained analytics for identifying specific network services (e.g., video streaming, smart TP-Link Cam, and smoke detector). Another reason is that the existing studies have focused either on machine learning or deep learning approaches, while there is a great need with the evolving nature of traffic for studying and comparing both methods. Moreover, to contribute to the design of more appropriate classifiers, we investigate not only one particular mechanism (e.g., random forest and naïve Bayes in Sivanathan *et al.* (2018) or SVM and unsupervised k-means in Fan & Liu (2017)), but various machine learning and deep learning-based classifiers, while providing extensive comparisons and recommendations. We evaluate the performance of the different deployed classifiers in the platform from different aspects, including the classification accuracy and the time and space complexity, using common sets of traffic traces with corresponding ground truth and under the same hardware. The experimental results show that the random forest (machine learning approach) achieves the highest accuracy for IoT and non-IoT application identification. However, it is highly dependent on the feature selection process, and it may be inefficacious for real-time classification with an extensive training dataset and a large number of decision trees. On the other hand, while the LSTM (deep learning mechanism) provides accurate results, it eliminates the need for handcrafted feature engineering by automatically learning important features from the raw input data. However, it also requires a high computational cost. Finally, we recommend the deployment of a multi-stage engine, including CNN (deep learning) and random forest mechanisms, to be implemented over distributed architectures to preserve end-user's privacy and to ensure high classification accuracy with low computational complexity.

CHAPTER 4

PARALLEL ROUTE OPTIMIZATION AND SERVICE ASSURANCE IN ENERGY-EFFICIENT SOFTWARE-DEFINED INDUSTRIAL IoT NETWORKS

Yosra Njah [a] , Mohamed Cheriet [a]

[a] Department of Automated Production Engineering, École de Technologie Supérieure (ÉTS), 1100 Notre-Dame Ouest, Montréal, Québec, Canada H3C 1K3

Published in IEEE Access, February 2021

Abstract

In recent years, the Industrial world has been embracing new digital technology, including the Internet of Things (IoT) paradigm that promises revolutionizing-prospects in numerous industrial applications. However, many deployment challenges related to real-time big data analytics, service assurance, resource optimization, energy consumption, and security awareness are raised. In this work, we focus on service assurance and resource optimization, including energy consumption challenges over Industrial Internet of Things (IIoT)-based environments since the existing network routing algorithms cannot meet the strict heterogeneous quality of service (QoS) requirements of industrial communications while optimizing resources. We take advantage of the flexibility and programmability offered by the promising software-defined networking paradigm, and we propose a centralized route optimization and service assurance scheme, named ROSA, over a multi-layer programmable industrial architecture. The proposed solution supports a wide range of heterogeneous flows, such as ultra-reliable low-latency communications (URLLC) and bandwidth-sensitive services. The routing optimization problems are formulated as multi-constrained shortest path problems. The Lagrangian Relaxation approach is used to solve the NP-hard complexity. Hence, we deploy a pair of parallel routing algorithms run according to the flow type to ensure QoS requirements, efficiently allocate constrained resources, and enhance the overall network energy consumption. We conduct extensive simulations to validate the proposed ROSA scheme. The experimental results show promising performance in terms of reducing bandwidth utilization by up to 22%, end-to-end delay at least by 21%, packet loss by more than 19%, flow violation by about 16%, and energy consumption up to 14% as compared to well-known benchmarks in QoS provisioning and energy-aware routing problem.

Keywords: Industrial Internet of Things (IIoT), Software-Defined Networking (SDN), multiprogrammability, traffic engineering, Quality of Service (QoS), energy awareness, resource optimization.

4.1 Introduction

Industrial environments are going through a full digital transformation by embracing intelligent systems empowered by advanced analytics (e.g., 5G, cyber-physical systems, cloud computing, big data, and artificial intelligence). One of the smart industry's key enabling technologies is the Industrial Internet of Things (IIoT) paradigm, which introduces the set-up of the Internet of Things (IoT) technology into the industrial sector (Sisinni *et al.*, 2018; Zanella *et al.*, 2014). The IIoT, interchangeably referred to as the Industrial Internet, represents particularly one of the pillars for automation and data exchange in manufacturing technologies, i.e., the next industrial revolution (called Industry 4.0) (Da Xu *et al.*, 2014; Wollschlaeger *et al.*, 2017).

Hence, numerous opportunities have been created to boost smart management and production in various industrial sectors, such as healthcare, agriculture, transportation, and utility companies (Faheem *et al.*, 2018). Each IIoT application consists of various services that aim to improve industrial production lines and safety, such as design engineering and manufacturing, inventory tracking, predictive maintenance, and environmental security monitoring (Wan *et al.*, 2017). Real-time data acquisition, analytics, and remote control enable smart objects to operate autonomously while anticipating potential failures and critical situations, as shown in Figure 4.1, which depicts an example of an IIoT-based smart factory.

This heterogeneous connected world of smart devices, data, and people enables the promotion of business decisions and operational processes, which leads to unprecedented levels of efficiency and productivity. Numerous industrial companies, such as General Electric (General-Electric, Accessed: October 20, 2020), Daimler (Daimler, Accessed: October 20, 2020), Nokia (Nokia,

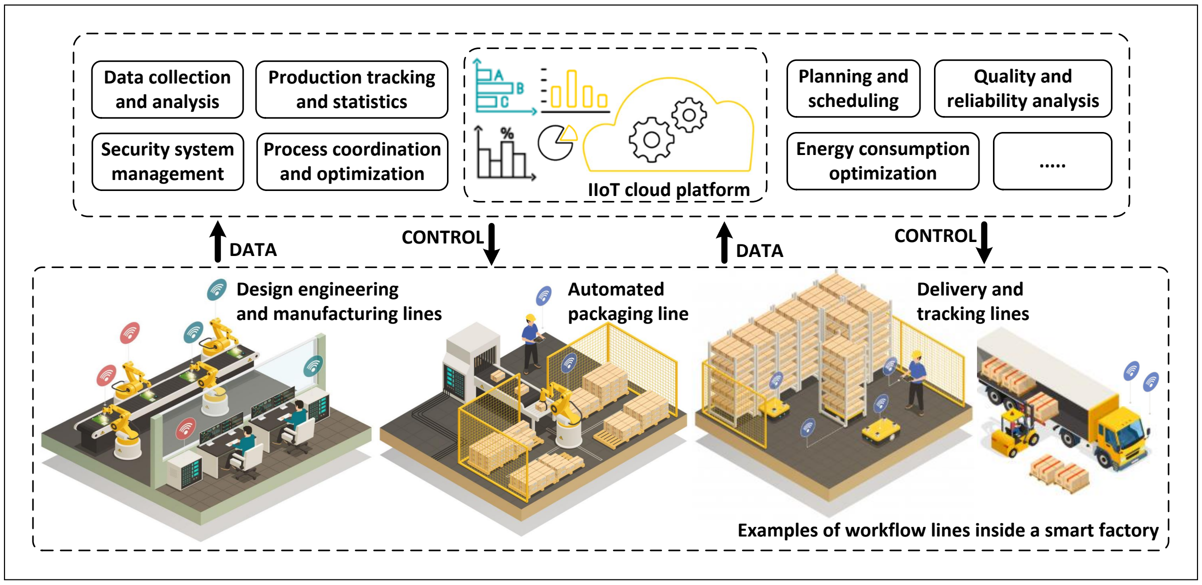

Data collection and analysis
Production tracking and statistics
Security system management
Process coordination and optimization
IIoT cloud platform
Planning and scheduling
Quality and reliability analysis
Energy consumption optimization
.....
DATA
CONTROL
DATA
CONTROL
Design engineering and manufacturing lines
Automated packaging line
Delivery and tracking lines
Examples of workflow lines inside a smart factory

Figure 4.1 Example of an IIoT-based smart factory environment

Accessed: October 20, 2020), and Thames Water (Thames-Water, Accessed: October 20, 2020), are revolutionizing smart manufacturing to boost production efficiency, reliability, and safety. The General Electric Company predicts the benefits of the IIoT will comprise 46% of the global economy. In the energy sector, they calculate an impact of 100% on energy production and 44% on energy consumption globally. It is also expected that this digital transformation of the entire value chain will add up to 14.2 trillion USD to the global economy by 2030 (General-Electric, Accessed: October 21, 2020) (Serpanos & Wolf, 2017).

However, with this digital transformation across all industrial domains, several network challenges in terms of flexibility, reliability, scalability, and security are raised (Vitturi *et al.*, 2019) (Haseeb *et al.*, 2020). On the one hand, a smart industrial network generates thousands or even millions of concurrent heterogeneous flows. Consequently, constrained network resources (e.g., bandwidth and devices) acquire unprecedented demands, which raise different issues for network administrators in terms of advanced solutions' implementation related to network architecture, flow management, and resource allocation optimization. Furthermore, regarding the massive number of heterogeneous flows, the quality of data transmission is particularly critical. It has to be optimized to be as timely and reliable as possible according to each industrial service's requirements (Qi & Tao, 2019). For instance, factory automation services (e.g., high-speed assembly, packaging, palletizing, etc.) that need to be monitored automatically at all times, at different production sites, and without human intervention are generally considered to be highly challenging in terms of latency and reliability (Schulz *et al.*, 2017). Network latency, with a few seconds or even a few milliseconds as in high-precision manufacturing, has a significant impact on the entire workflow of production lines. Hence, industrial systems require accurate quality of service (QoS) provisioning to operate properly under three communication scenarios, named enhanced mobile broadband (eMBB), massive machine-type communication (mMTC), and ultra-reliable low-latency communication (URLLC) (Cheng *et al.*, 2018).

On the other hand, the dynamic adaptability of the network to any characteristic change (e.g., demand rate, resource availability, and failure recovery) represents a fundamental aspect that should be considered during the design of network architecture in order to ensure more efficiency and flexibility. Indeed, with the traditional networks, it is highly challenging to ensure flexible production lines' reconfiguration, especially with heterogeneous resources and thousands or even millions of real-time flows. Accordingly, if we need to provide a new cooperation mechanism, we should update the communication protocols of all the related devices, one by one, which leads to reduced network efficiency and degraded service performance. Therefore, considering the great number of heterogeneous wired and wireless communication systems, the deployment of an IIoT-based environment requires the setup of a holistic smart architecture that enables dynamic, flexible, and efficient remote controls and configurations (Njah *et al.*, 2020). Moreover, it is crucial to highlight that the intensive setup of multiple IIoT-based devices leads to tremendous growth in the amount of generated electric data. Therefore, energy-awareness also presents a fundamental concern that should be addressed in the design of an IIoT-based platform to reduce network energy consumption and pollution emissions (Wang *et al.*, 2016) (Xiang *et al.*, 2016).

Thus, to leverage a sustainable smart industrial application, it is fundamental to devise compromises between network service assurance, resource optimization, and energy consumption. Motivated by this statement, we answer the following questions: *How to flexibly configure facilities and make production process changes in real-time? How to ensure an enhanced*

QoS for numerous business-critical operations? How to optimize the massive connectivity and resources with energy and cost-effective strategies?

To answer the above questions, we present in this paper a Resource Optimization and Service Assurance scheme, named ROSA, over energy-efficient Software-Defined Industrial-IoT networks. In our work, the Software-Defined Networking (SDN) paradigm presents a key enabling technology to fulfill programmability, automation, and flexibility in the network (Xia *et al.*, 2015; Kreutz *et al.*, 2015). The proposed scheme is different from existing works for two main reasons. First, numerous research studies address service assurance and resource optimization problems in different SDN-based environments, such as smart home, smart campus, and data-center networks (Karakus & Durresi, 2017; Habibi Gharakheili *et al.*, 2017; Njah *et al.*, 2020; Zhu *et al.*, 2020). However, only a few research studies deal with the SD-IIoT network, especially the joining problem between the heterogeneous service assurance, resource optimization, and energy awareness aspects, which is still an open issue with many challenges for building sustainable and smart industrial networks. Second, most of the existing QoS-aware flow management solutions address problems for a specific type of service, particularly the time-sensitive one (e.g., video streaming), by solving a delay-constrained least-cost (DCLC) problem. This strategy does not fit the QoS requirements of multiple heterogeneous services in smart industrial networks. In this paper, we propose a global approach that addresses not only one type of service, but various services characterized by heterogeneous QoS requirements. These services are categorized into classes managed within a parallel fashion in the network using the multi-programmability feature of SDN. The specific contributions of this paper are summarized as follows:

1. **SD-IIoT programmable network architecture and traffic characterization design:** We propose an adaptive software-defined based IIoT architecture that provides flexible real-time monitoring and configuration of the underlying industrial infrastructure. We also review the main characteristics of industrial communications, including eMBB, mMTC and URLLC services. We categorize them using delay, packet loss, and bandwidth requirements into two main traffic classes, named *DetNet* and *Scheduled* flows, to promote resource optimization and service assurance.

2. **Service aware routing and resource optimization:** We formulate the QoS routing problem as multi-constrained shortest path models. This takes into account the type of traffic, the associated QoS requirements, and the constrained network resources. Because of the NP-hard complexity of the proposed models, we introduce a centralized multi-program platform based on the Lagrangian Relaxation approach. The proposed scheme is composed of triplet algorithms. One algorithm is devised to conduct a parallel routing mechanism, while a pair of routing algorithms is designed to manage the *DetNet* and *Scheduled* classes of traffic.
3. **Energy awareness:** Using the flexibility and programmability aspects of SDN, we introduce an energy-aware data transfer mechanism. The proposed approach efficiently reduces the infrastructure energy while establishing dynamic routes by aggregating flows across activated network resources. With this energy awareness strategy, our ROSA scheme supports the necessity of performing compromises based on the flow type between service assurance, resource allocation optimization, and energy consumption reduction to enhance the overall network performance.
4. **Simulation and evaluation:** We conduct extensive simulations using the Floodlight SDN controller and the Mininet network emulator to prove the validation of the proposed platform. The experimental results show promising performance in terms of reducing bandwidth utilization up to 22%, end-to-end delay at least by 21%, packet loss by more than 19%, flow violation by about 16%, and energy consumption up to 14%, as compared to well-known benchmarks in the QoS provisioning and energy-aware routing problem, i.e., SWAY (Saha *et al.*, 2018), ERRS (Long *et al.*, 2018), and LARAC (Juttner *et al.*, 2001).

The remainder of this chapter is organized as follows. Section 4.2 reviews relevant related works. Section 4.3 discusses the programmable SDN-based architecture and traffic characteristics for smart industry systems. Section 4.4 presents the ROSA optimization problems. Section 4.5 presents the parallel ROSA platform. The prototype implementation and results analysis are presented in Section 4.6. Finally, the paper concludes and discusses future works in Section 4.7.

4.2 Related work

The literature on the SDN-based QoS-aware routing problem is vast. Thus, in this section, we review recent relevant works from the perspective of QoS-aware routing and optimization over software-defined based networks in general and, specifically, industrial ones that are attracting increasing attention from both academia and industry.

4.2.1 Unicast QoS routing approach

The authors of (Guck *et al.*, 2016), (Guck *et al.*, 2018), and (Zoppi *et al.*, 2018) are pioneering in the field of QoS-aware routing for SDN-based industrial networks. Guck *et al.* (2016) have proposed a centralized QoS control framework in SDN-based industrial networks. The proposed solution manages QoS through a function split between DCLC routing and resource allocation using network calculus. Guck *et al.* (2018) provided a survey about the QoS routing problem in SDN-based networks. The survey presents a comparative study of 26 DCLC-based algorithms within a four-dimensional (4D) evaluation framework. The four dimensions correspond to the topology type, two forms of scalability of topology, and the tightness of the delay constraint. They conclude with the outperformance of the LARAC algorithm (Juttner *et al.*, 2001) in the vast majority of the evaluations. Zoppi *et al.* (2018) devised a reliability-aware dynamic scheduler for industrial networks where the LARAC scheme was chosen to perform QoS-aware routing and to guarantee the delay bound of each application. However, in these works (i.e., (Guck *et al.*, 2016), (Guck *et al.*, 2018), and (Zoppi *et al.*, 2018)), the proposed frameworks are designed to ensure a strict end-to-end delay parameter, which is typically required for industrial systems, but not enough to fulfill heterogeneous QoS industrial communications. Furthermore, all these works do not address the energy awareness problem, which is a crucial aspect in the design of an IIoT-based platform, since industrial networks are characterized by high energy consumption.

It is noteworthy to mention that the LARAC algorithm (Juttner *et al.*, 2001) was also adopted in several SDN-based QoS routing studies, e.g., (Egilmez *et al.*, 2013) and (Rezaee & Moghaddam, 2019). Egilmez *et al.* (2013) presented an SDN-based analytical framework to optimize the

QoS-aware routing for the video streaming service. They assigned the highest routing priority to the video streaming service, while the rest of the services were considered as best-effort flows. Then, they transformed the QoS-aware routing into a Constrained Shortest Path (CSP) problem solved using the LARAC algorithm (Juttner *et al.*, 2001). Like (Guck *et al.*, 2016), (Guck *et al.*, 2018), and (Zoppi *et al.*, 2018), Egilmez *et al.* (2013) addressed only the QoS routing for video streaming and did not consider the network traffic heterogeneity with various constraints. Another relevant example is in (Rezaee & Moghaddam, 2019), where Rezaee *et al.* presented a wide area measurement system using the SDN paradigm to measure, collect, and analyze data in the smart grid network. They considered two different classes of traffic. Thus, available bandwidth resources were assigned to high-priority packets that contained new data, while less-important packets that contain older data were dropped using a queue management approach. The LARAC algorithm (Juttner *et al.*, 2001) was deployed to solve the DCLC problem (applied to the bandwidth) and find the best path. However, in addition to the unicast proposed approach, which does not fit the large number of heterogeneous services in smart networks, another disadvantage in this work resides in dropping packets containing older data, which significantly affects the network services' reliability.

4.2.2 Multicast QoS routing approach

Long *et al.* (2018) proposed a routing scheme to enhance communication latency performance and energy consumption in IIoT networks. Within each path selection, they aggregated network data through different clusters across a hierarchical framework to reduce energy while estimating the latency by using hop count. However, while Long *et al.* (2018) considered reducing energy as an essential factor, they did not consider heterogeneous application-dependent requirements, such as loss-sensitive, delay-sensitive, or a combination of any QoS traffic types. Considering only hop count is not enough to validate low-latency transmission. Furthermore, aggregating network flows without considering prioritization management and efficient bandwidth allocation mechanisms leads to network congestion and service performance degradation. Therefore, the energy and hop count-based proposed model is insufficient for handling heterogeneous IIoT services, especially that the requirements of industrial systems in terms of delay and reliability provisioning are more important than energy saving.

Naeem *et al.* (2020) proposed an SDN-based energy-efficient and QoS-aware parallel routing scheme for IIoT-based smart healthcare networks. The authors considered a max-flow-min-cost optimization problem with multi-constrained QoS parameters while characterizing medical services as jitter-sensitive, loss-sensitive, and delay-sensitive flows. The objective was to maximize flows gathering over the active resources while minimizing the bandwidth costs and fulfilling the QoS requirements. However, Naeem *et al.* (2020), in the SDN-based smart healthcare network, adopted the Yen's-K shortest path algorithm (Yen, Accessed: October 20, 2020) that was taken into consideration in (Guck *et al.*, 2018), where Guck *et al.* proved the outperformance of the LARAC algorithm (Juttner *et al.*, 2001) as compared to 26 SDN-based routing algorithms. Similarly, Saha *et al.* (2018) proposed a parallel QoS routing scheme running two separate single-metric models over SD-IoT networks. The first model was devised for delay-sensitive traffic, and the second one was designed for loss-sensitive traffic. However, the Yen's-K shortest path algorithm (Yen, Accessed: October 20, 2020) was used as the basis for both of them. Furthermore, considering the delay and the loss sensitivities as the two main characteristics for classifying all SD-IoT network traffic is inefficient, since, for example, URLLC services are characterized equally with these two parameters. Thus, to which class would such services be assigned, i.e., according to (Saha *et al.*, 2018), should URLLC services be assigned to the loss-sensitive class or the delay-sensitive class?

Differing from existing works, we propose a global approach that addresses not only one type of service, but various services characterized by heterogeneous QoS requirements. We address the issue of multi-constrained QoS provisioning with efficient resource utilization and energy consumption in IIoT-based networks, the related research on which, to the best of authors' knowledge, has not yet been sufficiently explored. Thus, using the flexibility and multi-programmability feature of SDN and the GEN-LARAC mechanism (i.e., the best algorithm according to (Guck *et al.*, 2018)), we design different algorithms to be run in parallel on the basis of the type of traffic in order to achieve a balance between QoS provisioning, resource optimization, and energy awareness.

4.3 SD-IIoT network architecture and traffic characterization

In a smart industrial environment, all kinds of intelligent equipment supported by wired and wireless networks are widely adopted, and both real-time and delayed communications coexist. Therefore, in this section, we propose an IIoT architecture based on the advancement of SDN technology to introduce more programmability and flexibility within a smart industry network. Then, we discuss the QoS requirements that characterize industrial communications.

4.3.1 Software-defined IIoT architecture

In a smart industrial-based environment, heterogeneous systems should interact effectively and flexibly to accomplish a large number of services while meeting the fundamental challenges, including timeliness, security, reliability, and scalability. In fact, with the traditional industrial networks, if we need to provide a new cooperation mechanism, all the communication protocols of all the related devices should be updated statically one by one, which leads to low industrial system efficiency, scalability, and interoperability (Lin *et al.*, 2017). Therefore, pure software methods are required to enable remote control and efficient configuration of all heterogeneous industrial network resources and services. In this regard, using the emerging software-defined based technology (Bera *et al.*, 2017; Macedo *et al.*, 2015), we introduce a flexible SD-IIoT programmable architecture, as shown in Figure 4.2. We divide the proposed architecture into four main planes: industrial end-host-based plane, data plane, control plane, and application plane.

1. **The industrial end-host-based plane:** This layer incorporates a variety of intelligent systems, including robots, automated guided vehicles (AGV), sensors and actuators, advanced meters, etc. The efficient interaction among industrial nodes is the foundation of the smart industrial paradigm. Thus, data forwarding between industrial nodes is performed using combinations of wireless and wired connections. Forwarding paths can be locally estimated using node-to-node negotiations (Xu *et al.*, 2018). However, in

some cases (e.g., unpredictable network information), it is challenging to meet real-time system management subject to traffic characteristics and constrained resources. Therefore, interoperability between industrial nodes must be continuously monitored by the centralized control plane (Al-Rubaye *et al.*, 2017).

2. **The SD-Industrial data plane:** This layer consists of all physical and virtual forwarding devices, such as SDN-based switches, gateways, access points, base stations, and routers. All the data plane devices present simple high-speed forwarding elements that are fully-programmable by the control plane. Consequently, whenever forwarding devices receive new traffic flows, they send routing-requirements through the southbound interfaces to the above control plane for evaluating resource allocation procedures.
3. **The SD-Industrial control plane:** This layer has a global view of the network status and is responsible for all network management decisions. Using a set of programs within the application plane, which is at the top of the SD-Industrial platform, the control plane monitors the network, supports each demand with the appropriate resources, and configures with new rules (flow engineering policies) the corresponding data plane's forwarding devices. Furthermore, the distributed controlling approach is designed to overcome the scalability issues (e.g., the increased complexity of large-scale heterogeneous networks, the continuous expansion in the flow number, and the requirements for detailed analysis and real-time monitoring). This strategy enables the deployment of multiple processing nodes, such as a cluster of controllers and fog nodes that jointly manage the data plane, wherein each node controls only a specific part of the network resources and its corresponding heterogeneous flows (Njah *et al.*, 2020; Li *et al.*, 2018a).
4. **The SD-Industrial application plane:** This layer configures the control plane with the specific network management tasks through the northbound interfaces. It must include many industrial engines, such as data analysis, network security, service assurance, resource optimization, topology discovery, and failure prediction, to ensure efficient and secure operational processes. However, despite the promising SDN features, it is still a great challenge (but an urgent need) to design many engines and functions over the application plane for the IIoT network management. Accordingly, this proposed work focuses on meeting

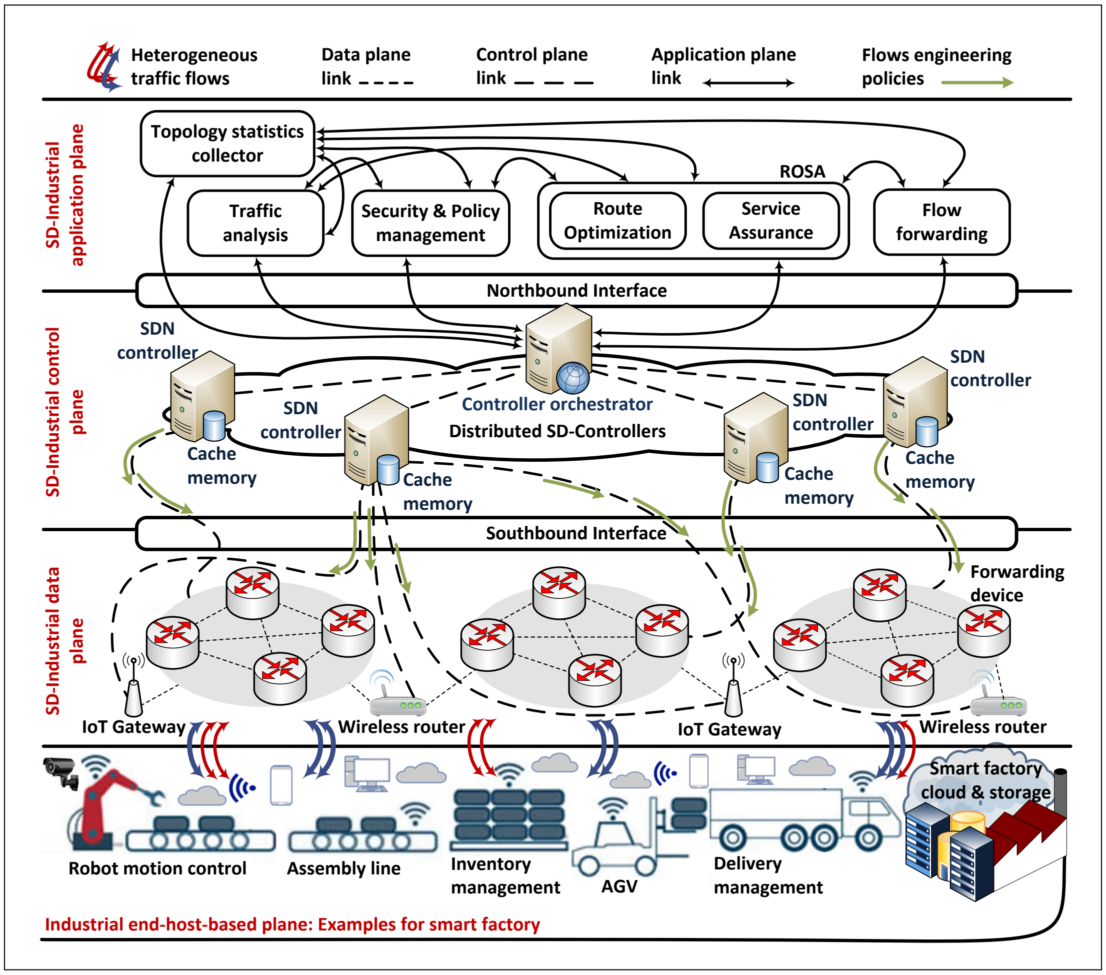


Figure 4.2 SD-IIoT programmable network architecture in the context of smart factory

concurrent heterogeneous services' requirements while optimizing resources, including the overall energy consumption of the IIoT network.

As shown in Figure 4.2, we propose ROSA, a centralized Route Optimization and Service Assurance module. It enables the control plane to configure the data plane according to each service's requirements while allocating the available network resources with an optimized strategy. It is also noteworthy that the number of SDN flow-rules adopted by the industrial forwarding devices is limited. Therefore, we propose the deployment of cache memory in SDN controllers, as depicted in Figure 4.2. This technique enables the controller quickly to answer frequently requested flows. In other words, if a switch fails to match an incoming flow, it reports the routing requirements to the controller. The controller first performs the flow-cache memory matching. If it finds the required rules, it configures them into the corresponding switches. Otherwise, it calls ROSA to compute the new route and assign new engineering policies to selected paths' devices. The deployment of the cache memory approach enables reducing the time and workload of route computation. Consequently, it enhances the control plane's efficiency and the service quality (Li *et al.*, 2016).

4.3.2 Characteristics of industrial services

In addition to the traditional industrial mobile broadband (MBB) traffic flows characterized by high data rate demands, the Industrial IoT-based network includes new use cases characterized by new heterogeneous requirements. On the one hand, low latency and high reliability characteristics are fundamentally required by many critical industrial services, which fall under the umbrella of Ultra-Reliable Low-Latency Communications (URLLC). For example, some factory automation services' reliability requirements are typically 10^{-9} packet loss rate, while the end-to-end delay requirements are less than 10 ms (Schulz *et al.*, 2017). On the other hand, besides the various manufacturing data related, for example, to workers' behavior information, disturbance information, equipment status, etc. that need high-density sensing (Wan *et al.*, 2016), certain traffic flows corresponding to orchestration and interconnection between heterogeneous devices and networks require tremendous amounts of data (Cheng *et al.*, 2018). Those kinds of traffic are classified as massive Machine-Type Communications (mMTC), which also lead to significant increases in network energy consumption (Zhang & Ji, 2019). Finally, it is noteworthy that various QoS requirements characterize industrial services, wherein flows' rates could be regular, irregular, frequent, and non-frequent. For instance, in conventional situations, services related to the production lines inside a smart factory are characterized by regular and frequent rates. Meanwhile, data covering events and alarms (e.g., safety-critical services) are irregular, non-frequent, and need to be detected immediately to prevent emergencies (Mocnej *et al.*, 2018). Therefore, industrial communications should be characterized according to QoS requirements (e.g., rate, packet loss, and latency), then categorized into traffic classes in order to be supported by specific network route planning mechanisms for service assurance and resource optimization.

In our work, we are considering two main classes of traffic, namely *DetNet flows* and *Scheduled flows*. The former class requires "determinism", i.e., it includes network flows requiring deterministic forwarding delay value with extremely low packet loss. Hence, each *DetNet flow* is characterized by the maximum end-to-end delay and the packet loss probability requirements (Finn *et al.*, 2017). The *Scheduled* class includes the enormous amounts of traffic, which are injected into the network due to the large entities getting access to the Internet, as well as traditional traffic demands that are not sensitive to delay but are characterized by high-rate requirements. This second class of traffic perilously consumes a great deal of network resources. Typically, the non-controlled management of the different traffic classes, particularly the *Scheduled flows*, creates network congestion and subsequently conducts to performance degradation of the mission-critical industrial services. Therefore, assuring each flow's QoS requirements while optimizing network resources with efficient rate allocation mechanisms is the objective of our ROSA scheme.

4.4 ROSA problem formulation

In this section, we formulate the ROSA problems to manage optimally the different classes of traffic in the SD-IIoT network while taking into consideration the constrained network resources, the service-dependent requirements, and the network energy consumption.

Let us consider the SD-IIoT network as a connected graph $H = (V, L)$, where V is the set of all SDN-enabled nodes, and L is the set of communication links between them. Table 4.1 summarizes the main notations of the parameters used in this work.

4.4.1 Constrained network resources

As constrained network resources, we consider the limited number of software-defined rules in the flow table of each forwarding device and the limited available capacity in each link. Hence,

Table 4.1 Summary of main notations

Notation	**Description**
F	Set of the traffic demands (flows).
f_k	The k^{th} flow in F.
P	Set of paths for routing flow f.
p	A path, between source node s and destination node t, that could be used by flow f.
e	A directed link (i, j) outgoing from node i and incoming to node j.
$E(e)$	Energy of link e.
$D(e)$	Delay of link e.
$C(e)$	Resource capacity of link e.
$C^{res}(e)$	Residual capacity of link e.
$W(e)$	Bandwidth utilization ratio of link e.
$Q(e)$	Packet-loss probability of link e.
$Z(i)$	Number of flow-rules at node i.
D_f^{max}	Maximum end-to-end delay acceptable by a flow f.
Q_f^{max}	Maximum packet-loss probability acceptable by a flow f.
R_f^{min}	Minimum required rate by a flow f.
Z^{max}	Maximum number of flow-rules in an SDN-enabled node.

first, to determine whether or not a flow $f_k \in F$ is routed through a link e, we define an identity function $\delta_{f_k}(e)$, as shown in (4.1).

$$\delta_{f_k}(e) = \begin{cases} 1, \text{ if } f_k \text{ is routed on } e \in L, \\ 0, \text{ otherwise.} \end{cases} \tag{4.1}$$

As presented in table 4.1, the link e is defined as the link outgoing from node i and incoming to node j. We denote the neighborhoods of the switch $i \in V$, i.e., the set of all nodes connected directly to i, by $N(i)$. A flow f_k is forwarded on link $e \in L$ only if a particular flow-rule is configured at switch $i \in V$ to the switch $j \in N(i)$ (Naeem *et al.*, 2020).

Thus, the maximum number of flow-rules at switch i can be formulated as shown in (4.2), where the number of flow-rules associated with the k^{th} flow f_k at device i can be determined by (4.2a), and the total number of flow-rules associated with all flows $f_k \in F$ going through device i is defined by (4.2b). Finally, as shown in (4.2c), this flow-rule number should not exceed Z^{max}, the maximum number of flow-rules that can be configured at any software-defined forwarding device $i \in V$.

$$Z_{f_k}(i) = \sum_{j \in N(i)} \delta_{f_k}(i, j), \forall i \in V, \tag{4.2a}$$

$$Z(i) = \sum_{f_k \in F} Z_{f_k}(i), \forall i \in V, \tag{4.2b}$$

$$Z(i) \leq Z^{max}, \forall i \in V. \tag{4.2c}$$

On the other hand, the capacity constraint associated with each link $e \in L$ is determined in (4.3), where the residual bandwidth of a link $e \in L$ is defined by the remaining bandwidth after forwarding the corresponding flows $f_k \in F$ with their associated rate requirements through the link, as shown in (4.3a). Hence, the link's residual capacity of each link $e \in L$, should be greater than the network congestion level $\mathcal{Y}$, as shown in (4.3b). The network congestion level $\mathcal{Y}$ enables restricting the bandwidth utilization to a specific value (e.g., 20% of the capacity) in order to prevent congestion (Egilmez *et al.*, 2013).

$$C^{res}(e) = C(e) - \sum_{f_k \in F} R_{f_k}^{min} \delta_{f_k}(e), \forall \ e \ \in \ L, \tag{4.3a}$$

$$\mathcal{Y} \leq C^{res}(e), \forall \ e \ \in \ L. \tag{4.3b}$$

4.4.2 Heterogeneous service assurance, energy awareness, and multi-CSP formulation

With the ROSA scheme, we aim to select optimal routes to accommodate a set of heterogeneous flows F while ensuring the required QoS for each flow and maximizing the overall network performance. Therefore, we formulate two separate Constrained Shortest Path (CSP) models. The first one is devised for the *DetNet* traffic. It addresses a multi-constrained least-cost routing problem while taking into consideration the delay and the packet-loss requirements of each *DetNet* flow. The second model is devised for the *Scheduled* traffic and aims mainly to reduce the network energy consumption and ensure an efficient bandwidth allocation.

4.4.2.1 The DetNet routing problem

As detailed in Section 4.3.2, each *DetNet* flow, $f_k \in F$, needs a real-time transmission in addition to the strict reliability requirement. Therefore, the proposed *DetNet* optimization problem aims to minimize the delay, which is the most critical metric, while considering the delay and the packet-loss requirements of each flow, as shown in (4.4).

$$\min \quad D_{f_k}(p)\ ,\ \forall\, p \in P,\ \forall\, f_k \in F \tag{4.4a}$$

$$\text{s.t.} \quad D_{f_k}(p)\ \leq\ D^{max}_{f_k},\ \forall\, p \in P,\ \forall\, f_k \in F, \tag{4.4b}$$

$$Q_{f_k}(p)\ \leq\ Q^{max}_{f_k},\ \forall\, p \in P,\ \forall\, f_k \in F. \tag{4.4c}$$

Equations (4.4b) and (4.4c) define the delay and loss constraints. The threshold $D^{max}_{f_k}$ and $Q^{max}_{f_k}$ present the QoS requirements characterizing flow $f_k \in F$ in terms of delay and loss, respectively. The $D_{f_k}(p)$ and $Q_{f_k}(p)$ present the delay and loss of path p experienced by flow f_k, and they are determined as defined in (4.5) and (4.6), respectively. The delay on the path is computed by the sum of all the delays of the links in p, as shown in (4.5). The end-to-end loss rate probability on path p is calculated by the product of the individual packet loss ratios per link of all links belonging to p (Njah *et al.*, 2020), as shown in (4.6).

$$D_{f_k}(p) = \sum_{e \in p} D(e)\,\delta_{f_k}(e),\ \forall\, p \in P. \tag{4.5}$$

$$Q_{f_k}(p) = 1 - \prod_{e \in p} \Big(1 - Q(e)\,\delta_{f_k}(e)\Big),\ \forall\, p \in P. \tag{4.6}$$

4.4.2.2 The Scheduled routing problem

With the *Scheduled* routing model, we focus on reducing the network energy consumption while ensuring an efficient bandwidth allocation. We use the SDN flexibility to control, on the basis of the link's load, the energy metric (cost of the link) assignment (Cisco-CCNA, Accessed: October 20, 2020). Hence, the activated links are characterized by low cost-energy metrics, whereas high cost-energy metrics characterize the non-activated links. When selecting a path that minimizes the cost-energy, with this flexible SDN-based metric assignment, we prevent the activation of new network links, and we aggregate the *Scheduled* flows into the already activated network resources.

$$\min \qquad E_{f_k}(p), \ \forall\, p \in P, \ \forall\, f_k \in F, \tag{4.7a}$$

$$\text{s.t.} \qquad R^{min}_{f_k} \le C_{f_k}(p), \forall\, p \in P, \ \forall\, f_k \in F. \tag{4.7b}$$

Equation (4.7a) presents the objective function, aiming to minimize the energy while selecting network paths. $E_{f_k}(p)$, the end-to-end energy of path p experienced by flow f_k, is calculated by the sum of the energy metrics of all the links belonging to p, as shown in (4.8). As mentioned above, we set the energy metric $E(e)$ based on each link's activation with a corresponding load using the Cisco cost metric assignment approach (Cisco-CCNA, Accessed: October 20, 2020). Equation (4.7b) determines the bandwidth constraints of the *Scheduled* flows that should be met when selecting network paths. $R^{min}_{f_k}$ denotes the minimum rate requirement of flow $f_k \in F$, and $C_{f_k}(p)$ presents the capacity of path p. Equation (4.9) calculates $C_{f_k}(p)$, which is defined as the minimum residual capacity of all the links belonging to path p (Saha *et al.*, 2018).

$$E_{f_k}(p) = \sum_{e \in p} E(e)\, \delta_{f_k}(e), \ \forall\, p \in P. \tag{4.8}$$

$$C_{f_k}(p) = \min_{e \in p} \Big(C^{res}(e)\, \delta_{f_k}(e) \Big), \forall\, p \in P. \tag{4.9}$$

4.5 ROSA Scheme

To solve the formulated NP-hard problems efficiently, we design a parallel-based routing platform named ROSA. Since appropriate algorithm design plays a crucial role in network and service performance, we devise different flow-based algorithms using the corresponding optimization problem and the Lagrangian Relaxation approach for managing the *DetNet* and *Scheduled* classes.

4.5.1 The ROSA parallel-routing algorithm

Algorithm 4.1 takes as input the set of characterized *DetNet* and *Scheduled* flows with the correspondent QoS requirements of each flow, in terms of delay, bandwidth, and packet loss, as well as the network status (e.g., number of flow-rule, delay, available bandwidth, energy cost, etc.). The *ROSA parallel routing* algorithm outputs an optimal path for each flow $f_k \in F$, if it exists. Paths are calculated within a parallel computation strategy based on the *DetNet-routing* algorithm and the *Scheduled-routing* algorithm, which are called depending on the type of the flow, as shown in lines 1–9. The *DetNet* flows are prioritized compared to the *Scheduled* flows, as explained in Section 4.3.2. After configuring the engineering-rules into the appropriate devices, the energy metrics are updated with lower numbers across allocated routes, as shown in lines 4 and 8, in order to route as many future flows as possible through the activated resources.

The combination of service assurance, energy awareness, and resource allocation optimization in (4.2), (4.3), (4.4), and (4.7) introduces NP-hard problems. The main difficulty lies in the integrality condition of the variable δ_{f_k}, which has to be either 0 or 1. A plethora of QoS-aware routing algorithms has been developed to find optimal solutions (Guck *et al.*, 2018). In the design of our approach, we use as a building block the GEN-LARAC scheme, a generalization of the LARAC algorithm applicable to multiple metrics (Xiao *et al.*, 2016). The LARAC algorithm is one of the best approaches for solving CSP problems (Guck *et al.*, 2018; Zoppi *et al.*, 2018; Egilmez *et al.*, 2013; Rezaee & Moghaddam, 2019). It is a polynomial-time algorithm that efficiently finds the optimal route for a specific demand in $O([l + v\log(v)]^2)$ time complexity, where v represents the number of nodes and l the number of links in the topology. On the basis of

Algorithm 4.1 The ROSA parallel-routing algorithm

```
Input: Network $H(V, L)$  ▷ The available nodes, the number of flow-rule $Z(v)$ and
  $Z^{max}$ in each node $v \in V$, the available bandwidth $C^{res}(e)$ and $\mathcal{Y}$, the packet-loss
  $Q(e)$, the delay $D(e)$, and the energy-cost $E(e)$ of each link $e \in L$.
Input: Set of flows $F$, where each flow $f$ is characterized as $DetNet$ or $Scheduled$
  based on its QoS requirements.
Output: Set of routes that can forward the flows in $F$.
Parallel Routing ($H$, $F$)
1: while all flows $f_k \in F$ have not been forwarded do
     ▷ Prioritizing $DetNet$ flows.
2:   if $\exists$ $DetNet$ flow not forwarded then
3:       path ← Call ROSA DetNet-routing algorithm.
           ▷ Routing the $m^{th}$ $DetNet$ flow using path.
4:       ▷ Updating path energy with lower metrics.
5:       $m \leftarrow m + 1$, $k \leftarrow k + 1$.
     ▷ Routing $Scheduled$ flows.
6:   if $\exists$ $Scheduled$ flow not forwarded then
7:       path ← Call ROSA Scheduled-routing algorithm.
           ▷ Routing the $n^{th}$ $Scheduled$ flow using path.
8:       ▷ Updating path energy with lower metrics.
9:       $n \leftarrow n + 1$, $k \leftarrow k + 1$.
End of Parallel Routing
```

the Lagrangian Relaxation approach, LARAC computes paths using the Dijkstra algorithm while eliminating restrictions and aggregating them into the objective function using the Lagrange multiplier (Juttner *et al.*, 2001).

4.5.2 The ROSA DetNet-routing algorithm

Algorithm 4.2 takes as input, from Algorithm 4.1, the network's state and a *DetNet* flow f. Its objective is to find the best path that minimizes the delay while fitting the requirements of flow f and the constrained network resources. Hence, it solves the multicast routing problem in (4.4), which is a delay and packet-loss constrained least cost (delay) problem, as well as

Algorithm 4.2 The ROSA DetNet-routing algorithm

```
Input: Network $H(V, L)$, $flag \leftarrow false$
Input: A $DetNet$ flow $f$ from the set of flows $F$.
Output: The optimal route.
DetNet Routing ($H, s, t, D_f^{max}, Q_f^{max}, R_f^{min}$)
1:  Step 1: $p_D \leftarrow Dijkstra(s, t, D)$
2:  if $D_f(p_D) \leq D_f^{max}$ then
3:      if $Q_f(p_D) \leq Q_f^{max}$ then
          ▷ QoS requirements are validated.
4:          if $\mathcal{Y} \leq C_f^{res}(p_D)$ and $Z_f(p_D) \leq Z^{max}$ then
              ▷ Constrained resources are validated.
5:              return $p_D$ ▷ Optimal solution.
6:          else Delete current $[p_D]$ from $P$;
                ▷ Go to Step 1.
7:      else ▷ Go to Step 2.
8:  else return "no feasible solution." ▷ Violated flow.
9:  Step 2: $p_Q \leftarrow Dijkstra(s, t, Q)$
10: if $Q_f(p_Q) \leq Q_f^{max}$ then ▷ Go to Step 3.
11: else return "no feasible solution." ▷ Violated flow.
12: Step 3: while ($flag == false$)
13:     $\lambda \leftarrow \frac{D_f(p_D) - D_f(p_Q)}{Q_f(p_Q) - Q_f(p_D)}$
14:     $r \leftarrow Dijkstra(s, t, G_\lambda)$
15:     if $G_\lambda(r) = G_\lambda(p_D) = G_\lambda(p_Q)$ then
          ▷ Convergence to feasible solution.
16:         if $\mathcal{Y} \leq C_f^{res}(p_Q)$ and $Z_f(p_Q) \leq Z^{max}$ then
              ▷ Constrained resources are validated.
17:             $flag \leftarrow True$; return $p_Q$
                  ▷ Optimal solution.
18:         else Delete current $[p_Q]$ from $P$;
                ▷ Go to Step 2.
19:     else if $D_f(r) \leq D_f^{max}$ and $Q_f(r) \leq Q_f^{max}$
20:             then $p_Q \leftarrow r$
21:     else $p_D \leftarrow r$
End of DetNet Routing
```

problems (4.2) and (4.3). First, as shown in line 1, the Dijkstra algorithm is called to find the shortest route with the minimum cost in terms of delay. Then, it checks not only the strict delay and packet-loss requirements of flow f but also the validity of the constrained resources in terms of bandwidth and flow-rules number, as defined in problems (4.2) and (4.3). If all the conditions are satisfied, the optimal path is returned to Algorithm 4.1 for routing flow f and updating the allocated resources' energy metrics. Otherwise, if the delay of the computed path $D_f(p_D)$ does not satisfy the delay constraint, D_f^{max}, so there is no feasible solution for flow f (i.e., the demand is violated). If the delay is met, but the packet-loss is not satisfied, Algorithm 4.2 moves to Step 2 (line 9) for computing the shortest path in the network using the packet loss metric. If flow requirements are satisfied, but the resources are not sufficient, Algorithm 4.2 deletes the current solution from the set of paths P and returns to Step 1 to compute a new route that minimizes the delay in the topology while rechecking the same conditions. In Step 2, if the packet loss of the computed path $Q_f(p_Q)$ satisfies the constraint Q_f^{max}, Algorithm 2 moves to Step 3. Otherwise, there is no feasible solution.

In Step 3, Algorithm 4.2 computes paths using the Lagrange dual function while aggregating restrictions into the objective function (Xiao *et al.*, 2016). Considering that the delay plays the roles of cost and constraint, simultaneously, as detailed in (4.4), the aggregated cost function G_λ is defined as the aggregated sum of the delay and the weighted packet-loss of the link e, as shown in (4.10a). Furthermore, the Lagrange multiplier λ is determined in (4.10b), where p_D and p_Q present the shortest paths already computed in the previous steps in terms of delay and packet loss, respectively.

$$G_\lambda(e) = D(e) + \lambda \times Q(e), \forall\, e \in L. \tag{4.10a}$$

$$\lambda = \frac{D(p_D) - D(p_Q)}{Q(p_Q) - Q(p_D)}. \tag{4.10b}$$

Hence, as shown in line 14, Algorithm 4.2 computes the shortest path r using the Dijkstra algorithm applied on the link aggregated cost G_λ. If the condition $G_\lambda(r) = G_\lambda(p_Q) = G_\lambda(p_D)$ is satisfied, then the algorithm converges to the optimal path. Otherwise, the path r is set as the new p_D or p_Q according to whether r is unfeasible or feasible, respectively. These sub-steps of Step 3 are repeated while checking the validity of the available resources until the algorithm converges to the optimal solution.

4.5.3 The ROSA Scheduled-routing algorithm

Algorithm 4.3 solves the unicast QoS routing problem in (4.7), which is a bandwidth-constrained least cost (energy) problem, as well as problems (4.2) and (4.3). Similar to the *DetNet-routing* algorithm, it takes as input, from Algorithm 4.1, the current state of the network and a *Scheduled* flow. The objective of the *Scheduled-routing* algorithm is to reduce the overall network energy consumption by routing *Scheduled* flows through activated links, which are characterized by low energy metrics compared to the non-allocated links whose activation creates more energy. Hence, this energy-based scheme selects paths with minimum energy while guaranteeing an efficient bandwidth allocation by fitting the requirement of flow f. The required rate of flow f, R_f^{min}, presents the threshold that has to be ensured throughout the selected path.

The first step of this algorithm is to find the shortest path p_E minimizing the cost in terms of energy, as shown in line 1. Then, it checks whether the computed path's available bandwidth fits the flow's requirements and whether its constrained resources are sufficient. If both conditions (lines 2–3) are satisfied, then an optimal solution is achieved. If the resources are not satisfied, Algorithm 4.3 deletes the current solution p_E from the set of paths P and returns to Step 1 to compute a new path p_E that minimizes the energy while rechecking the same conditions. If the required rate is not satisfied, Algorithm 4.3 moves to Step 2 (line 7) for computing the shortest path using the bandwidth utilization ratio, i.e., the $W(e)$ metric.

In Step 2, it should be noted that calculating the shortest path using the Dijkstra algorithm based on bandwidth requires some adaptations compared to the standard cost-based Dijkstra algorithm. In fact, problems such as bandwidth utilization and congestion need to be treated differently by inspecting the utilization of each link in the path separately. For example, in Figure 4.3a, *Path 1* is selected compared to *Path 2*, using the standard Dijkstra approach, since it has the minimum sum-of-links utilization ratio (i.e., $W(p_1) = 130\%$), despite its second link being congested at 80% utilization of the link's capacity. On the other hand, updating Dijkstra while excluding highly congested links enables selecting routes with maximum available bandwidth, as shown in

Algorithm 4.3 The ROSA Scheduled-routing algorithm

```
Input: Network $H(V, L)$, $flag \leftarrow false$
Input: A Scheduled flow $f$ from the set of flows $F$.
Output: The optimal route.
Scheduled Routing ($H$, $s$, $t$, $R_f^{min}$)
1:  Step 1:  $p_E \leftarrow Dijkstra(s, t, E)$
2:  if $R_f^{min} \leq C_f(p_E)$ then
3:    if $\mathcal{Y} \leq C_f^{res}(p_E)$ and $Z_f(p_E) \leq Z^{max}$ then
          $\triangleright$ Constrained resources are validated.
4:        return $p_E$  $\triangleright$ Optimal solution.
5:    else Delete current [$p_E$] from $P$;
          $\triangleright$ Go to Step 1.
6:  else $\triangleright$ Go to Step 2.
7:  Step 2:  $p_W \leftarrow Dijkstra - Bandwidth(s, t, W)$
8:  if $R_f^{min} \geq C_f(p_W)$ then
9:     return "no feasible solution."  $\triangleright$ Violated flow.
10: else $\triangleright$ Go to Step 3.
11: Step 3:  while ($flag == false$)
12:   $\lambda \leftarrow \frac{E_f(p_E) - E_f(p_W)}{C_f(p_W) - C_f(p_E)}$
13:   $r \leftarrow Dijkstra(s, t, G_\lambda)$
14:   if $G_\lambda(r) = G_\lambda(p_E) = G_\lambda(p_W)$ then
        $\triangleright$ Convergence to feasible solution.
15:     if $\mathcal{Y} \leq C_f^{res}(p_W)$ and $Z_f(p_W) \leq Z^{max}$ then
          $\triangleright$ Constrained resources are validated.
16:       $flag \leftarrow True$;  return $p_W$
              $\triangleright$ Optimal solution.
17:     else Delete current [$p_W$] from $P$;
          $\triangleright$ Go to Step 2.
18:   else if $R_f^{min} \leq C_f(p_W)$ then $p_W \leftarrow r$
19:   else $p_E \leftarrow r$
End of Scheduled Routing
```

Figure 4.3b, where the selected *Path 2* does not include congested links, its available capacity $C(p)$ is equal to 50%, and so enables data transfer with less packet loss.

Hence, with Step 2, if the computed path does not meet the required rate R_f^{min}, flow f is violated, since there is no feasible solution within the available network resources and all the remaining

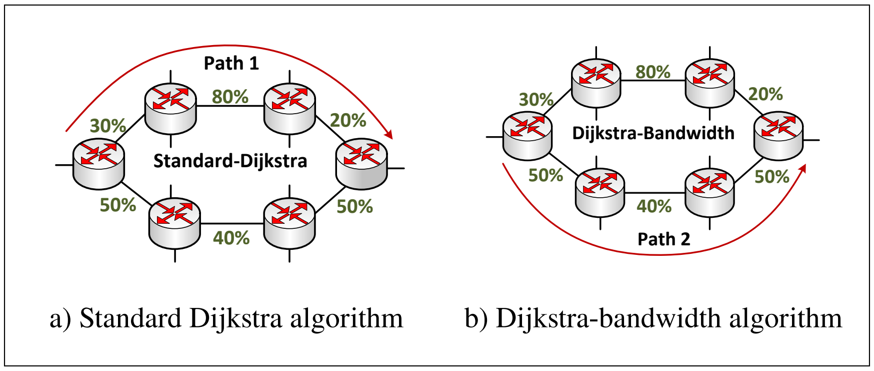


Figure 4.3 Path selection via bandwidth utilization ratios

routes in the topology have more utilization than the computed path. Otherwise, Algorithm 4.3 stores path p_W as a feasible solution with minimum bandwidth utilization but not as an optimal one, since it does not reduce the energy consumption, and consequently moves to Step 3. With the calculated p_E and p_W, the *Scheduled-routing* algorithm computes the shortest path r using the Lagrange dual function (Xiao *et al.*, 2016), where the Lagrange multiplier λ is defined as shown in (4.11a), and the cost function G_λ of each link $e \in L$ is determined by (4.11b) as the aggregated sum of the energy and the residual capacity.

$$\lambda = \frac{E(p_E) - E(p_W)}{C(p_W) - C(p_E)}. \tag{4.11a}$$

$$G_\lambda(e) = E(e) + \lambda \times C^{res}(e), \forall\ e \in L. \tag{4.11b}$$

Step 3 is repeated until the condition $G_\lambda(r) = G_\lambda(p_W) = G_\lambda(p_E)$ is satisfied, where the optimal route is found. Thus, Algorithm 4.3 checks the validity of the available resources and returns the path to Algorithm 4.1, which installs engineering-rules in the appropriate forwarding devices for routing flow f, updates the allocated resources' energy metrics if required, and moves to the management of the next flow. As a result of this energy-based flow aggregation strategy, the ROSA *Scheduled-routing* algorithm enables to reduce the energy consumption of the IIoT-based network while ensuring efficient bandwidth allocation for each flow without producing congestion and affecting running flows that are sharing common resources.

Table 4.2 Simulation parameters

Parameter/Value											
System		**SDN technology**		**Topology**		**Traffic characteristics**		**Bechmark (cost)**		**Controller configuration**	
OS	Ubuntu 16.04	**Floodlight**	Ver 1.2	**Switch**	22	**Delay**	1 - 150 ms	**ERRS**	Energy	**Updating time**	2 sec
RAM	16 GB	**Mininet**	Ver 2.2	**Links**	64	**Loss**	0 - 20%	**SWAY**	Delay & Loss	**Samples**	5
CPU	Core i7	**OpenFlow**	Ver 1.3	**Bandwidth**	10MB	**Rate**	20 bps - 250 Kbps	**LARAC**	Energy	**Threshold**	10%

4.6 Performance evaluation

4.6.1 Simulation environment

Table 4.2 provides the main simulation parameters. All the simulations were carried out on a PC with an Intel Core i7 CPU @ 3.4 GHz with 16 GB memory. To build the experimental environment, we use Floodlight 1.2 (Floodlight, Accessed: October 20, 2020) as the SDN controller and Mininet 2.2 (Lantz, Accessed: October 20, 2020) with OpenFlow 1.3 (OpenFlow Switch, 2015) to emulate a realistic SDN-based virtual architecture. Figure 4.4 illustrates part of the network topology adopted in this work, which consists of 64 links and 22 nodes. The channel bandwidth is set to 10 MB/s. Different traffic flows are injected using D-ITG traffic generator (Botta *et al.*, 2012) into the network with different requirements according to the *DetNet* and the *Scheduled* classes.

As baselines, we utilize the SWAY scheme (Saha *et al.*, 2018), the ERRS mechanism (Long *et al.*, 2018), and the LARAC algorithm (Juttner *et al.*, 2001) to reveal the enhancement of the proposed ROSA scheme. As explained in Section 4.2, on related works, the SWAY scheme employs two different strategies to manage the loss and delay-sensitive flows on the basis of the Yens' K-shortest paths algorithm (Yen, Accessed: October 20, 2020), while ERRS and LARAC schemes forward the traffic on the shortest energy paths while satisfying the delay requirements.

To collect network statistics and install rules for flow forwarding, we use the Floodlight *REST-API*. We employ the Floodlight entry pusher API to configure flow rules into network devices and the Floodlight statistics collection module to monitor the network's state (Controller, Accessed: October 20, 2020). The efficiency of service assurance and resource allocation depends critically

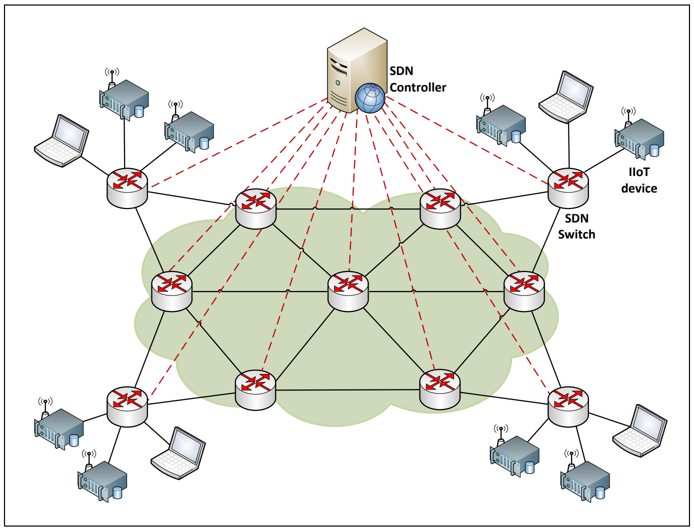


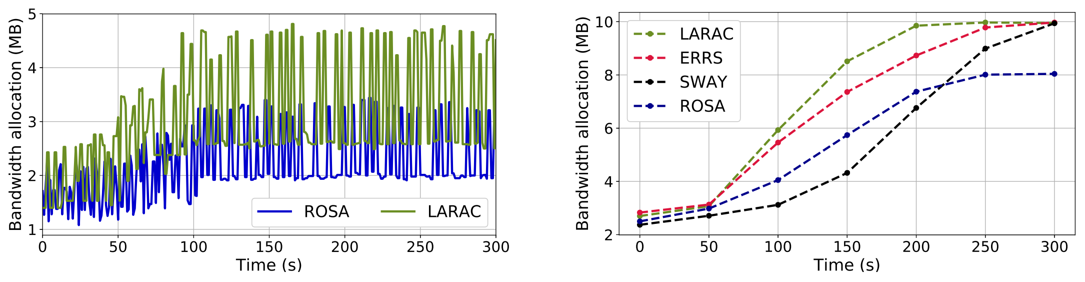
Figure 4.4 Example of a simulation topology

on the effectiveness of the topology statistics collector module and the accuracy of the collected information. Floodlight adopts the Link Layer Discovery Protocol (LLDP) to discover and update the entire network topology. The interval at which statistics are collected is set by default to 10 seconds. To avoid not getting network values updated in near real time, we reduce that value to 2 seconds. Hence, the topology state is updated frequently every 2 seconds, which provides more up-to-date values. Furthermore, each link-state update depends on two parameters: a history window and a threshold. By default, Floodlight takes a history of 10 LLDP samples. Then, it computes the average of this history. The state will be updated only if the history's average fluctuates 50% from the current state. Therefore, we reduce the history window to 5 samples, so that each sample will be more significant to the average. Furthermore, we reduce the threshold to 10%, so that the controller will be more sensitive to changes.

4.6.2 Simulation results

We investigate different scenarios within a test period of 300 seconds and with various numbers of flows. We evaluate the performance of the proposed ROSA scheme in terms of bandwidth allocation optimization and service assurance in relation to delay and packet loss. Then, we analyze the percentages of flow violation and network energy consumption.

Bandwidth allocation: Figure 4.5 shows the bandwidth allocation evaluation during the test period of 300 seconds. In our scenarios, we start by generating the *DetNet* flows. Then, from 50 seconds, we also generate the *Scheduled* flows where the loads increase significantly. In Figure 4.5a, we depict examples of bandwidth allocation variations through selected paths to validate the system functionality. We take into consideration only the ROSA and LARAC schemes to provide clear graph visualization. We generate corresponding flows for the two schemes without achieving the bandwidth limitation level (10MB). From this figure, it is obvious how the ROSA algorithm selects paths with low bandwidth utilization (optimized by 7% on average) compared to the LARAC algorithm as a result of using different routing approaches within a parallel management strategy. In the remaining scenarios, we inject heterogeneous flows from the *DetNet* and *Scheduled* classes increasingly while surpassing the bandwidth limitation level during the time of measurement. Figure 4.5b provides the average bandwidth allocation for all the schemes. At higher traffic loads, the proposed ROSA scheme achieves minimum bandwidth utilization compared to the benchmark mechanisms. The LARAC and ERRS algorithms have, respectively, 22% and 16% higher bandwidth utilization compared to the ROSA scheme, since they are based on only one technique to manage all network flows. The SWAY algorithm achieves less bandwidth utilization at low traffic loads than ROSA because it distributes the flows' rates through different paths using different techniques for managing loss-sensitive and delay-sensitive flows. Though ROSA also uses different approaches to manage heterogeneous network flows, it gathers the *Scheduled* flows into the activated links to reduce the network energy consumption. This flow aggregation leads to the augmentation of the bandwidth utilization of ROSA compared to SWAY. Indeed, gathering flows' rates at lower loads does not negatively affect service quality but reduces network energy consumption absolutely. Finally, ROSA reduces bandwidth utilization at higher loads by 14% compared to the SWAY scheme because of the ROSA bandwidth-constrained allocation strategy (i.e., $\mathcal{Y}$ in (4.3)), which controls rate allocation and flow aggregation at congestion levels. In practice, a link is expected to be congested if its bandwidth utilization surpasses 75% - 85% (Egilmez *et al.*, 2013). In our experiments, we consider that a link is congested if 80% of its bandwidth is utilized. Thus, we set $\mathcal{Y}$ to 20%.



a) Variation of bandwidth allocation b) Average of bandwidth allocation

Figure 4.5 Evaluation of bandwidth allocation

End-to-end delay: In Figure 4.6a, we evaluate the average end-to-end delay. As shown in this figure, ERRS and LARAC mechanisms provide very close results throughout the experiments because they target the same metrics, i.e., minimizing the energy while satisfying delay requirements. They have, respectively, 18% and 21% higher end-to-end delay compared to the proposed ROSA scheme. According to the SWAY algorithm, up to 200 seconds of the measurement time (i.e., at low traffic loads), it achieves minimum end-to-end delay in consequence of its distributed rate allocation approach, which leads to low latency but high energy consumption. However, at low traffic loads, ROSA aggregates *Scheduled* flows over activated links, since such gatherings do not affect the quality of service but reduce the overall network energy consumption. On the other side, with the increasing number of flows at higher loads, the SWAY end-to-end delay results are reversed; here, ROSA outperforms SWAY with 7% improvement because of the deployment of the ROSA bandwidth-constrained allocation strategy, which controls network congestion levels.

Packet loss: As already mentioned, packet loss is one of the critical factors to ensure reliability for industrial communications. In Figure 4.6b, we evaluate the effect of increasing the number of flow allocations and loads on the packet-loss ratio during the time of measurement. As shown in this figure, the proposed ROSA scheme outperforms the existing benchmark because of its parallel route mechanism, its packet-loss constraint for each *DetNet* flow, its efficient bandwidth allocation technique for each *Scheduled* flow, and the bandwidth-constrained allocation strategy

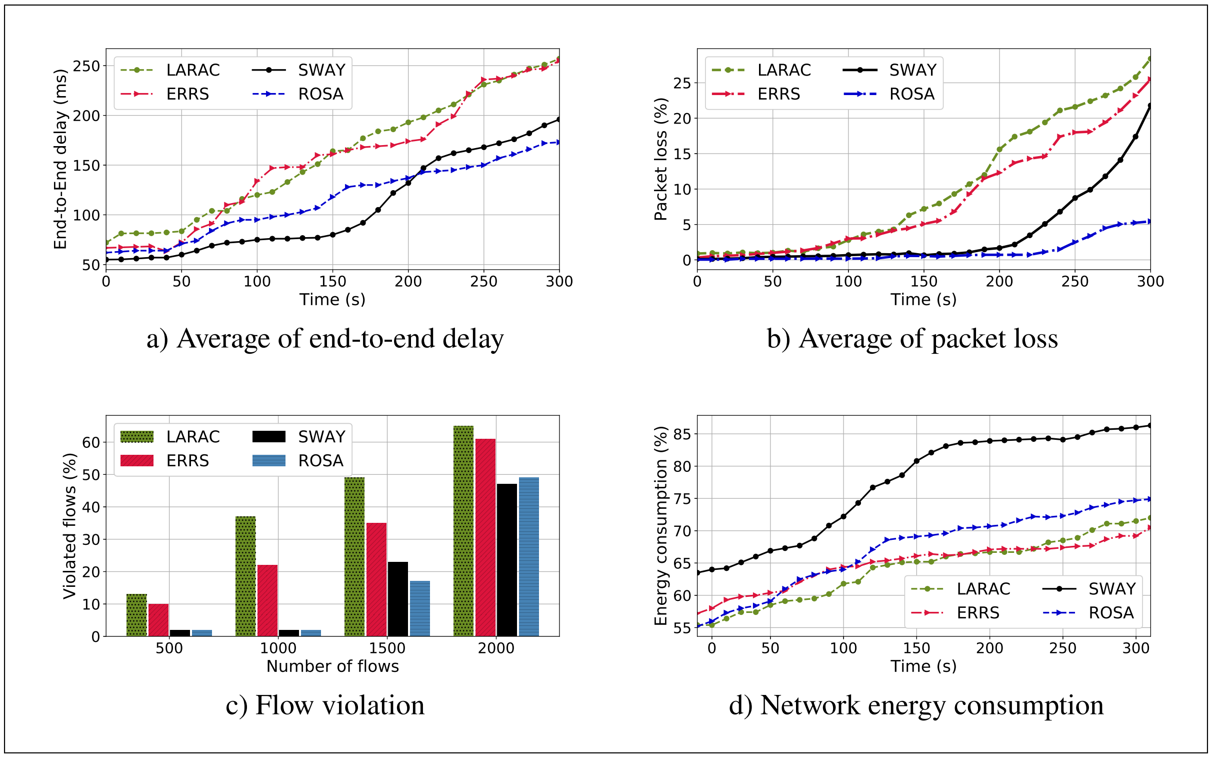


Figure 4.6 Network performance based on (a) End-to-end delay; (b) Packet loss; (c) Flow violation; and (d) Energy consumption

at the network congestion levels. The ROSA scheme reduces the packet-loss ratio by more than 19%, 14%, and 8% on average compared to LARAC, ERRS, and SWAY, respectively.

Flow violation: When network loads achieve the bandwidth limitation level, routing schemes reject new flows. Figure 4.6c illustrates the percentage of violation according to the number of flows. While ensuring a reliable transfer for the existing services in the network paths, ROSA achieves a significant percentage of fulfilled flows compared to the benchmark schemes. The proposed solution achieves 16%, 14%, and 7% reductions in flow violation compared to the LARAC, ERRS, and SWAY mechanisms, respectively. It is noteworthy that with 2000 flows, SWAY provides a slight reduction comparing to ROSA since our scheme deploys the efficient bandwidth-constrained allocation method (i.e., $\mathcal{Y}$), which rejects flows at network congestion levels to ensure reliability for the existing services in the network. As mentioned previously, in our experiments, we set $\mathcal{Y}$ to 20%.

Energy consumption: Figure 4.6d shows the percentage of energy consumption in the network. As depicted in this figure, as a result of the ROSA *Scheduled-routing* algorithm, our scheme enables to save a significant amount of energy in comparison to SWAY. The ROSA approach achieves a 14% reduction in energy consumption compared to SWAY. The ERRS and LARAC algorithms provide the lowest percentages of energy consumption with 6% and 4% reductions, respectively, compared to the ROSA scheme. However, although ERRS and LARAC achieve the best energy consumption, they have the worst bandwidth allocation and service assurance, as shown in Figures 4.5 to 4.6c. ERRS and LARAC target only one objective (i.e., minimizing the network energy) without considering service characteristics (e.g., bandwidth allocation, packet-loss, or a combination of any of the QoS traffic types) to ensure reliability, which represents one of the most critical factors in industrial networks. With the ROSA scheme, we ensure service requirements for the *DetNet flows*; meanwhile, with the *Scheduled flows*, we minimize the energy consumption while assuring an efficient bandwidth allocation.

In all, the proposed ROSA solution outperforms the conventional methods in terms of trade-off between bandwidth allocation, end-to-end delay, packet-loss, and energy efficiency. All the existing mechanisms focus on restrictive network criteria (e.g., energy parameter with the ERRS scheme, packet-loss and delay with the SWAY scheme, etc.), which degrade either service or network performance. The proposed ROSA parallel-based strategy, which includes delay, packet-loss, and bandwidth-constrained routing approaches, while also being aware of the energy consumption, presents promising effects on resource optimization, service assurance, and overall network energy consumption.

4.7 Conclusion and future directions

In this paper, we aim to respond to supporting the digital transformation of the industrial environments from traditional forms to intelligent, connected, and low-carbon and sustainable industrial strategies. We address the issue of multi-constrained QoS provisioning with efficient resource utilization and energy consumption in IIoT-based networks, the related research on which, to the best of authors' knowledge, has still not been sufficiently explored. We present a centralized route optimization and service assurance scheme over a multi-layer programmable industrial architecture. We prove that SDN is a key enabling technology for enhancing IIoT network management by offering programmability and flexible dynamic reconfigurability. We demonstrate how service characterization plays a crucial role in appropriate algorithm design and network performance. Thus, we classify heterogeneous industrial flows into *DetNet* traffic (e.g., URLLC flows) and *Scheduled* traffic (e.g., mMTC flows). We introduce a good trade-off between network service assurance, resource optimization, and energy consumption reduction; to do so, we propose a parallel-based routing platform that runs different algorithms to compute paths according to the flow's type, the constrained available resources, and the network energy consumption. The proposed algorithms are based on the Lagrangian Relaxation approach to solve different multi-constrained shortest path problems that are NP-hard by nature. The obtained results confirm that the proposed ROSA scheme has significantly improved the total performance of IIoT network in terms of service assurance and reliability, resource utilization optimization, and energy consumption reduction.

As future works, our SDN-based parallel-routing approach will be extended with additional machine learning-based routing algorithms, such as the backup algorithm that computes redundant paths to ensure systems' availability in case of failures. Moreover, we will focus on the efficiency and scalability of the control plane with the deployment of the cluster-based distributed controller technology, since this issue has significant impacts on network performance with heterogeneous data streams and massive devices. Finally, ROSA' s efficiency will be tested using industrial protocols in a real smart factory in order to address practical concerns.

CHAPTER 5

SERVICE AND RESOURCE AWARE FLOW MANAGEMENT SCHEME FOR AN SDN-BASED SMART DIGITAL CAMPUS ENVIRONMENT

Yosra Njah [a] , Chuan Pham [a] , Mohamed Cheriet [a]

[a] Department of Automated Production Engineering, École de Technologie Supérieure (ÉTS), 1100 Notre-Dame Ouest, Montréal, Québec, Canada H3C 1K3

Published in IEEE Access, June 2020

Abstract

Recently, campuses have been embracing smart digital technologies in order to boost the efficiency of education and creativity. Thus, massive heterogeneous flows are generated as a result of multitude simultaneous access from several heterogeneous devices. This is putting pressure on campuses to make better management of their constrained resources and to ensure the required Quality of Service (QoS). In this paper, we propose a multi-flow management scheme over a software-defined smart digital campus network, named Service and Resource Aware Flow Management (SRAFM). Our approach offers a unified fully-programmable architecture, a distributed end-host-based flow characterization plane, and a centralized software-defined optimization model to efficiently manage heterogeneous flows. Network functionalities, including QoS aware routing and resource allocation optimization, are formulated as a mixed-integer linear programming problem. Due to its NP-hard complexity, we propose an approximation algorithm in a decomposed fashion based on Lagrangian Dual Decomposition (LDD) and subgradient methods to find an optimal solution for flow management. We evaluate our scheme from different aspects, including the number of simultaneous heterogeneous flows, QoS provisioning, characterization impacts, and network scalability. Compared to the well-known benchmarks in QoS aware routing and optimization problem, SWAY and LARAC, our simulation results conducted with a large number of flows over a small-scale network show promising performance. The proposed scheme significantly improves the cost reduction by 51% as compared to LARAC, the end-to-end delay by 21% and 34%, the bandwidth availability by 27% and 36%, and the QoS violation by 11% and 29% as compared to SWAY and LARAC, respectively.

Keywords: Smart digital campus, Internet of Things (IoT), Software-Defined Networking (SDN), flow characterization, Quality of Service (QoS), distributed rate allocation, and resource optimization.

5.1 Introduction

Campuses are now embracing smart digital technologies (e.g., Industrial Internet, Internet of Things, and Smart Cities) to create intelligent, green, and safe educational environments. Staff and students are empowered with smart services, which boost efficiency for learning, collaborating, creating, and sharing. Many research studies (Sutjarittham *et al.*, 2019; Yang *et al.*, 2018; Sivanathan *et al.*, 2017; Uskov *et al.*, 2015, 2016) and industrial companies, such as Cisco (Cisco, Accessed: July 12, 2019), Campus Management Corp (Campus-Management, Accessed: July 7, 2019), Deloitte (Deloitte, Accessed: July 11, 2019), and Ruckus (Ruckus, Accessed: July 16, 2019) have revolutionized how to design, build and manage smart campus networks to move towards digital education.

The smart digital campus, or modern campus, is equipped with thousands of heterogeneous Internet of Thing (IoT) and non-IoT devices that autonomously interact with each other to unleash or use a massive number of services appropriate for smart living and learning applications. With this unlimited development of heterogeneous devices and services, the campus network size keeps scaling up, while massive heterogeneous traffic flows continue to grow inexorably. Accordingly, network resources and bandwidth acquire an unprecedented demand. For example, Sivanathan *et al.* (2017) showed that when there is an activity from IoT devices and non-IoT devices, the campus network load peaks at around 17 Mbps. Meanwhile, the average load is 400 Kbps with IoT devices. Another study produced by Cisco (Cisco, Accessed: April 22, 2019) shows that some types of network applications, like video-based applications supplied by providers such as Youtube, Netflix, and Hulu, will grow at a compound annual growth rate

(CAGR) of 31%, while online gaming traffic will have a traffic growth rate of 47%, and traffic including web, email, and data will have a CAGR of 18%.

These services' massive data streams are leading to unprecedented challenges for network administrators in terms of advanced solutions development for network flow management and resource allocation control with minimum cost, especially with the constrained campus network resource problems (e.g., limited link capacity, constrained device with limited CPU, memory, and power resources) (Ersue *et al.*, 2015; Bormann *et al.*, 2014; Saha *et al.*, 2018). Furthermore, these heterogeneous services require various specific QoS requirements. For instance, certain mission-critical flows generated during some emergency/urgent periods, such as gas monitors, smoke sensors, and disaster sensors, require transference of data in real-time. Some other applications, such as peer-to-peer file sharing, software updates, and cloud-based file storage systems, are not time sensitive but very bandwidth-hungry applications that might occupy all available bandwidth in the case of inefficient management. Especially, certain types of services, such as online learning and video conferencing, require bandwidth and real-time guarantees to run without degraded performance.

In fact, network services may violate QoS levels because of four main network factors: low bandwidth, high latency, WiFi signal interference, and overloaded constrained device. If all network equipment are working properly, then bandwidth and latency are the two likely reasons (Saunders *et al.*, 2012). In addition, even though a campus network has been designed with highly adequate resource bandwidth, but without efficient engineering mechanisms for service and resource management, critical flows may compete with all kinds of traffic, including bandwidth-hungry flows; consequently this causes serious flow QoS violations in the network. Moreover, the traditional campus network architecture has limited global state visibility, i.e., it lacks a global view of the available network resources and the overall network architecture, since each router performs a hop-by-hop routing using its coupled control and data planes (Wang, 2001). This makes traditional network architecture hard and time-consuming to configure devices and manage resources and traffic flows.

Software-defined networking (SDN) has emerged as an efficient network management paradigm to overcome these issues by decoupling the data and control planes. The network control functionalities are further away from network devices and centralized into a logically centralized point, called the control plane (the brain of the network). In other words, network decisions are made by the control plane with a global network view and use different programs in the application plane to optimize network resource management. The forwarding function is performed according to engineering policies programmed and configured by the control plane into network devices that are converted into simple high-speed forwarding elements. This emerging paradigm is particularly attractive for addressing many network optimization problems, such as dynamic flow control, flexible network resource management, and QoS provisioning (Macedo *et al.*, 2015; Xia *et al.*, 2015; Kreutz *et al.*, 2015; Shu *et al.*, 2016).

The above-mentioned issues and the adoption of software-defined based implementation motivate us to focus on the following questions: *How can an efficient fully-programmable SDN-based solution for the smart digital campus be designed? How can the different requirements of each network traffic flow within a set of hundreds, even thousands of flows, be addressed? What are the network policies that proactively automate the management process of a set of heterogeneous flows while fulfilling the required QoS, improving available resources, and minimizing the routing cost?*

To answer these questions, in this paper, we propose a multi-flow management scheme over a software-defined smart digital campus network named Service and Resource Aware Flow Management (SRAFM). The proposed scheme is different from existing works for two reasons. First, it offers a whole solution in terms of architecture, flow characterization, and QoS aware routing and resource optimization to manage the masses of heterogeneous flows generated from thousands of interconnected devices. Thus, over a unified fully-programmable architecture, SRAFM implements a distributed end-host-based flow characterization solution and a centralized software-defined optimization model for service and resource-aware routing problem. Second, in terms of QoS aware routing and resource optimization problem, SRAFM provides an optimal solution for an entire set of simultaneous flows in the network, which is more challenging than most prior schemes that often address this issue with a flow-per-flow strategy. Furthermore, regarding the network in terms of flow-per-flow management cannot guarantee a global optimal allocation in the network; in some special cases, it may reduce the network performance by wrong routing decisions in advance.

The major contributions of this work can be summarized as follows:

1. **Design of a smart campus programmable architecture for flexible flow characterization and management:** We discuss the design challenges to deploy a unified fully-programmable architecture that controls wired and wireless software defined-based smart campus networks. Since flow characterization presents a fundamental network functionality to support QoS aware routing and optimization, we provide an overview of the different types of services and their QoS requirements on the basis of delay and bandwidth characterization. Accordingly, we identify the design challenges confronting the characterization of a large number of heterogeneous flows in a fully-programmable architecture, and we propose a distributed end-host-based plane for flow characterization based on a combined approach of device and service identifications. While designing this specific network-level in the architecture, we take into consideration the end-user's privacy, the processing time, the controller overhead, and the network bandwidth consumption.
2. **QoS aware routing and optimization:** SRAFM controls not only delay-sensitive flows, as do prior works, but all smart campus network flows to ensure service QoS requirements, to control bandwidth-hungry services, and to optimize network resource allocation. We propose a centralized optimization framework that manages flows based on an SDN proactive and reactive strategy and optimizes the system cost in terms of joining resource cost and path loss. Since the formulated multi-constraints optimization problem is NP-hard, the Log-det approximation function (Fazel *et al.*, 2003) is used to relax the problem. Then, by using the Lagrangian Dual Decomposition approach, we decompose the relaxed problem into per-flow sub-problems that can be solved simultaneously in a decomposed fashion. Accordingly, the SRAFM routing strategy finds an optimal solution not only for each flow

independently but for the whole set of flows while performing coordination between their various requirements and the available resources.

3. **Simulation results:** We evaluate the proposed scheme through various simulation aspects, including the number of simultaneous heterogeneous flows, QoS requirements, characterization impacts, and network scalability. We compare our scheme to the SWAY (Saha *et al.*, 2018) and Lagrange Relaxation based Aggregated Cost (LARAC) (Juttner *et al.*, 2001) algorithms, the well-known benchmarks in QoS aware routing and optimization problem. Our simulation results conducted with a large number of flows over a small-scale network show promising performance. Thus, SRAFM achieves 51% in terms of cost reduction as compared to LARAC. In addition, it improves the end-to-end delay by 21% and 34%, the bandwidth availability by 27% and 36%, and the QoS violation by 11% and 29%, as compared to SWAY and LARAC, respectively. To evaluate the system cost, we consider only LARAC, since SWAY adopts two different cost functions with different metrics in the same network.

The remainder of this paper is organized as follows. Section 5.2 reviews related relevant works from the perspective of QoS aware routing and optimization over a software-defined-based environment. Section 5.3 discusses the software-defined smart campus network, the unified fully-programmable architecture, and the strategies to deploy flexible multi-flow characterization and optimization solutions. Sections 5.4 and 5.5 respectively present the optimization problem and the proposed QoS-aware routing algorithms. The SRAFM operational scenario and the analysis of the results are presented in Sections 5.6 and 5.7, respectively. Finally, we conclude and discuss future work in Section 5.8.

5.2 Related work

In the context of smart digital campus networks, many research studies have been proposed to design and build smart digital campus networks with the appropriate technologies (e.g., traffic profiling, prediction of student attendance, etc.) to move towards digital education (Sutjarittham *et al.*, 2019; Yang *et al.*, 2018; Sivanathan *et al.*, 2017; Uskov *et al.*, 2015, 2016). However, none of these works target service and resource-aware traffic management and optimization. On the other hand, in the context of QoS provisioning and resource optimization over smart networks, the body of literature is vast (Karakus & Durresi, 2017), and it covers different SDN-based networks, such as smart home networks (Gharakheili *et al.*, 2016; Habibi Gharakheili *et al.*, 2017) and industrial networks (Guck *et al.*, 2017; Henneke *et al.*, 2016). However, the management of a large number of heterogeneous flows to improve service and network performance over smart digital campus networks is still an open issue with many challenges.

While few research works deal with QoS provisioning and resource optimization over SDN-based smart digital campus networks, industrial companies, such as Cisco (Cisco, Accessed: July 12, 2019), Campus Management Corp (Campus-Management, Accessed: July 7, 2019), Ruckus (Ruckus, Accessed: July 16, 2019), and Huawei (Huawei, Accessed: December 19, 2019) have revolutionized how to design, build and manage smart campus networks to move towards digital education. Thus, the remainder of this section reviews some relevant related works from the perspective of QoS-aware routing and optimization problems in SDN-based environments, particularly the most common aspect, i.e., the delay-constrained least-cost (DCLC) routing problem.

Egilmez *et al.* (2012, 2013) propose an optimization model to ensure end-to-end multi-level QoS for video streaming service over SDN-based networks. They treat the base layer of video bit streams as a level-1 QoS flow, while packets of enhancement layers are treated as level-2 QoS or as best-effort flows. The rest of the network traffic is also managed as best-effort flows. They pose optimization QoS routing as a constrained shortest path problem in which delay and packet-loss are considered as QoS requirements, and they use the LARAC scheme (Juttner *et al.*, 2001) as a QoS routing algorithm. Yu *et al.* (2015) propose a QoS routing scheme for video streaming traffic over SDN networks. Similar to the previous works, Yu *et al.* (2015) also treat the base layer and enhancement layer of video bit streams separately as two levels of QoS flows. However, the proposed routing solution is based mainly on the shortest path algorithm to route the base layer packets (level-1 QoS flows), if it meets the delay variation constraints. Otherwise, a QoS routing algorithm is invoked to select the required path. Despite the fact that

(Egilmez *et al.*, 2012, 2013; Yu *et al.*, 2015) cover several critical issues in terms of multi-level QoS, they do not consider different types of network services. In other words, their proposed schemes address problems for a specific service, i.e., video streaming, which might not fit the large number of heterogeneous services in smart networks.

Guck *et al.* (2018) provide a comprehensive survey of QoS routing algorithms in SDN-based networks. They implemented 26 DCLC algorithms and compared their run-time and cost efficiency within a four-dimensional (4D) evaluation framework. The four dimensions correspond to the type of topology, two forms of scalability of topology, and the tightness of the delay constraint. They conclude with the outperformance of two routing algorithms in the vast majority of the evaluations, namely, LARAC (Juttner *et al.*, 2001) and Search Space Reduction Delay-Cost-Constrained Routing (SSR+DCCR) (Guo & Matta, 2003). All the evaluated algorithms, in (Guck *et al.*, 2018), are devised to deal with single metric routing schemes (i.e., delay). However, a multi-metric QoS routing optimization should be considered to fulfill heterogeneous service requirements. In other words, the QoS provisioning problem should not take into consideration only time sensitive traffic in which the delay is a highly critical parameter, but also small IoT flows and mission-critical data that need different QoS requirements. Furthermore, bandwidth-hungry flows should be under the control of specific engineering policies, because transferring such type of traffic using only the best-effort mechanism, as in (Egilmez *et al.*, 2012, 2013; Yu *et al.*, 2015), can create congestion and degrade the performance of critical applications when the same network resources are shared.

Saha *et al.* (2018) propose two different QoS routing strategies to address the issue of heterogeneous flows. One is devised to deal with delay-sensitive flows, and the other is devised to deal with loss-sensitive flows. Both of the deployed algorithms are based on the Yens K-shortest paths algorithm (Yen, Accessed: October 20, 2020), which is included in the comparison performed by (Guck *et al.*, 2018). However, because of the different deployed strategies to deal with the two classes of traffic, the authors consider two different cost functions. Thus, they minimize the delay metric for delay-sensitive traffic and the loss metric for loss-sensitive traffic, subject to different constraints. By contrast, we consider that every flow is sensitive to loss, and we propose an optimization problem that minimizes the operational cost of the selected path in addition to the loss-rate, according to the delay and bandwidth sensitivity metrics.

Finally, it is worth mentioning that the literature encloses a multitude of approaches that address the QoS-aware routing problem over the SDN-based environment, using different mechanisms such as machine learning (Lin *et al.*, 2016), node characterization (Beshley *et al.*, 2017), queue scheduling (Guck *et al.*, 2017), multi-path selection (Dutra *et al.*, 2017), etc. However, to the best of our knowledge, all these proposed works are based on a per-flow approach, i.e., using the current state of the network, the optimal routing solution is selected independently for each flow, and not for all flows in the system. Though this flow-by-flow technique can reach a fast routing decision, it cannot provide a global optimal solution for the whole set; in some cases, it violates service requirements and network performance due to previous decisions.

5.3 A smart digital campus network

In this work, we propose a multi-flow management scheme called Service and Resource Aware Flow Management (SRAFM) for the smart digital campus network. This proposed scheme requires being deployed in a flexible programmable architecture. In addition, the large number of services leads us to design a specific network-level in the architecture to analyze the nature of heterogeneous flows before performing traffic routing and resource optimization.

Thus, in this section, we propose the software-defined smart campus architecture, a unified fully-programmable architecture, which manages the heterogeneous wired and wireless network components. Then, we discuss the characteristics of the network services in terms of delay and bandwidth requirements, and we discuss the distributed end-host-based plane for flow characterization.

5.3.1 Software-defined campus architecture

As depicted in Figure 5.1, the software-defined campus network is based on a fully-programmable paradigm in which all the network devices in each layer (i.e., access, aggregation, core, and wireless backhaul layers) are controlled and programmed by the centralized control plane (Huawei, Accessed: December 19, 2019). This control plane can include one or multiple controllers to handle the increased management complexity of large-scale wired and wireless networks. Thus, it is possible to control and manage the wired network over the access, aggregation, and core layers with one SDN controller, or to slice network views in a way that each layer is managed via a different SDN controller. On the other hand, to manage the wireless backhaul layer, it is important to note that SDN has been designed for wired networks; but wireless networks have different requirements, and there is not yet a consensus or standard on how to program wireless forwarding elements (Macedo *et al.*, 2015). The two main challenges of software-defined wireless implementation are the configuration of the wireless forwarding elements by the control plane, and the interaction between the access layer and the programmable wireless backhaul forwarding elements. For these purposes, significant research studies and industrial implementations have been proposed as an extension of the SDN paradigm to incorporate mobile-specific functionalities. For example, Huawei (2015) presents Huawei's agile campus network solution, a fully-programmable architecture that includes an access controller and programmable agile switches enabling unified wired and wireless traffic forwarding. Nunez *et al.* (2016) propose featuring the wireless backhaul forwarding elements and the typical SDN controller with wireless agent extensions that enable the management of packets and forwarding rules in a technology-agnostic manner. Seppänen *et al.* (2014) propose a network abstraction approach by hiding the wireless network from the SDN layer. Thus, instead of controlling the wireless forwarding elements directly with the SDN controller, the whole wireless network is seen as a single SDN switch, controlled like a standard SDN device.

In our work, as shown in Figure 5.1, for the wireless backhaul layer management, a controller, named Software-Defined Radio (SDR) controller, is proposed to manage the data connections between the radio access elements (e.g., wireless access points), the Wireless Backhaul Forwarding Elements (WBFE, such as base stations), and the operators' SDN enabled devices in the network. All programmability functionalities in the wireless data plane (such as defining forwarding rules and radio resource management) are implemented using the SDR controller.

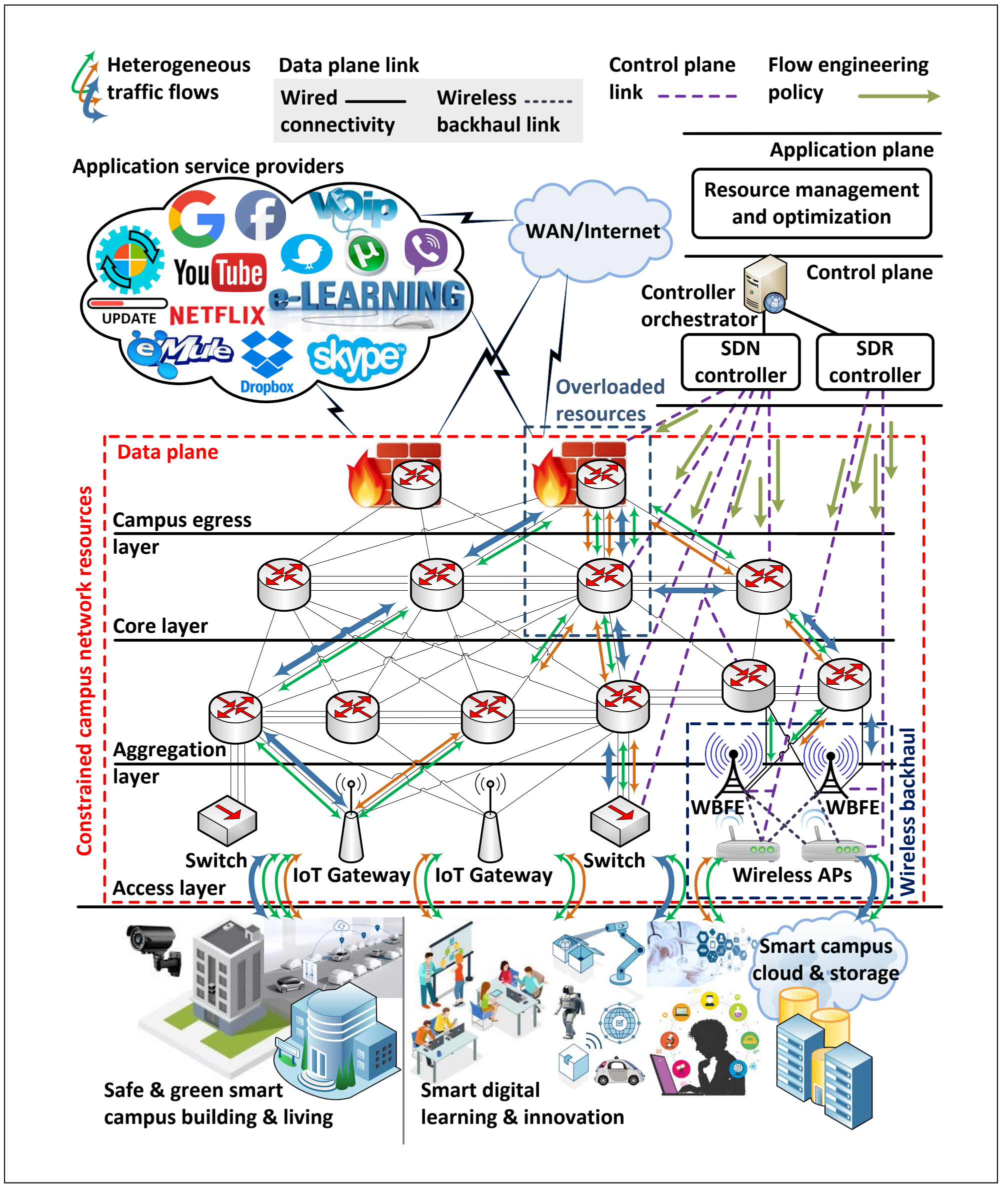


Figure 5.1 Software-defined smart campus network architecture

Then, a controller orchestrator is required to ensure the unified interaction of all the heterogeneous network technologies and operators, via the coordination between the different controllers, and the establishment of compatible configurations between the wired and wireless networks.

However, as detailed in (Macedo *et al.*, 2015), where Macedo *et al.* survey SDN, SDR, and network function virtualization (NFV) technologies, achieving unified management of wired and wireless programmable networks is certainly a big challenge, and it is fundamental that these technologies complement each other to develop a highly flexible programmable network. Thus, in our work, we focus on service management and resource optimization for the software-defined campus network, while assuming that the unified fully-programmable campus architecture is established. In other words, the technical interaction between the different controllers performed by the controller orchestrator and the required features over the wireless forwarding elements to ensure a unified programmable management process are beyond the scope of our current research project. We refer interested readers to (Shantharama *et al.*, 2018; Grandi *et al.*, 2018; Santos & Kassler, 2017; Ricardo, 2016) for more details about the design challenges of a unified fully-programmable architecture.

5.3.2 Heterogeneous flow characterization

5.3.2.1 Smart digital campus network traffic

As shown in Figure 5.1, over the smart digital campus network, different services related to two main axes, digital learning and smart campus environment, are generated from IoT and non-IoT devices (Cisco, Accessed: July 12, 2019). Table 5.1 presents a taxonomy and examples of such services.

On the one hand, smart digital learning and innovation services are generated from IoT and non-IoT devices to maximize the potential of learning and research. For example, courses and training are offered by top faculty and leaders from the same university or around the world, and they are accessible to students anywhere and at any time due to innovative learning facilities such as IoT-based classrooms, virtual classes, IoT sensors for note sharing, etc. (Campus-Management, Accessed: July 7, 2019; Chan & Chan, 2018).

On the other hand, smart campus building and living related services are generated mainly from IoT devices to offer a safe, green, and smart living environment. This axis includes certain security services, which report urgent alarm events via messages, high-resolution images, and

Table 5.1 Taxonomy of smart digital campus network services generated from IoT and non-IoT devices

Digital learning and innovation services	**Priority level**	**QoS Sensitivity to:**	
		Delay	**Bandwidth**
Video-based communications for online education, administrative, and financial services: e.g., online courses, conferences, training sessions, face-to-face meetings over distance, etc (Uskov *et al.*, 2015; Chan & Chan, 2018).	Highly critical	Real-time e.g., < 150 ms	High / medium rate
Smart library, courses, and administration's cloud-based resource transfer and storage: e.g., lectures and assignments, heavyweight software updates, etc (Chan & Chan, 2018).	Critical	Non real-time	High rate
Video-based services for students and staff during spare-time: e.g., playing online video games, watching videos from Youtube, Netflix, Hulu, Facebook, etc (Habibi Gharakheili *et al.*, 2017; Chan & Chan, 2018).	Non critical	Real-time e.g., < 250 ms	High / medium rate
Bulk transfer-based services: e.g., video downloads (for offline viewing), peer-to-peer file sharing, software updates, etc (Habibi Gharakheili *et al.*, 2017) (Chan & Chan, 2018; Sivanathan *et al.*, 2018).	Non critical	Non real-time	High rate
Safe & green smart campus building and living	**Priority Level**	**QoS Sensitivity to:**	
		Delay	**Bandwidth**
Video surveillance and security monitoring: e.g., behavior-based emergency services, audio panic alerts, security video surveillance, access control, etc (Saha *et al.*, 2018; Schulz *et al.*, 2017; Mocnej *et al.*, 2018).	Highly critical	Real-time e.g., < 150 ms	High / medium rate
Smart and green campus services: e.g., building automation services, connected lighting, energy and waste management, smart parking, etc (Saha *et al.*, 2018) (Schulz *et al.*, 2017; Mocnej *et al.*, 2018).	Critical	Near real-time e.g., < 30 s	Low rate

videos (Deloitte, Accessed: July 11, 2019). It also includes various intelligent applications, such as smart access, building automation systems, smart parking, and payment. Besides, it provides services, such as heating adaptation, light-adjustment, and water conservation, which aim to transform the traditional campus environment into a model of a green institution at low cost by reducing energy and carbon footprint (Ruckus, Accessed: July 16, 2019).

5.3.2.2 Services' QoS requirements characterization

Since the number of services generated from IoT and non-IoT devices is unlimited and network resources are constrained, we define a set of traffic classes with their appropriate QoS requirements and priority levels in the network.

As shown in Table 5.1, we characterize services' QoS requirements on the basis of delay and bandwidth sensitivities. Thus, we separate network campus services into two main sets: delay and bandwidth sensitive flows. We refer to these two sets, respectively, as F_{ds} flows and F_{bs} flows. Consequently, the whole set of campus network flows F is represented as follows:

$$F = F_{ds} \cup F_{bs}. \tag{5.1}$$

The delay-sensitive class includes video-based services and IoT services. All these services are characterized, firstly, by time constraints that have to be deterministically guaranteed. We define, within the F_{ds} set, three levels of prioritization. The first one (Highly critical / Real-time) includes ultra-high-definition video-based services related, for examples, to security video surveillance, on-line course learning, video conferencing, etc. As shown in Table 5.1, real-time and high bandwidth are both the main characteristics of these services to ensure an efficient end-to-end delivery without interruptions and packet loss (Sivanathan *et al.*, 2018; Chan & Chan, 2018; Chen *et al.*, 2004). All the flows of this level will be managed proactively by the SRAFM scheme since they are regular and frequent in the network. The second level (Critical / Near real-time) includes IoT services that are generated, for examples, by building automation systems, connected lighting, smart parking, etc. As shown in Table 5.1, these irregular and infrequent services are characterized by near real-time (e.g., tolerable delay of 30 s) and low-rate requirements (i.e., each IoT device exchanges a small amount of data per-flow) (Schulz *et al.*, 2017; Mocnej *et al.*, 2018). These smart IoT services require a critical priority level in the network, since competing with traditional flows (e.g., bulk transfers) can significantly affect the performance of these low-rate IoT applications (Saha *et al.*, 2018). Finally, the third level (Non critical / Real-time) includes non-critical video-based services, which are generated, for instances, by students playing online games and/or watching videos supplied by providers such as Youtube and Netflix (Habibi Gharakheili *et al.*, 2017). However, the flows of this level require a strict end-to-end delay; for example, an online game requires less than 250 ms to run smoothly (Cacheda *et al.*, 2007).

The F_{bs} set includes services that utilize large amounts of bandwidth and place enormous strain on the network. As shown in Table 5.1, it includes a significant number of applications, such as peer-to-peer file sharing, large downloads, and software updates, which are very bandwidth-hungry services but not sensitive to delay. This set of flows is also called bulk transfers. The non-controlled management of these bandwidth-hungry applications creates network congestion and leads to performance degradation of the delay-sensitive flows, i.e., the F_{ds} set. Consequently, these services should be managed through the network with a lower priority compared to the delay-sensitive services. We also categorize this class into two different levels. The first level (Critical / Non real-time) includes mission-critical data such as electronic books management, courses and administration's cloud-based resources transfer and storage, etc. The second level (Non-critical / Non real-time) involves non critical data, such as students' video downloads for offline viewing.

5.3.2.3 Distributed end-host-based flow characterization plane

Characterizing network flows with the appropriate performance levels and QoS requirements, as shown in Table 5.1, should be performed before flow management, since this impacts greatly routing decisions and resource allocation optimization. In this section, we discuss the services' massive data stream processing, particularly the challenges of the joint design of multi-flow characterization and management over a fully-programmable architecture, and we propose the distributed end-host-based plane for flow characterization.

With the programmable paradigm, multi-flow processing could be performed within either a centralized approach or a distributed approach. According to the centralized approach, since there is a lack of intelligence in the forwarding plane, thousands of heterogeneous flows are sent by forwarding devices to the centralized controller to be analyzed using flow characterization programs. Then, as illustrated in Figure 5.2(a), they are mapped to specific engineering policies, which are computed using traffic engineering and management programs. The centralized controller installs these computed rules over the programmable forwarding devices to enable the transfer of the corresponding flows (Saha *et al.*, 2018; Cui *et al.*, 2016). Nonetheless, with the

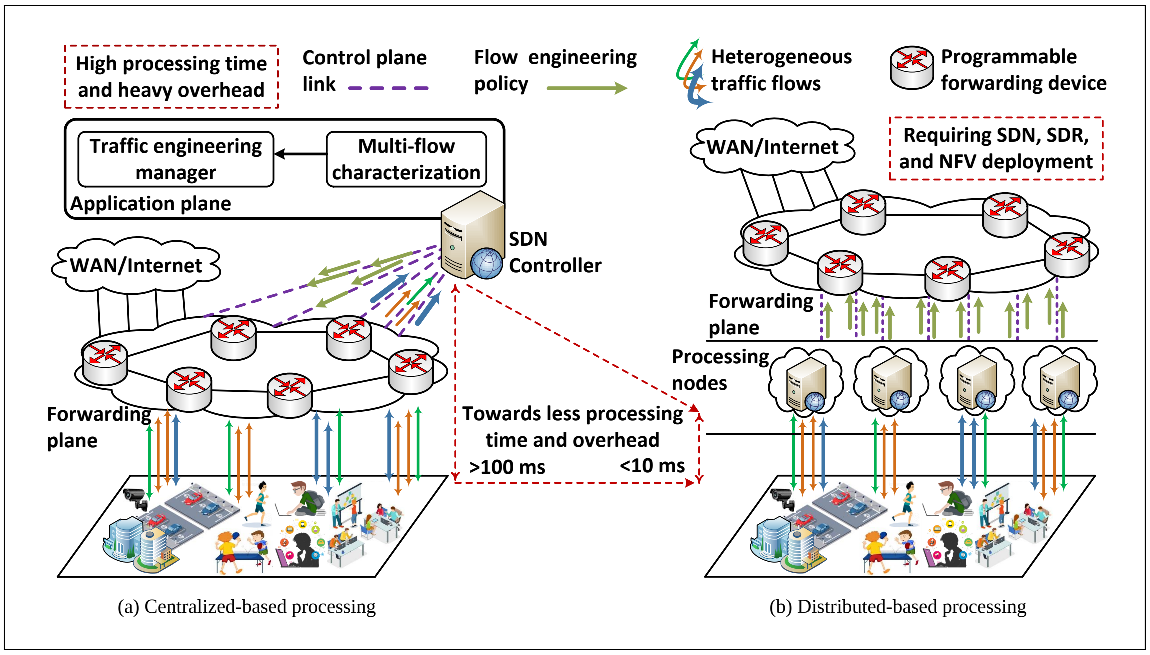

High processing time and heavy overhead
Control plane link
Flow engineering policy
Heterogeneous traffic flows
Programmable forwarding device
Traffic engineering manager
Multi-flow characterization
Application plane
SDN Controller
WAN/Internet
Forwarding plane
Towards less processing time and overhead
>100 ms
<10 ms
Requiring SDN, SDR, and NFV deployment
WAN/Internet
Forwarding plane
Processing nodes
(a) Centralized-based processing
(b) Distributed-based processing

Figure 5.2 Multi-flow processing over programmable architecture based on centralized vs. distributed approaches

continuous expansion in the flow number, data rates, and the requirements for detailed analysis, this approach seems to have limited scalability, and it leads to long processing time and heavy overhead that affect the performance of the delay-sensitive flows (Cui *et al.*, 2016; Hadi *et al.*, 2018; Mu *et al.*, 2018).

To overcome these problems, the distributed processing approach is proposed through the deployment of multiple processing nodes, such as a cluster of controllers or fog nodes, where each node controls only a specific part of the network's resources and its corresponding heterogeneous flows (Cui *et al.*, 2016) (Wu *et al.*, 2019). As shown in Figure 5.2(b), this approach provides data processing as close as possible to the end-devices, which enables reducing the processing time and the overhead as compared to the centralized approach. However, the administrator needs to encounter many issues related to the users' privacy, security, placement of the processing nodes, delay in computing, and energy consumption. Furthermore, being connected to heterogeneous devices, managing the distributed processing nodes, the connections between them, and the heterogeneous networks will be burden unless SDN, SDR, and NFV technologies are applied (Macedo *et al.*, 2015) (Wu *et al.*, 2019).

In our work, while taking into consideration the above concerns, we separate the flow characterization process from the traffic engineering and resource management process. We propose to perform flow characterization over each end-host in the campus network, based on a combined approach of a device (e.g., ID student laptop) and service (e.g., online gaming) identification, before forwarding flow through the network. We designate this approach as "distributed" since the characterization engine is not centralized over the centralized control plane but distributed over the end-hosts. This approach is developed as well in (Curtis *et al.*, 2011), where a shim layer is introduced over each end-host to detect the specific type of flow in a software-defined inter-data center network. The Differentiated Services Code Point (DSCP) bits are used to mark packets with the specific type of flow (Baker *et al.*, 2010). Thus, each non-intelligent forwarding device detects easily and directly the kind of the flow and decides to send it either to the destination using the proactively installed rules or to the centralized controller for path computation.

One of the benefits of characterizing flows by end-hosts is to protect users' privacy. In fact, accessing user data over a processing node (e.g., the SDN controller) and analyzing the corresponding flows may cause discomfort for end-users. Besides, characterizing flows before managing them through non-intelligent programmable devices enables saving network bandwidth and reducing processing time, since network flows are already characterized by sources, and consequently not all flows require the reactive intervention of the controller. Finally, this approach enables reducing controller tasks and improving controller efficiency. Section 5.6 presents more details about the functional description of this approach in the SRAFM scheme over the software-defined smart campus network.

5.4 SRAFM optimization model

In this section, we first outline the adopted network representation and related notations. Then, we present the SRAFM optimization model proposed, in this work, for a software-defined smart campus network.

5.4.1 Prerequisite notations and formalisms

Table 5.2 summarizes the notations used in this work. Let us assume that the network topology is represented by a connected graph $G = (V, E)$, where V is the set of all SDN-enabled devices (nodes), and E is the set of links.

With the following aspects, we formulate the SRAFM optimization model for the QoS aware routing and resource optimization problem.

5.4.1.1 The objective function

Assuming that every network service is sensitive to packet loss, we formulate the objective function to optimize the routing decision while considering both aspects: *the operational cost and the packet-loss*. Thus, we aim to minimize the global system cost experienced by a set of simultaneous flows F, as follows:

$$\min \sum_{f \in F} w_f \sum_{p \in P_f} \alpha \, S_p \, \delta(r_{f,p}) + \beta \, Q_p \, r_{f,p}. \tag{5.2}$$

In the remainder of this section, we elaborate on each of the settings used in this objective function. Hence, the path operational cost, S_p, is calculated by the sum of all the costs of the links belonging to path p, as shown in (5.3).

$$S_p = \sum_{e \in p} S_e \, , \; \forall \, p \in P. \tag{5.3}$$

Table 5.2 Summary key notations

Notation	Description
F	Set of all the traffic demands (flows).
f	A flow in F.
P	Set of paths for routing all flows F.
P_f	Set of paths for flow f.
P_e	Set of paths going through the link e.
p	A path, between source node s and destination node t, that could be used by flow f.
e	A directed link (i, j) outgoing from node i and incoming to node j.
$r_{f,p}$	Rate allocation for flow f on path p.
S_e	Operational cost of link e.
S_p	Operational cost of path p.
D_e	Delay of link e.
D_p	Delay of path p.
Q_e	Packet-loss probability of link e.
Q_p	Packet-loss probability of path p.
$C(e)$	Resource capacity of link e.
$C_{res}(e)$	Residual capacity of link e.
C_p	Resource capacity of path p.
D_f^{max}	Maximum end-to-end delay acceptable by a flow f.
R_f^{min}	Minimum required rate by a flow f.
w, α, β	Constants weights parameters.

The end-to-end loss rate probability, Q_p, on path p is calculated by the product of the individual packet loss ratios per link of all links belonging to path p (Begen *et al.*, 2005; Jurca & Frossard, 2007), as shown in (5.4).

$$Q_p = 1 - \prod_{e \in p} (1 - Q_e), \; \forall \; p \in P. \tag{5.4}$$

The $\delta(r_{f,p})$, defined in (5.5), presents an identity function to determine whether or not a flow f is routed through a path p.

$$\delta(r_{f,p}) = \begin{cases} 1, \text{ if } \ r_{f,p} > 0, \\ 0, \text{ if } \ r_{f,p} = 0. \end{cases} \tag{5.5}$$

We use the monetary parameters α and $\beta \in [0,\ 1]$ to model a multi-objective function enabling adjustment of the relative importance of the operational cost and the packet loss, depending on network and traffic characteristics (Egilmez *et al.*, 2013; Marler & Arora, 2010). Besides, we use the weight factor $w_f \in [0,\ 1]$ to adjust the priority level of the flows within the same set. Thus, paths with lower latency and higher bandwidth are assigned to high priority services, as detailed in Section 5.3.2.

5.4.1.2 Service end-to-end delay constraint

Equation (5.6) defines the end-to-end delay constraint, where the delay of the selected path D_p, as defined in (5.7), should be less than or equal to a specified end-to-end delay threshold, D_f^{max}, required by flow f.

$$D_p\ \delta(r_{f,p}) \ \leq \ D_f^{max},\ \forall f \in F. \tag{5.6}$$

$$D_p = \sum_{e \in p} D_e\,,\ \forall\ p \in P. \tag{5.7}$$

5.4.1.3 Service rate and network capacity constraints

Equation (5.8) defines the flow throughput constraint, where the rate of flow f on selected path p should be more than or equal to R_f^{min}, the minimum requirement of flow f.

$$\sum_{p \in P_f} r_{f,p} \geq R_f^{min},\ \forall f \in F. \tag{5.8}$$

Equation (5.9) defines the capacity constraint associated with each link $e \in E$, where the total rates of all the flows $f \in F$ going through a specified link e should not exceed the link's residual capacity.

$$\sum_{f \in F} \sum_{p \in P_e} r_{f,p} \leq C_{res}(e), \ \forall e \in E. \tag{5.9}$$

5.4.2 Optimization model

The objective of the SRAFM scheme is to select appropriate paths to accommodate a set of characterized network flows while minimizing the total system cost, ensuring the required QoS for each flow, and maximizing the overall network performance. Thus, path selection depends on the flows' QoS requirements and the constrained available network resources.
We formulate the optimization problem as follows:

$$\min \quad \sum_{f \in F} w_f \sum_{p \in P_f} \alpha \, S_p \, \delta(r_{f,p}) + \beta \, Q_p \, r_{f,p} \tag{5.10a}$$

$$\text{s.t.} \quad D_p \, \delta(r_{f,p}) \ \leq \ D_f^{max}, \forall f \in F, p \in P, \tag{5.10b}$$

$$\sum_{p \in P_f} r_{f,p} \geq R_f^{min}, \ \forall f \in F, \tag{5.10c}$$

$$\sum_{f \in F} \sum_{p \in P_e} r_{f,p} \leq C_{res}(e), \forall e \in E, \tag{5.10d}$$

$$\sum_{p \in P_f} \delta(r_{f,p}) = 1, \ \forall f \in F. \tag{5.10e}$$

Equation (5.10a) presents the objective function to be minimized while routing F_{ds} and F_{bs} sets of flows. Equations (5.10b) and (5.10c) present the delay and bandwidth constraints, where D_f^{max} and R_f^{min} characterize the QoS requirements of flow f in terms of delay and rate, respectively. Equation (5.10d) presents the capacity constraint associated with each link $e \in E$. Finally, in (5.10e), a single path routing decision is assumed for each flow. This assumption leads the controller to configure a small number of forwarding rules and simplifies our system in terms of time complexity since we regard a solution for a set of flows at the same time.

5.5 Approximation and decomposition framework for solving SRAFM

In this section, we propose a centralized scheme to be deployed over the SDN application plane. This SRAFM scheme solves the formulated optimization problem. First, since the SRAFM problem is NP-hard, we relax it into a tractable problem. Then, we design a per-flow decomposed algorithm to find the optimal routing solution in the network for a set of characterized flows.

5.5.1 The approximation algorithm

One of the difficulties of the primal problem (5.10) resides in the integrality condition of the binary variable $\delta(r_{f,p})$, which is activated only if a path p is selected to route flow f, as shown in (5.5). Relaxing this problem requires the relaxation of this variable by letting $\delta(r_{f,p})$ be a real variable in the range of [0,1]. For the SRAFM scheme, we approximate iteratively the original binary value $\delta(r_{f,p})$ into a real value using (5.11), a *Log-det* relaxation for approximation function (Fazel *et al.*, 2003), where $r^{t-1}_{f,p}$ is the rate result of the $(t-1)^{th}$ iteration and $\gamma > 0$ is a small positive constant.

$$\delta^t(r_{f,p}) = \frac{r_{f,p}}{r^{t-1}_{f,p} + \gamma}, \forall f \in F, \forall p \in P_f. \tag{5.11}$$

As shown in Algorithm 5.1 - line 5, in each iteration t, the integer-valued function $\delta(r_{f,p})$ is replaced by the new $\delta^t(r_{f,p})$ into the NP-hard original optimization problem (5.10) to formulate the tractable optimization problem, which will calculate the new rate value $r^t_{f,p}$ by invoking Algorithm 5.2, in Algorithm 5.1 - line 6. The relaxed problem is a convex optimization problem, which can guarantee the convergence as proven in (Liu *et al.*, 2016). Thus, these processes are repeated until the optimal rate value $r^*_{f,p}$ is achieved upon convergence, i.e., $r^{t-1}_{f,p} \approx r^t_{f,p} = r^*_{f,p}$, with an adequately small ϵ_1.

Algorithm 5.1 Approximation algorithm for SRAFM

```
1: **Initialization:** $t = 0$, $r_{f,p}^{0} = 1 - \gamma$;
2: **Repeat:**
3:   $t = t + 1$;
4:   $\delta^{t}(r_{f,p}) = \frac{r_{f,p}}{r_{f,p}^{t-1}+\gamma}$, $\forall f \in F, \forall p \in P_f$, where $r_{f,p}^{t-1}$ is the
     solution of the previous iteration, and $\gamma > 0$ is a small positive constant;
5:   Replace $\delta(r_{f,p})$ in (5.10) by $\delta^{t}(r_{f,p})$ to relax the problem;
6:   Call Algorithm 5.2 to find $r_{f,p}^{t}$;
7: **Until:** $\left| r_{f,p}^{t} - r_{f,p}^{t-1} \right| \leq \epsilon_1$;
8: **Return** $r_{f,p}^{*} = r_{f,p}^{t}$.
```

The understanding of the approximation of the modified problem to the original problem upon convergence is proved as follows:

$$\delta^{t}(r_{f,p}^{*}) = \frac{r_{f,p}^{*}}{r_{f,p}^{t-1} + \gamma} \approx \begin{cases} 1, \text{ if } r_{f,p}^{*} > 0, \\ 0, \text{ if } r_{f,p}^{*} = 0. \end{cases} \tag{5.12}$$

Equation (5.12) shows that, upon convergence, $\delta^{t}(r_{f,p}^{*})$ of the optimal solution approximately approaches the binary function $\delta(r_{f,p}^{*})$ of the original problem. As a result, the objective function involving $\delta^{t}(r_{f,p}^{*})$ eventually approximates that of the original problem.

5.5.2 The dual decomposition algorithm

The original problem (5.10) becomes more tractable by applying the *Log-det* relaxation function. However, its computational complexity is still very high, since this relaxed problem encompasses a wide range of heterogeneous flows with different service requirements. Therefore, to improve the performance of the resolution, we advocate the Lagrangian Dual Decomposition (LDD) approach to find the optimal solution in an efficient decomposed strategy. In other words, we

Algorithm 5.2 The dual decomposition algorithm for SRAFM

```
1: **Initilization:** $\lambda_e^0,\ l = 0$;
2: **Repeat:**
3:   **Foreach** $f \in F$ **do:**
4:     Solving the sub-problem (5.17) to find the rate
       allocation variable $r_{f,p}$, for each flow;
5:   **End**
6:   Perform a subgradient update using (5.18) to find the
     dual variables $\lambda_e^{(l+1)}$;
7:   $l = l + 1$;
8: **Until:** $\left|\lambda_e^{(l)} - \lambda_e^{(l-1)}\right| \leq \epsilon_2$;
9: **Return** $r_{f,p}$ to Algorithm 5.1 (line 6) as the rate result
           of the $t^{th}$ iteration.
```

decompose the relaxed optimization problem into *F-optimization* sub-problems, also called *per-flow optimization* sub-problems, that can be solved simultaneously with low complexity. Hence, first, the SRAFM scheme independently solves an optimization sub-problem for each flow within the whole set. Then, it coordinates between all of them and the allocated constrained resources.

For instance, to find the rate solution of the t^{th} iteration of Algorithm 5.1, the Lagrangian function of the relaxed primal problem in (5.10) can be formulated as follows, after decoupling the constraint in (5.10d):

$$L(r_{f,p}, \lambda_e) = \sum_{f \in F} w_f \Big(\sum_{p \in P_f} \alpha\, S_p\, \delta^t(r_{f,p}) + \beta\, Q_p\, r_{f,p} \Big) + \sum_{e \in E} \lambda_e \Big(\sum_{f \in F} \sum_{p \in P_e} r_{f,p} - C_{res}(e) \Big), \tag{5.13}$$

where $\lambda_e \geq 0$ presents the Lagrange multiplier associated with the link capacity. Consequently, the Lagrangian dual function is given by:

$$G(\lambda_e) = \begin{cases} \min \ L(r_{f,p}, \lambda_e), \\ \text{s.t. } (5.10b), \ (5.10c), \ (5.10e). \end{cases} \tag{5.14}$$

The dual problem is formulated as follows:

$$\max_{\lambda_e \geq 0} \ G(\lambda_e). \tag{5.15}$$

For a fixed dual variable λ_e, (5.14) is decomposed into *F-optimization* sub-problems, which can be solved with the following objective function:

$$\min_{f \in F} \quad \sum_{p \in P_f} r_{f,p} \ w_f \Big(\frac{\alpha \ S_p}{r_{f,p}^{t-1} + \gamma} + \beta \, Q_p \Big) + \sum_{e \in E} \sum_{p \in P_e} r_{f,p} \ \lambda_e, \tag{5.16}$$

where the constant terms, in (5.13), can be removed, and the term $r_{f,p}^{t-1}$ is the rate solution of the previous iteration of Algorithm 5.1, which is calculated by Algorithm 5.2. Thus, the whole tractable optimization problem is decomposed into *per-flow optimization* sub-problems and presented as follows:

$$\min_{f \in F} \quad \sum_{p \in P_f} r_{f,p} \ w_f \Big(\frac{\alpha \ S_p}{r_{f,p}^{t-1} + \gamma} + \beta \, Q_p \Big) + \sum_{e \in E} \sum_{p \in P_e} r_{f,p} \ \lambda_e \tag{5.17a}$$

s.t.

$$D_p \ \delta^t(r_{f,p}) \leq D_f^{max}, \forall p \in P, \tag{5.17b}$$

$$\sum_{p \in P_f} r_{f,p} \geq R_f^{min}, \tag{5.17c}$$

$$\sum_{p \in P_f} \delta^t(r_{f,p}) = 1, \tag{5.17d}$$

$$0 \leq \delta^t(r_{f,p}) \leq 1. \tag{5.17e}$$

As shown in (5.17), each sub-problem is intended to select the path that has the minimum cost, which depends on the operational cost, the packet loss, and the rates of all its shared links. The dual variable, which depends on the flow rate variable, presents a penalty to prevent the allocation of flows into congested paths. Hence, flows are spread out into multiple different paths to obtain lower cost while ensuring the delay and bandwidth requirements using constraints (5.17b) - (5.17e), which are the remaining constraints of problem (5.10).

Finally, to solve the dual problem (5.15), the subgradient projection method is deployed (Lin *et al.*, 2006). Thus, the updating rule for the dual variable is formulated as follows:

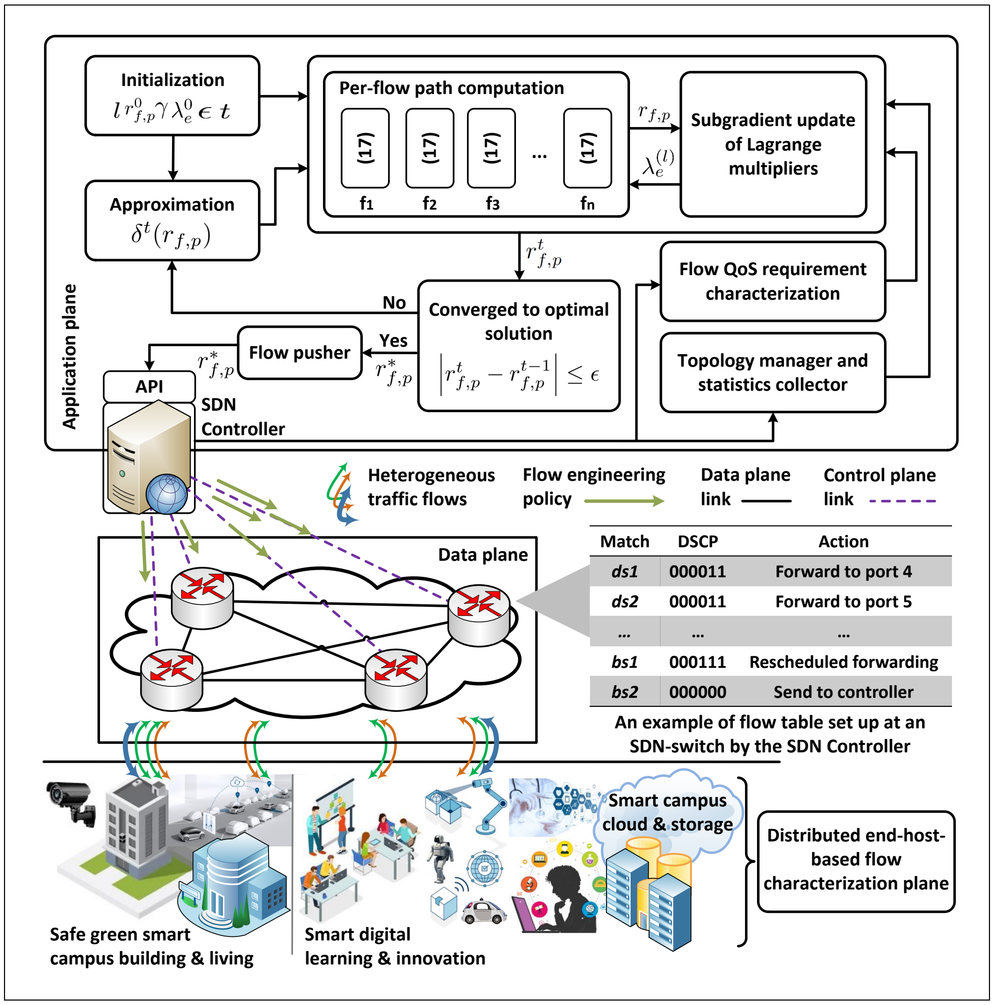
$$\lambda_e^{(l+1)} = \left[\lambda_e^{(l)} + \kappa \Big(\sum_{f \in F} \sum_{p \in P_e} r_{f,p} - C_{res}(e) \Big) \right]^+, \forall e \in E. \tag{5.18}$$

The $\lambda_e^{(l+1)}$ reports the evolution of the rate values at each link $e \in E$, where κ is a non-negative step-size used to adjust the convergence of the dual decomposition algorithm (Palomar & Chiang, 2006).

The steps of Algorithm 5.2 can be summarized as follows. Given the value of $\delta^t(r_{f,p})$ from Algorithm 5.1 and the set F of network flows, at each iteration, Algorithm 5.2 solves independently the *per-flow optimization* problems to find an optimal path p for each flow $f \in F$ (lines 3 - 5). Then, using the computed value $r_{f,p}$, Algorithm 5.2 (line 6) updates the Lagrangian multiplier λ_e for each link $e \in E$ using the subgradient method in (5.18). The algorithm goes in a loop until the change of dual values approximates the stop threshold (line 8). Accordingly, the rate result $r_{f,p}$ is returned to Algorithm 5.1 (lines 6 - 7) as the solution for its t^{th} iteration (i.e., $r^t_{f,p}$) to check the convergence to the optimal-rate solution $r^*_{f,p}$.

5.6 SRAFM functional description

In this section, we discuss the different components included in the whole SRAFM scheme. Figure 5.3 presents an overview of the functional description of SRAFM.


Initialization
$l\, r^0_{f,p} \gamma \lambda^0_e \epsilon\, t$
Approximation
$\delta^t(r_{f,p})$
Per-flow path computation
(17)
(17)
(17)
...
(17)
f1
f2
f3
fn
$r_{f,p}$
$\lambda^{(l)}_e$
Subgradient update of Lagrange multipliers
$r^t_{f,p}$
Flow QoS requirement characterization
No
Converged to optimal solution
$\left| r^t_{f,p} - r^{t-1}_{f,p} \right| \leq \epsilon$
Yes
Flow pusher
$r^*_{f,p}$
$r^*_{f,p}$
Topology manager and statistics collector
API
SDN Controller
Application plane
Heterogeneous traffic flows
Flow engineering policy
Data plane link
Control plane link
Data plane
Smart campus cloud & storage
Distributed end-host-based flow characterization plane
Safe green smart campus building & living
Smart digital learning & innovation

Match	DSCP	Action
ds1	000011	Forward to port 4
ds2	000011	Forward to port 5
...	...	...
bs1	000111	Rescheduled forwarding
bs2	000000	Send to controller

An example of flow table set up at an SDN-switch by the SDN Controller

Figure 5.3 SRAFM functional description

5.6.1 Distributed end-host-based flow characterization plane

When end-users (e.g., students, professors, etc.) connect to the network using their user ID, traffic flows are actively characterized in order to be signaled for the non-intelligent forwarding devices. Flows are characterized on the basis of a combined approach of device-level (e.g., ID professors' devices, ID online-courses' devices, ID students' devices, etc.) and service-level (e.g., video-conferencing, skyping, online gaming, etc.). Then, to ensure an efficient and easy detection of each flow type by the SDN-forwarding device, the DSCP field (as one of the marking techniques) is used to mark the corresponding packets of each flow with the required priority level. Accordingly, the programmable forwarding element detects the type of each flow smoothly and forwards it either to the controller or to the destination using the corresponding engineering policies (actions), as shown in Figure 5.3.

As explained in Section 5.3, this distributed end-host-based flow characterization plane enables protecting the privacy of end-users, reducing the processing time required for heterogeneous flow analysis, and reducing controller tasks.

5.6.2 Software defined-based proactive and reactive approach for flow management

In our work, the proactive and reactive SDN operational modes are deployed to manage heterogeneous network flows. It is worth mentioning that the existing controllers (e.g., Floodlight, OpenDaylight, NOX, etc.) are by default configured to use only the reactive mode. Meanwhile, the SDN-forwarding devices (e.g., OpenFlow Logical Switch) are by default featured to support both of the SDN modes (Mendiola *et al.*, 2017).

With SRAFM, we propose a new controller design that supports proactive and reactive strategies, simultaneously, to manage the regular and irregular network flows, respectively. On the one hand, as explained in Section 5.3.2, the regular and frequent flows of the F_{ds} set (e.g., courses, video-conferences, and video-surveillance) could be managed within a proactive strategy since the controller is proactively aware of the state of the network and the required resources to ensure the best QoS for these highly critical flows. Hence, their rules are configured in advance in the network, so when they come to the non-intelligent forwarding devices, they are easily identified because of DSCP bits and directly transferred to the destination using a matching between the DSCP bits and the corresponding actions in the flow table, as shown in Figure 5.3. This SDN operational mode does not require invoking the controller to manage these demands. Consequently, it avoids the time and the amount of overhead needed during the negotiation between the SDN-forwarding devices and the controller to compute paths and configure the network. It also reduces latency and bandwidth consumption of frequent demands, which affect the control plane scalability significantly. On the other hand, the irregular and infrequent IoT flows, the non-critical flows of the F_{ds} set, and the F_{bs} flows are managed in the network using the SDN reactive approach. In other words, switching devices need to request the controller, when the first packet of each flow arrives, in order to compute the corresponding routing policy based on the current state of the network, the number of demands (i.e., flows), and the flows' QoS requirements.

To the best of our knowledge, none of the existing works, in the literature, considered both reactive and proactive SDN modes for heterogeneous flow management and resource optimization.

5.6.3 Software-defined controller modules

The flow engineering policy (action) computation process is performed by mainly four SRAFM modules implemented on the SDN application plane. The first module is the topology manager and statistics collector module. It enables maintaining a global view of the network, which is an input to the path computation modules. It continuously monitors and stores information about all the links and devices currently up in the network. The efficiency of path calculation and resource allocation in the network relies on the accuracy of the network data collected by this function. Hence, many technologies and parameters need to be deployed and configured by the administrator in this functionality to discover and update the entire network state with an efficient strategy, such as the REST-API (Floodlight, Accessed: October 20, 2020), the Link Layer Discovery Protocol (LLDP) (Krishnan *et al.*, 2007), the interval at which network statistics are collected, etc.

The second module is the *flow QoS requirement characterization* module. On the basis of the DSCP bits, this module analyzes the specific flows' QoS requirements and priority levels, as explained in Section 5.3. This analysis is also employed as an input for the *per-flow path computation* module to find the optimal solution and to map the set of flows to the appropriate network engineering policies. Accordingly, the *per-flow path computation* algorithm, which is the third main module in the controller, calculates the optimal solution for the set of characterized flows, based on the QoS requirements, the network topology status, and the *Lagrange multipliers*, as shown in Figure 5.3, and detailed in Sections 5.4 and 5.5. Finally, when the algorithm converges to the optimal solution, the SDN controller installs the action rules in the forwarding devices over the selected paths using the flow pusher module (the fourth module).

5.7 Performance Evaluation

In this section, we evaluate our SRAFM scheme, first, in terms of running time, system cost, end-to-end delay, average rate allocation, and resource availability while comparing our proposed mechanisms to the state-of-the-art methods. Then, we discuss the percentages of QoS violation and flow rejection. In each of these experiments, we emphasize the importance of performing flow characterization over the distributed end-host-based plane.

5.7.1 Simulation settings and benchmarks

5.7.1.1 Hardware and simulation settings

All the experiments were carried out on a PC with an Intel Core i7 CPU @ 3.4 GHz with 16 GB memory. The simulations are performed using the modern programming environment, Julia software (Bezanson, Accessed: April 23, 2020; Bezanson *et al.*, 2017; Lubin & Dunning, 2015; Besard *et al.*, 2018), through the mathematical language called Julia for Mathematical Programming (JuMP) (Dunning *et al.*, 2017) and the optimization solver called IPOPT (Andreas Wächter, 2019. Accessed: April 2, 2019). According to the network topology, we consider Abilene from the SNDlib library, which consists of 12 nodes and 15 links (Orlowski *et al.*, 2010). Then, to evaluate the applicability of our proposed scheme over a large-scale network, we adopt AttMpls topology from the Internet Topology Zoo, which consists of 25 nodes and 57 links (Saha *et al.*, 2018). To evaluate the performance of our algorithms under different network conditions, initially, all links' capacities are set to 10 GB and maximum 30% of the links' capacities are randomly consumed. Initially, the delays of links have also been randomly set from 5 to 10 ms and we assume that this value includes the processing, transmission, and queuing delays. We set the operational cost metric randomly according to the load of each link (Cisco-CCNA, Accessed: October 20, 2020), similarly for the delay and the packet-loss metrics (Egilmez *et al.*, 2012), during flow allocation.

5.7.1.2 Benchmark schemes

To evaluate our work, we use LARAC (Juttner *et al.*, 2001) and SWAY (Saha *et al.*, 2018) algorithms as baselines. The LARAC algorithm is a single metric QoS routing scheme, i.e., a delay-constrained least-cost algorithm. It uses the Lagrange relaxation method to iteratively calculate the best QoS path on the basis of an aggregated concept of cost, which includes the operational cost and the delay (Juttner *et al.*, 2001). The SWAY scheme is a multi-metric QoS routing approach that considers two different strategies to manage loss and delay-sensitive flows. Both of the SWAY algorithms are based on the Yens K-shortest paths algorithm (Yen, Accessed: October 20, 2020). Thus, SWAY minimizes the delay metric as a cost function for the delay-sensitive traffic and the loss metric as a second cost function for the loss-sensitive traffic, according to different constraints. In fact, we cannot implement exactly the same algorithms as SWAY since the types of traffic are different. Hence, we adopt only the same strategy of using two different algorithms for delay and bandwidth sensitive flows. Then, according to the type of flow (either ds or bs), SWAY alternates between the two algorithms.

5.7.1.3 Heterogeneous flows generation

We randomly generate different smart campus services through the network with different QoS requirements, as detailed in Section 5.3 and shown in Tables 5.1 and 5.3. We generate 30% of the whole set of heterogeneous flows as IoT services, 50% as delay-sensitive, and 20% as bandwidth-hungry applications. As shown in Table 5.3, the rate requirements of the generated flows are set in the range of 150 Kbps to 100 Mbps. In addition, to evaluate the end-to-end delay, the thresholds of the heterogeneous F_{ds} services are set according to the type of traffic, either interactive applications in the range of 150 to 400 ms or IoT services, for which we use two different services with delay constraints of 2.6 s and 0.9 s, as shown in Table 5.3. It is noteworthy to mention that the values adopted in our simulation in terms of rate and delay present examples

Table 5.3 Examples of campus network services and their requirements

Activity	**Delay**	**Rate**
Email and web browsing	≤ 2 - 5 s	10 Kbps
Downloading a digital book of 1 MB	–	1.5 Mbps
Online learning	≤ 150 ms	2 Mbps
HD-quality video streaming	≤ 400 ms	4 Mbps
Skype-group video session (7-10 people)	≤ 150 ms	8 Mbps
Game: World of Warcraft	≤ 250 ms	50 Kbps
Downloading movies for offline watching during 8 minutes	–	100 Mbps
Motion sensor - IoT service	≤ 2.6 s	2.8 - 3.8 Mbps
Voice command and control IoT service	≤ 0.9 s	150 Kbps
Video surveillance	≤ 150 ms	20 Mbps

of characteristics for these services. For instance, an online game service could run smoothly with a delay less than 250 ms, as mentioned in Table 5.3, while an excellent online game (which is not our objective) requires less than 50 ms; also, the more players in an online game, the more data are exchanged between players (Mocnej *et al.*, 2018).

5.7.2 Results and discussion

5.7.2.1 System running time evaluation

Table 5.4 evaluates the average running time by the number of flows over each topology for each routing strategy. We observe that SRAFM takes more time to select the optimal solution compared to the benchmark algorithms. This difference in terms of required time to achieve the optimal solution is due to the complexity of our algorithm compared to state-of-the-art ones. SRAFM performs coordination between the service quality requirements of the whole set of flows and the available network resources to select the best solution, while LARAC and SWAY select a local best path for each flow until the whole set of flows is achieved. Though these flow-per-flow routing strategies can obtain a fast solution to make a decision for all flows, they cannot reach an optimal routing decision for the whole set, because of the great number of

Table 5.4 Evaluation of running time (s)

Topology		Abilene			AttMpls		
Number of flows		1000	1500	2000	1000	1500	2000
Algorithms	SRAFM	126	534	1053	302	917	1726
	SWAY	4.2	–	–	–	–	–
	LARAC	5.6	–	–	–	–	–

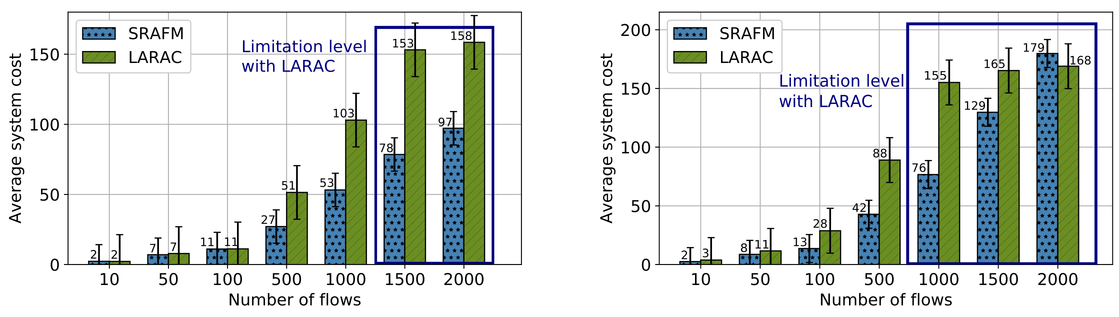
rejected flows, as shown in Table 5.4 from the set of 1500 flows with the Abilene topology and the set of 1000 flows with the AttMpls topology.

It is also worth mentioning that solving SRAFM with network topologies as Abilene and AttMpls is not difficult and provides a straightforward solution for a set of simultaneous flows. However, in a real network, the number of paths is extremely large, which makes this problem more complicated. Therefore, the use of parallel computation in multi-core systems and the deployment of the cluster-based distributed controller technology are advised to alleviate this challenge, especially over large-scale networks.

5.7.2.2 System cost evaluation

In this section, we evaluate the average of the system cost according to the selected solutions. Since the SWAY algorithm takes into account two different cost functions (i.e., it minimizes the delay metric for delay-sensitive flows and the bandwidth utilization metric for bandwidth sensitive flows), we consider only the LARAC algorithm to evaluate the system cost of the SRAFM scheme. Figure 5.4 depicts the change in the average cost after the allocation of the whole set of flows.

As shown in Figure 5.4a and Figure 5.4b, SRAFM shows its outperformance compared to the LARAC algorithm in terms of cost as a result of its distributed rate allocation approach. With the LARAC algorithm, path selection does not depend on the allocation of flow rates, and the cost function increases in accordance with the path operational cost and the delay that are updated after flow allocation has been performed (Juttner *et al.*, 2001). SRAFM achieves



a) With characterization b) Without characterization

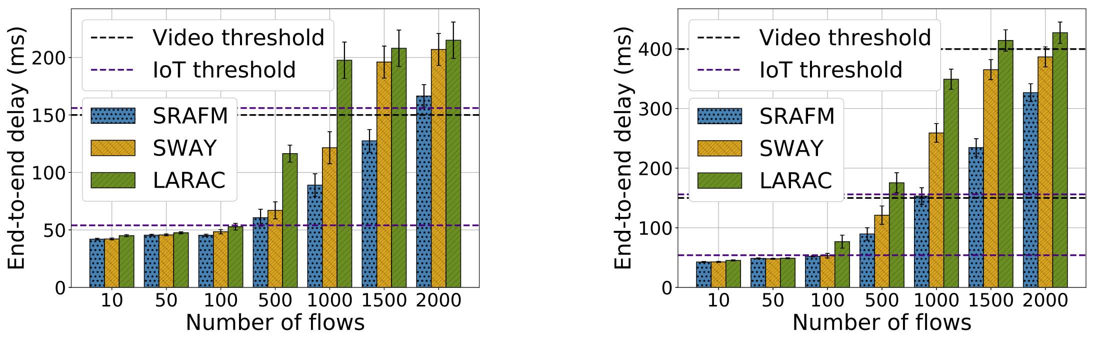
Figure 5.4 System cost evaluation

51% reduction in system cost as compared to LARAC. The system cost with LARAC does not increase compared to SRAFM from 1500 characterized flows and 1000 non-characterized flows, as shown in Figure 5.4a and Figure 5.4b, respectively. This is indirectly dependent on the number of QoS violated flows, i.e., reaching the limitation level. In fact, at the limitation level, LARAC rejects all new flows as a result of QoS violation, so its system cost does not increase compared to the SRAFM scheme, which still accepts flows and consequently its system cost continues to increase. For example, in Figure 5.4b with 2000 non-characterized flows, the system cost of SRAFM is greater than the cost of LARAC because of the flow acceptance by SRAFM, but its rejection by LARAC.

As shown in Figure 5.4b, both of the algorithms (i.e., SRAFM and LARAC) reveal the necessity to perform traffic characterization before traffic management, which reduces the system's cost by an average of 47% compared to traffic management without the characterization process.

5.7.2.3 QoS requirement provisioning and flow violation

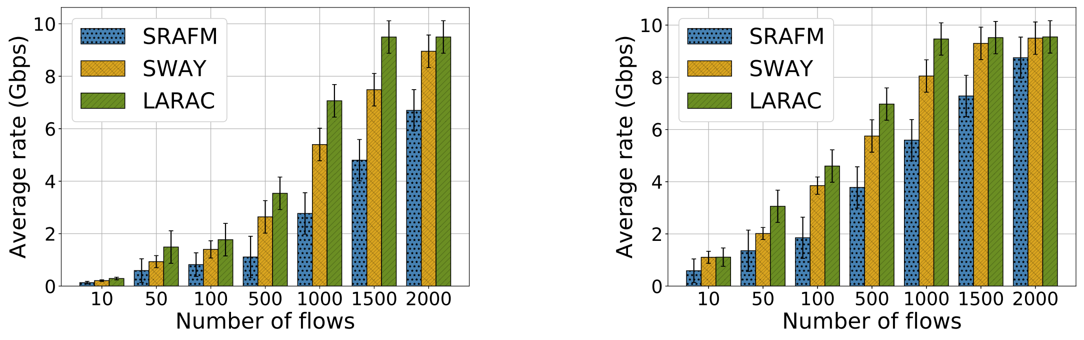
Figures 5.5, 5.6, and 5.7 evaluate, respectively, the average of the end-to-end delay, the average of the flow rate allocation, and the lowest bandwidth availability after performing flow allocations over the Abilene topology. From these figures, it is evident that SRAFM obtains the highest



a) With characterization

b) Without characterization

Figure 5.5 End-to-end delay evaluation



a) With characterization

b) Without characterization

Figure 5.6 Rate allocation evaluation

network performance as compared to SWAY and LARAC, especially with the increasing number of flows.

SRAFM reduces, on average, the end-to-end delay by 21% and 34% as compared to SWAY and LARAC, respectively. Traffic characterization improves the system end-to-end delay with more than 56%, as compared to the management of the heterogeneous flows without characterization, particularly with a huge set of flows.

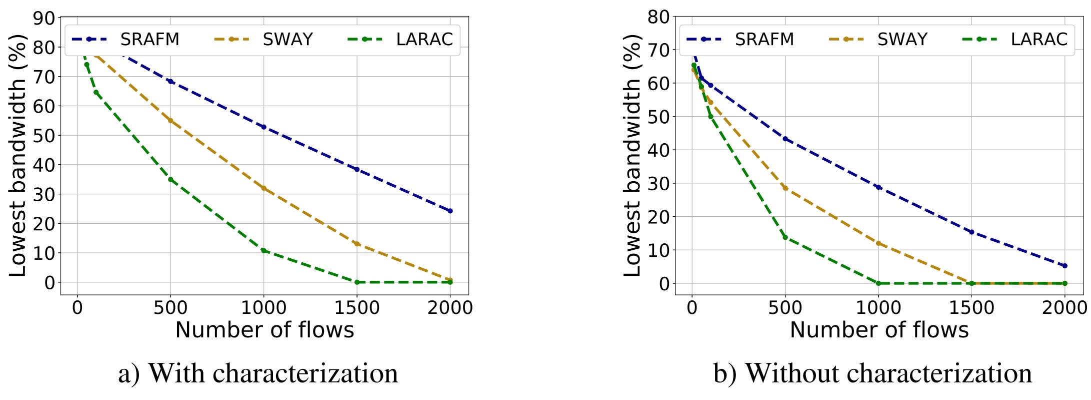
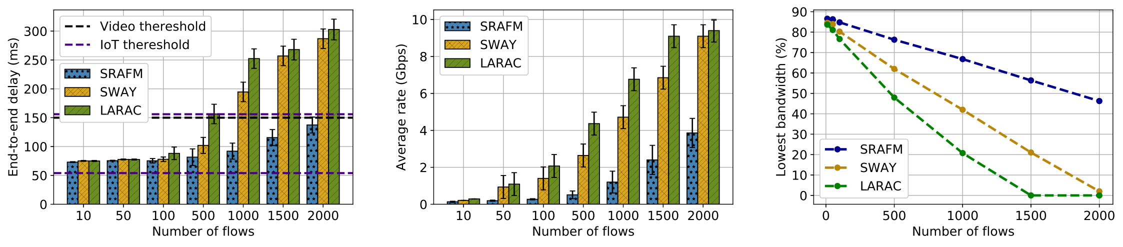


Figure 5.7 Bandwidth availability evaluation

Figures 5.6 and 5.7 evaluate the average rate allocation and the lowest available bandwidth after performing flows rate allocations in the network. SRAFM reduces on average the rate allocation as a result of the distributed approach by 32% and 48% as compared to SWAY and LARAC, respectively. SRAFM achieves an average improvement of 27% and 36% in terms of available bandwidth as compared to SWAY and LARAC, respectively. Furthermore, with the benchmarks schemes, the lowest available bandwidth achieves 0% with huge sets of flows (i.e., 1500 and 2000 flows), which is not the case with SRAFM. Finally, as shown in Figures 5.7a and 5.7b, controlling bandwidth-hungry flows using flow characterization provides a 22% improvement in bandwidth availability and enables postponing network congestion in terms of flow numbers.

This improvement in terms of delay and bandwidth availability depends on the routing strategy of each algorithm. With SRAFM, the rate allocation on each path for each flow depends on the rate requirements of all the flows. In other words, the SRAFM algorithm allocates network flows distributively to network paths that have low bandwidth utilization due to the dual variable, which presents a penalty in the cost minimization function. Furthermore, with SRAFM, the selected solution is related to two constrained metrics, the delay and bandwidth thresholds. On the other hand, with the current state of the network, the benchmark algorithms perform flow-per-flow allocation for the whole set of flows. Thus, at a limitation level, the new services



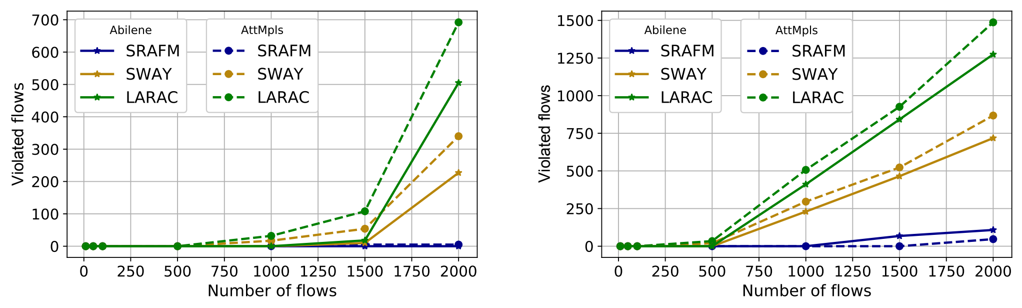
a) End-to-end delay b) Rate allocation c) Bandwidth availability

Figure 5.8 Varying number of characterized flows managed over a large-scale network

are rejected because of the paths' insufficient bandwidth (with the SWAY algorithm) and high latency (with both the SWAY and LARAC algorithms).

Figure 5.8 evaluates the end-to-end delay, the average of rate allocation, and the lowest bandwidth availability, respectively, after the allocation of characterized flows over the AttMpls topology.

From Figures 5.8a, 5.8b, and 5.8c, it is evident that SRAFM obtains the highest network performance compared to SWAY and LARAC, and also compared to the small-scale Abilene topology. SRAFM achieves 47% and 56% reduction in end-to-end delay compared to SWAY and LARAC, respectively. Our algorithm achieves 14% reduction in end-to-end delay with the AttMpls topology compared to the Abilene network, with the huge sets of flows. In terms of rate allocation, with AttMpls, SRAFM reduces the average rate allocation by 62% and 74% compared to SWAY and LARAC, respectively, and by 49% compared to the Abilene network. With AttMpls, SRAFM also improves the lowest available bandwidth by 28% compared to the Abilene network. This improvement in terms of network performance with a large-scale network is related to the distributed rate allocation approach over the whole topology deployed by SRAFM and so for path delay, which is related not only to the number of links but also to the number of flows processed throughout the path. On the other hand, with local-based path selection strategies like SWAY and LARAC, the average rate allocation remains similar compared to the Abilene network, and the end-to-end delay increases because of the large-scale setting.



a) With characterization b) Without characterization

Figure 5.9 QoS violated flows within Abilene and AttMpls topologies

Table 5.5 Examples of values for QoS violated flows

Abilene topology		Characterized flows							Non-characterized flows						
		10	**50**	**100**	**500**	**1000**	**1500**	**2000**	**10**	**50**	**100**	**500**	**1000**	**1500**	**2000**
Algorithms	**SRAFM**	0	0	0	0	0	0	0	0	0	0	0	0	68	108
	SWAY	0	0	0	0	0	10	227	0	0	0	0	230	465	718
	LARAC	0	0	0	0	0	18	505	0	0	0	0	411	842	1274
AttMpls topology		**Characterized flows**							**Non-characterized flows**						
		10	**50**	**100**	**500**	**1000**	**1500**	**2000**	**10**	**50**	**100**	**500**	**1000**	**1500**	**2000**
Algorithms	**SRAFM**	0	0	0	0	0	0	0	0	0	0	0	0	0	47
	SWAY	0	0	0	0	17	54	340	0	0	0	20	296	523	868
	LARAC	0	0	0	0	32	108	692	0	0	0	34	507	926	1487

This SRAFM's distributed rate allocation approach has an improvement not only on the end-to-end delay and the rate allocation but also on the violated flows. Figure 5.9 and Table 5.5 show the outperformance of our proposed scheme in terms of flow violation as compared to the benchmark schemes in the Abilene and AttMpls networks, particularly with a massive number of flows.

As shown in Figure 5.9a, with the Abilene topology, SWAY and LARAC achieve the limitation level from 1500 flows; but with the AttMpls topology, the limitation level is achieved from 1000 characterized flows. With 2000 characterized flows in the Abilene topology, 11% and 29% of the whole set are rejected by SWAY and LARAC, respectively. In the AttMpls topology, likewise, 17% and 36% of the whole set are rejected by SWAY and LARAC, respectively. This increase in violated flows in the AttMpls topology compared to Abilene with SWAY and LARAC is due to the local rate allocation strategy and the increase in terms of end-to-end delay. Meanwhile, the distributed rate allocation approach adopted by SRAFM enables an extenuated use of the topology resources, which leads to outperformance in terms of flow violation, with 0%.

Figure 5.9b proves that traffic characterization diminishes significantly the percentage of rejected flows. Without flow characterization, SRAFM achieves the limitation level with a set of 1500 flows over the Abilene topology, and with a set of 2000 flows over the AttMpls topology. SRAFM reduces the percentage of QoS flow violations with 2000 flows by 30% and 58% in the Abilene network compared to SWAY and LARAC, respectively. In AttMpls, SRAFM reduces the percentage of QoS flow violations with 2000 flows by 41% and 71% compared to SWAY and LARAC, respectively.

5.8 Conclusion and future directions

In this paper, we proposed a multi-flow management scheme for a software-defined smart digital campus network, taking into account the available network resources, the heterogeneous flow types, and their different QoS requirements. We presented a unified fully-programmable architecture, a distributed end-host-based flow characterization plane, a controller design with a proactive and reactive flow management strategy, and a centralized software-defined optimization model to manage the massive heterogeneous flows generated from thousands of interconnected devices.

We considered heterogeneous flows as either delay-sensitive or bandwidth-hungry services, but all as loss-sensitive. We introduced an approximation algorithm to relax the NP-hard proposed optimization problem. Furthermore, a per-flow decomposed optimization algorithm was deployed using LDD and sub-gradient methods to calculate the optimal routing paths for the whole set of flows on the basis of a distributed rate allocation design. The simulation

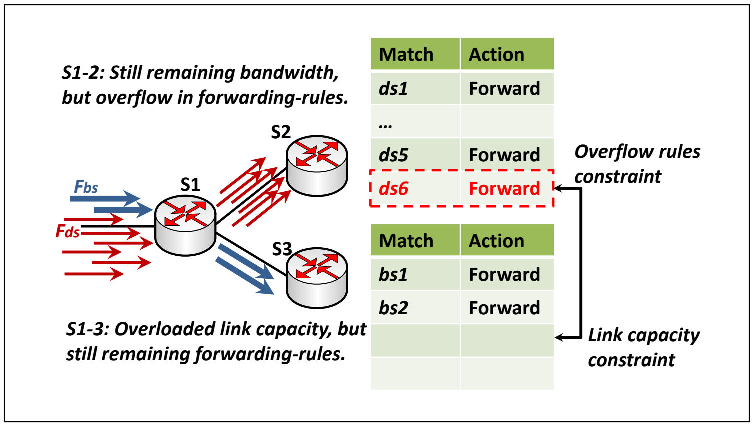

S1-2: Still remaining bandwidth, but overflow in forwarding-rules.
S2
S1
S3
Fbs
Fds
S1-3: Overloaded link capacity, but still remaining forwarding-rules.
Match | Action
ds1 | Forward
...
ds5 | Forward
ds6 | Forward
Match | Action
bs1 | Forward
bs2 | Forward
Overflow rules constraint
Link capacity constraint

Figure 5.10 Conflict of device forwarding-rules overflow and link capacity overload in a large IoT environment

results showed the outperformance of our proposed scheme in terms of reduction in system cost, end-to-end delay, average rate allocation, and rejected flow percentages.

The proposed SRAFM solution for a smart campus environment could be implemented in a large variety of smart networks, such as hospitals, small and medium-sized enterprises (SMEs), governmental office buildings, etc. It is also possible to implement the SRAFM optimization model over a large IoT environment. However, in such an environment, the limited number of flow rules to be configured on an SDN-forwarding device should be taken into account. In fact, without the rule-constraint, even when the remaining available bandwidth of a specific link is sufficient to accommodate new flow rates, the maximum number of flow rules to be configured on an SDN-forwarding device remains a constraint. Thus, this creates a conflict of device forwarding rules overflow, which produces flow rejection, as shown in Figure 5.10.

Our future work will focus on the efficiency and scalability of the control plane, on the deployment of the cluster-based distributed controller technology, and on the deployment of the distributed flow characterization solution, since all these issues have significant impacts on network performance with massive data streams.

CHAPTER 6

MACHINE LEARNING AND DEEP LEARNING-BASED EVALUATION FRAMEWORK FOR IoT AND NON-IoT SMART NETWORK TRAFFIC ANALYTICS

Yosra Njah [a,b] , Misha Cattaneo [b] , Jean Hennebert [b] , Mohamed Cheriet [a]

[a] Department of Automated Production Engineering, École de Technologie Supérieure (ÉTS), 1100 Notre-Dame Ouest, Montréal, Québec, Canada H3C 1K3

[b] Haute École d'Ingénierie et d'Architecture de Fribourg, 1705 Fribourg, Switzerland

Submitted to IEEE Internet of Things Journal, October 2021

Abstract

The explosive expansion of mobile and sensing devices, cloud services, and video traffic has raised unprecedented challenges for network administrators regarding the effective management of constrained resources and massive heterogeneous flows. An accurate understanding of the different types of network traffic demands (e.g., business-critical, low priority, malicious traffic, etc.) is fundamental because it enables administrators to be fully aware of network content and thereby proactively optimize network performance. However, traditional classification approaches (e.g., port and payload-based inspection) have become inefficient because of the rise of different emerging technologies, such as new services, encryption, and encapsulation. This work provides a comprehensive evaluation framework about flow-based inspection methods to serve as a map for designing lightweight engines for fine-grained Internet of Things (IoT) and non-IoT service differentiation in smart environments while supporting end-user privacy and the evolving nature of traffic. First, we present a feature importance review that determines the best minimal set of attributes characterizing IoT and non-IoT services. Second, to review traffic classification, we conduct rigorous comparisons between different machine and deep learning-based models. In particular, while deep learning deployment is still thorny and presently less approved in network traffic classification, we justify the deep learning high complexity and training requirements. The experimental results show that random forest achieves the highest accuracy for service identification. However, it is highly dependent on the selected features. On the other hand, while deep learning techniques (CNN and LSTM) achieve accurate inference, they eliminate the need for handcrafted feature engineering through automatic learning processes.

Keywords: Smart environments, Internet of Things (IoT), flow-based inspection, feature engineering, machine learning, random forest, and deep learning.

6.1 Introduction

With the deployment of the Internet of Things (IoT) in smart environments (e.g., homes, campuses, vehicles, and cities), the number of devices connecting to the Internet is ballooning. Thus, as a result of the various general Internet applications developments, such as web technologies, social media, online video games, etc., as well as the plethora of IoT devices that autonomously interact with each other and with remote controllers and servers on the Internet (e.g., edge/fog and cloud data-centers), the network traffic data is exploding at an unprecedented rate, bringing the next wave of Internet growth (Cisco, Accessed: September 12, 2021). According to Cisco, the fastest growing mobile device category is the IoT, which is also referred to as Machine-to-Machine (M2M) connections, followed by smartphones. The M2M is projected to grow at a 30% compound annual growth rate (CAGR) from 2018 to 2023 (i.e., from 6.1 billion in 2018 to 14.7 billion by 2023). Smartphones will grow at a 7% CAGR within the same period (Cisco, Accessed: September 12, 2021). The International Data Corporation (IDC) estimates that 6 billion users, or 75% of the world's population, will be interacting with online data every day by 2025. Another study produced by vXchnge company shows that there will be 41 billion IoT devices by 2027, while 70% of automobiles will be connected to the Internet by 2023 (vXchnge, Accessed: September 12, 2021). More precisely, connected home applications (e.g., home automation, home security and video surveillance, connected white goods, and tracking applications) will represent 48% or nearly half of the total M2M connections by 2023 (Cisco, Accessed: September 12, 2021).

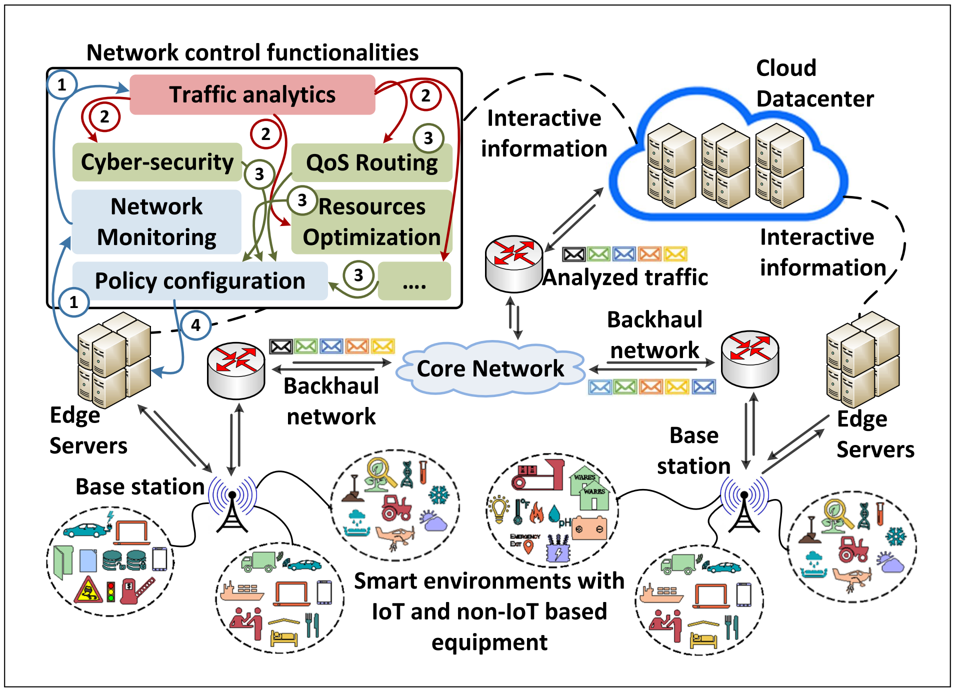


Figure 6.1 Example of smart network architecture

While dynamic smart environments offer enormous potential in terms of convenience and business value in a variety of fields such as healthcare, manufacturing, education, and transportation (Al-Fuqaha *et al.*, 2015; Li *et al.*, 2018b); however, the increasing volume, variety, and velocity of the produced data raise different challenges for network administrators regarding the effective management of the network performance, including massive numbers of heterogeneous flows and constrained network resources.

Therefore, to enhance the automation of network operation and management activities (e.g., traffic engineering, resources allocation, Quality of Service (QoS) provisioning, cyber-security control, and intrusion detection), a deep understanding of the network traffic nature and characteristics is fundamental for operators, since it enables them to be fully aware of the network content. Furthermore, the design of data analytics mechanisms in a massive network environment differs according to the network management engines' purposes. For example, operators of smart cities and campuses need to identify heterogeneous services, particularly hungry bandwidth flows, such as broadcast and peer-to-peer traffic, that impact critical services in order to control the appropriate levels of reliability. As well, they need to profile traffic patterns and behaviors (e.g., a vast array of unexpected DNS response packets) to identify attacks and enhance network-level cyber-security mechanisms. In our work, we focus on analyzing traffic flow to associate it with network services (i.e., service differentiation). It is noteworthy that the terms traffic "identification" and "classification" are used interchangeably when traffic analysis is related to the detection of a specific type of traffic (e.g., video streaming). Furthermore, over the last decade, the research and industrial communities have investigated, designed, and developed a large body of traffic classification approaches for the non-IoT traffic category generated from general-purpose devices (e.g., phones and laptops). However, despite the worldwide prevalence of heterogeneous IoT devices and intelligent environments, there is still not a very good understanding of all network traffic natures, including IoT and non-IoT traces, in order to accurately profile and classify all network traffic patterns (Dainotti *et al.*, 2012).

The most common network traffic identification approaches can be decomposed into three main categories: port number inspection methods, payload inspection-based methods, and flow inspection-based methods. The first approach identifies network traffic using the transport layer port number while assuming that all applications use predictable well-known port numbers listed by the Internet Assigned Numbers Authority (IANA). Although this approach is simple to implement and fast to infer network flows, it is inaccurate and unreliable for many reasons. For example, certain network applications (e.g., P2P applications) may use non-standardized port numbers to disguise their traffic and overcome port-based filters or firewalls. Furthermore, some newly designed applications with no IANA registered ports use randomly well-known ports already assigned to other applications. Accordingly, as alternatives, Deep Packet Inspection (DPI) or payload-based approaches emerged that inspect the packet content using pattern or signature matching mechanisms (Finsterbusch *et al.*, 2013). Although these techniques are considered highly accurate for traffic identification, they face different challenges. First, governments may prohibit third parties to inspect payloads for security reasons (e.g., privacy violation). Second, they are easily circumvented by encryption, protocol obfuscation, and encapsulation, which diminish this technique's outstanding advantage (i.e., the high accuracy).

Furthermore, the classifiers must be aware of each network application's packet payload syntax, while an up-to-date list of application signatures requires a huge manual effort and incurs high computation and storage costs (Dainotti *et al.*, 2012; Taylor *et al.*, 2017).

The issues related to the port and payload-based inspection approaches have motivated the research and development community to find new recognition techniques that do not require packet examination. Thus, the flow-based inspection approach emerged, where a flow is defined as an unidirectional exchange of consecutive packets having the same quintuple: $\big($IP source, port source, IP destination, port destination, transport-level protocol$\big)$. This mechanism is based on the analysis of the statistical flow features, such as flow duration, inter-packet arrival times, and packet size per flow characterizing each specific application (e.g., online game session). Multiple sophisticated machine learning-based analytical mechanisms have been proposed to increase classification efficiency (Kim *et al.*, 2008; Nguyen & Armitage, 2008), whereas the performance of each traffic classifier in terms of accuracy, speed, and resource consumption depends not only on the differences among algorithms and their specific configuration but also on the selection of the most crucial features for classification (Liao *et al.*, 2020).

This work presents a technical survey (e.g., Finsterbusch *et al.* (2013); Guck *et al.* (2018)) about flow-based inspection methods. It offers the research and development community a machine and deep learning-based extensible comprehensive evaluation framework to serve as a baseline for the design of lightweight engines for fine-grained IoT and non-IoT service differentiation in smart environments while supporting end-user privacy and the evolving nature of traffic. We inspect IoT and non-IoT traffic traces to consider the important statistical features for optimally classifying heterogeneous smart network services. We review different algorithms from supervised and unsupervised machine learning models to deep learning-based models while evaluating the accuracy performance and assessing the spatial and temporal complexity of each classification mechanism. Our proposed framework differs from the current published works for three main reasons. First, we do not focus only on the identification of IoT-based demands or only on the identification of the general-based demands, but on both, since they coexist and share the same network resources in smart environments. Second, through these two main classes, we fill the gap in the literature, and we perform fine-grained analytics for identifying specific network services (e.g., video streaming, smart TP-Link Cam, and smoke detector), which is highly important for the deployment of different network control functionalities (e.g., resources allocation and QoS provisioning). Third, to contribute to the design of more appropriate classifiers, we investigate not only one particular approach (e.g., random forest and naive Bayes in (Sivanathan *et al.*, 2018) or SVM and unsupervised k-means in (Fan & Liu, 2017)), but various machine learning and deep learning-based classifiers. Furthermore, we provide extensive comparisons and recommendations that can serve as a strong foundation for selecting and deploying further advanced classification mechanisms. The major contributions of this work can be summarized as follows:

1. **Feature engineering:** We present a feature importance review that determines the best minimal set of characteristics for classifying IoT and non-IoT network traffic flows. The produced sets of features consider the type of network application, early-stage classification, and the correlation between features while filtering irrelevant and redundant attributes, thereby preserving the classification accuracy and decreasing the computational complexity and data storage requirements.
2. **Classification engines:** We carry out rigorous evaluations and comparisons between different machine learning and deep learning-based techniques. In particular, while deep learning deployment is still thorny and at present less approved in network traffic classification, we justify the deep learning high complexity and training exigencies in heterogeneous intelligent environments. The proposed framework serves as a reliable map for selecting and designing the best algorithm for fine-grained IoT and non-IoT services differentiation.
3. **Performance evaluation:** We evaluate the proposed framework from different aspects, including the classification analytical accuracy through confusion matrix and the time and space complexity of the different models. The experimental results show that the random forest approach achieves the highest accuracy for IoT and non-IoT application identification. However, it is highly dependent on the selected set of features. On the other hand, while the

deep learning approach is effective, it eliminates the heavy feature selection process and enables automatic important-feature selection.

The rest of this paper is organized as follows: Section 6.2 reviews relevant related works. Section 6.3 presents the proposed extensible evaluation framework, including IoT and non-IoT heterogeneous traffic characteristics, early-stage identification, and the adopted machine learning and deep learning-based classification engines. Section 6.4 presents the performance evaluation criteria, notably the confusion matrix. Section 6.5 presents the set of traffic traces adopted in this work. Section 6.6 presents the performance evaluation through different scenarios with rigorous comparisons and recommendations. Finally, Section 6.7 presents conclusions and future directions.

6.2 Related work

Various machine-learning-based approaches have been proposed in the literature for traffic classification. Sivanathan *et al.* (2018) proposed a 2-stage machine-learning model using random forest and naive Bayes algorithms for real-time IoT devices classification. They considered different flow features, such as port numbers, domain names, cipher suites, and flow volumes. The proposed model achieved over 99% accuracy in an environment of 28 IoT devices. However, the authors focused mainly on distinguishing between IoT and non-IoT traces without considering the specific service recognition that is highly important in many network management and prioritization operations.

Yamansavascilar *et al.* (2017) evaluated four machine-learning methods (J48, random forest, k-NN, and Bayesian network) to identify 14 different non-IoT services. They selected 12 attributes, including the flow duration, the minimum, maximum, and average statistics of the packet, in addition to the window size, from a set of 111 flow-based features. They focused on the techniques' accuracy performance and showed that the k-NN (k = 1) algorithm provided the most accurate result with 93.94%, while the second most accurate algorithm was the random forest technique with 90.87% accuracy. This work presents an important basis from the idea side; nevertheless, the authors addressed only general non-IoT applications. Furthermore, by considering the flow duration attribute as one of the critical features, they did not address the early-stage processing to ensure real-time classification and reduce resource consumption. Considering only the accuracy metric is not enough to evaluate the performance of different machine learning classification techniques.

Fan & Liu (2017) addressed the general non-IoT application identification using machine learning techniques while focusing mainly on supervised Support Vector Machine and unsupervised k-means clustering algorithms. They investigated how model adjustment and feature selection affect the performance of the classification techniques. Experimental results indicated that an overall accuracy of over 95% could be achieved with SVM, even with a small portion of the dataset for training. Although the accuracy of the k-means classifier is highly affected by the size of the dataset, it enables characterizing new or unknown application types. Sun *et al.* (2018) adopted SVM as the baseline method in traffic classification. While supporting the high accuracy of this technique, they addressed two main limitations, including the high training cost in terms of memory and CPU and the inability to support continuous learning. The experimental results proved the effectiveness of the proposed models in traffic classification.

Moore & Zuev (2005) used the naive Bayes technique for service identification with ten general non-IoT applications, e.g., bulk data transfer, database, interactive, mail, services, www, P2P, attack, games, and multimedia. Then, they extended the work with the application of the Bayesian neural network approach in (Auld *et al.*, 2007). They proved the accuracy enhancement compared to the naive Bayes technique and showed the high accuracy potential of the proposed model up to 98%. Furthermore, they used a list of 20 features from a set of 246 full-flow-based statistical features with their descriptions and importance ranking. However, many of these selected features, such as the effective bandwidth based upon entropy and the Fourier transform of packet inter-arrival times, are computationally challenging versus the accuracy detriment.

Lopez-Martin *et al.* (2017) used deep learning to conduct the IoT traffic classification. They explored the recurrent neural network (RNN), the convolutional neural network (CNN), and a combination of CNN and RNN models. They considered six features, including source and destination port numbers, number of bytes in the packet payload, TCP window size, inter-arrival time, and direction of the packet extracted from the first 20 packets of a flow. The designed combined model demonstrates good performance, attaining an accuracy of 96.32%. All the works discussed above show the non-convergent approaches deployed for addressing the same issue, i.e., traffic analysis and service differentiation. In this work, we conduct rigorous evaluations and comparisons through various machine learning and deep learning-based algorithms with different sets of features to offer a map for selecting and designing optimal engines for fine-grained IoT and non-IoT service identification.

6.3 An extensible comprehensive performance evaluation framework

In this section, we describe our extensible evaluation framework for analyzing traffic in intelligent environments. The target object to be classified is an unidirectional IP flow consisting of a series of packets sharing common quintuple characteristics: (IP source, IP destination, port source, port destination, transport-level protocol). As shown in Figure 6.2, the first part of the statistical-based analysis framework focuses on the data pre-processing mechanisms, including, for instance, the monitoring of traffic traces and the making of datasets with ground truths. Then, the rest of the steps are different depending on the basic adopted technique (i.e., either ML or DL approaches). For example, the standard machine learning-based classifiers rely on domain-expert driven handcrafted features within the feature engineering processes; then, the architecture involves the traffic classification engines. Meanwhile, the deep learning-based methods provide an end-to-end multilayer neural network model that eliminates the need for manual feature engineering by automatically learning important features from the raw input data and executes classification conjointly (Al-Garadi *et al.*, 2020). Consequently, all the classifiers' outputs are used for performance evaluation.

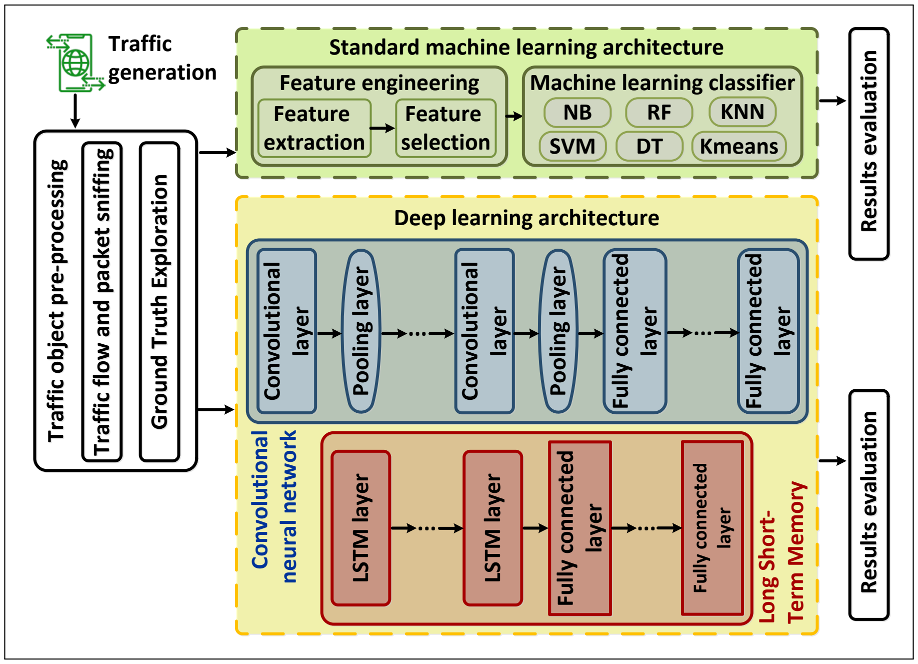


Figure 6.2 Machine learning and deep learning architectures

6.3.1 A review of flow-based feature engineering

Feature engineering is one of the crucial steps in the flow-based analysis mechanisms. It presents the process of careful feature creation through feature selection and extraction tasks. A feature is a characteristic that describes a distinctive property of a flow (e.g., flow duration, number of packets in a flow, and packet inter-arrival time). Feature engineering selects the most important features while removing unnecessary features that are irrelevant or redundant according to the requirements of the classification problem. Therefore, by finding the smallest needful subset of features characterizing heterogeneous traffic flows, the feature engineering process increases classification accuracy, prevents overfitting, reduces the dimensionality of voluminous datasets, decreases computational complexity, and reduces the required time for learning (Fahad *et al.*, 2013).

Identifying particular traffic classes or applications (e.g., VoIP, Skype, and Amazon Echo) requires discerning specific features. Therefore, feature engineering must contend with application software changes, including those designed to preclude classification. Table 6.1 represents

Table 6.1 Examples of network traffic statistical features

Flow-level features
Flow duration statistics (start time, end time), flow volume in bytes and packet count, etc.
Packet-level features
Packet inter-arrival time, Packet size, Packet payload size, Packet header size (Examples of statistics: min, max, mean, variance, standard deviation, median, first & third quartiles, root mean square, time series, and fourier transform, etc.)
Connection-level features
Number of TCP packets with specific fields set in the header (e.g., FIN, RSTS, PUSH, ACK, URG, SYN, CWE, ECE, ToS), TCP window size (all these statistics are set to zero for UDP packets), and source and destination port.

examples of the most common flow features enabling traffic identification. As shown in this table, network traffic features could be categorized into three networking levels: flow-level features, packet-level features, and connection-level features. The flow-level properties present a set of global per-flow characteristics, such as flow duration, number of packets per flow, total transferred bytes per flow, etc. The packet-level properties present more granular statistics characterizing packets inside the same flow, such as packet size and inter-packet arrival times (e.g., mean, root mean square, and variance). On the other hand, the connection-level properties present the interactions-oriented details from the transport layer between network entities (e.g., hosts or endpoints) during an interval of observation. For example, a P2P application running on a given host uses dynamic port numbers to disguise its traffic and circumvent filtering while communicating with its peers. Hence, the number of distinct ports, which is almost equal to the number of distinct IP addresses of the peers, could be one of the connection-level discriminators used for identifying P2P traffic (Boutaba *et al.*, 2018).

Table 6.2 provides a summary of the most important network traffic flow features for service differentiation. These optimum features are selected through a genetic algorithm from a set of 249 flow-based features provided with a full description in (Moore *et al.*, 2013) and (Sivanathan *et al.*, 2018). The underlying principle of the genetic algorithm starts with the initial whole feature space, from which this stochastic approach generates random initial solutions of selected features, called as population within a generation. Then, the best solutions that have the best classification accuracy are combined and mutated in order to produce a new population pushed

Table 6.2 Selected flow-based features

List of selected features
* Port number
* Packets size, packet payload size, and packet inter-arrival time related statistics in a flow (i.e., maximum, minimum, 1st and 3rd quartiles, mean, variance, and standard deviation)
* Initial window size in terms of bytes for TCP samples, while this attribute is set to zero for UDP flow.
* Statistics related to packets with PUSH and SACK bits set in the TCP header, while these attributes are set to zero for UDP packets.
Extra features for IoT traffic
* Sleep time, calculated as the accumulation of all periods (e.g., 2 seconds or greater) when no packet is sent.
* Active time, calculated as the ongoing time throughout consecutive packets.
* Number of servers, most frequent port number, NTP interval time, DNS requests, and DNS interval.

into the next generation. This process is iteratively repeated until convergence to the best generation (i.e., the optimal minimal set of features) that includes only important features and removes redundant and irrelevant ones. It is noteworthy that this genetic algorithm is deployed in conjunction only with ML-based methods since the deep learning mechanisms enable an automatic learning process. It should be mentioned that the flow-level features (e.g., flow duration, the total number of packets, and the total amount of bytes in a flow) also consist important features for service differentiation. However, we do not consider them, since we deploy the sub-flow-based processing approach for early service detection.

Sub-flow-based processing for early detection: To ensure real-time classification, particularly within an online session, some features (e.g., mean flow duration and total transferred bytes) are not suitable because they could be extracted only by the end of a flow. Therefore, a small number of the most recent packets of a complete ongoing flow, called sub-flows, need to be used to extract efficient features. This sub-flow-based processing mechanism presents another form of data reduction within network traffic flow analysis because it does not consider the whole flow processing but only a few early packets of a flow. Hence, it enables many benefits, such as low feature extraction complexity, low-latency analysis (particularly during online sessions since classification can occur early with the starting of each flow), low memory requirements, and low computational complexity. Different existing works have studied the adequate numbers for early-stage traffic identification. Some studies showed that the first 20 packets are enough for an effective process, while others proved that the first 10 packets are enough, and still others confirmed that five-to-seven packets are very effective (Boutaba *et al.*, 2018).

6.3.2 Machine learning and deep learning classification engines

Statistical-based traffic classification engines have proven to be effective, particularly with encrypted traffic that prevents payload inspection. The performance of such classifiers depends not only on the feature engineering efficiency but also on the differences among statistical-based algorithms (e.g., decision tree, neural network, Bayesian techniques, etc.) and their specific configuration. Thus, we deploy a set of engines from supervised and unsupervised machine learning techniques to deep learning-based models. We select the most common and popular methods in the state-of-the-art for network traffic classification (Bakker *et al.*, 2019; Mohammadi *et al.*, 2018; Pacheco *et al.*, 2019; Shafiq *et al.*, 2020). A brief summarization of each technique with a mathematical model is presented in this section, wherein their various implementations offer a reliable map for designing and deploying the best algorithm for fine-grained IoT and non-IoT services differentiation. Furthermore, while the deployment of deep learning approaches is still thorny and at present less approved in network traffic classification, we justify the high complexity and training exigencies of these approaches in smart environments (Mohammadi *et al.*, 2018).

6.3.2.1 Support vector machine (SVM) classifier

The SVM is a supervised machine learning technique that creates a separating hyperplane or a group of hyperplanes in the feature space between two or more classes. The key process for an optimal SVM-based classification is the production of decision boundaries (i.e., the separating hyperplanes) that maximize margins between the different classes. The margin is the distance between the hyperplane and the most adjacent data samples of each class, which are defined as support vectors. Suppose that the training data are linearly-separable in the feature space $\phi(x)$, i.e., as illustrated in Figure 6.3, the algorithm's objective function and constraints are shown in (6.1), where w and b that are defining the hyperplane represent, respectively, the weight vector

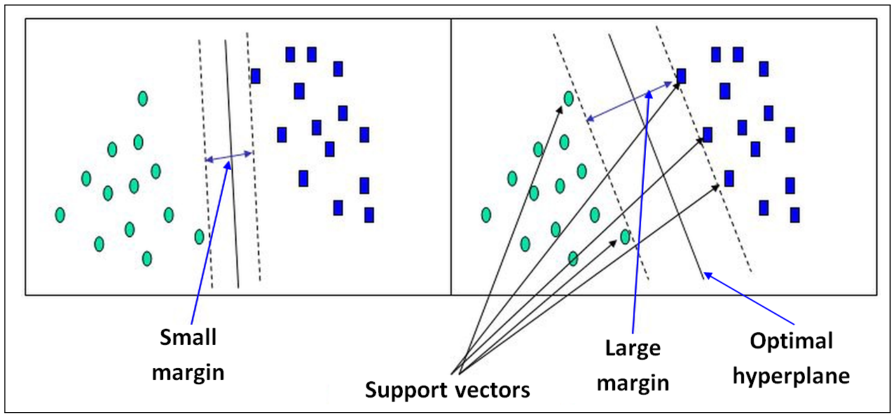


Figure 6.3 Support vector machine (SVM) classifier

and a scalar-valued bias, y_i is the corresponding output, which is the class, and n designates the number of training samples.

$$\begin{aligned} &\min_{\omega} \quad \frac{1}{2} \|\, \omega \,\|^2 \\ &\text{s.t.} \quad y_i\left(\omega^T \phi(x_i) + b\right) \geq 1, \; i = 1, ..., n \end{aligned} \tag{6.1}$$

When the data are not linearly separable, several Kernel methods (e.g., RBF, Polynomial, and Sigmoid) are used to model the non-linear decision boundaries and find the optimal solution between the different classes.

6.3.2.2 Bayesian theorem-based classifier

The Bayesian theorem, which states (6.2), determines the conditional probability of an occurring event (e.g., T, traffic type detection) based on prior information (e.g., c_i, features) related to an event that has already occurred (e.g., c). The naive Bayes classifier is a supervised machine learning algorithm, implementing the Bayesian theorem while maximizing (6.2) and assuming that all features are class-conditionally independent (i.e., there is no correlation or relationship between features). Hence, it is called "naive". It is one of the simplest, fastest, and most robust to irrelevant features algorithms.

$$P(c_i|T) = \frac{P(T, c_i)}{P(T)} = \frac{P(c_i)P(T|c_i)}{P(T)}, \quad \forall\, i = 1...N \tag{6.2}$$

6.3.2.3 k-nearest neighbor (k-NN) classifier

The k-NN classifier is a simple supervised machine learning technique that classifies a new instance based on the classes of the selected number (i.e., k) of its nearest neighbors. The k-NN building model starts by fixing the number of neighbors to be analyzed (i.e., k). Then, according to the unlabeled sample, it determines the k-nearest neighbors using distance functions (e.g., Euclidean), as shown in (6.3), where d denotes the distance between two instances, X_1 and X_2 represent the two corresponding feature vectors of the samples, and n is the number of features used in the model. Finally, k-NN determines the class of the unknown sample by the majority votes of its nearest neighbors. It is noteworthy that k presents a critical parameter, since the performance results (e.g., accuracy and processing time) are related to the analyzed neighbors.

$$\begin{aligned} d &= \sqrt{(X_1 - X_2)^T (X_1 - X_2)} \\ X_1 &= \left(x_{11}, x_{12}, ..., x_{1n}\right) \\ X_2 &= \left(x_{21}, x_{22}, ..., x_{2n}\right) \end{aligned} \tag{6.3}$$

6.3.2.4 Decision tree classifier

The decision tree is a supervised machine learning algorithm. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules for features' conjunction, and each leaf node represents the outcome (i.e., the final prediction with the class label). Hence, the classification of a sample proceeds from the root node to a suitable end leaf node with respect to the feature values. For example in (6.4), X indicates the input vector of a known number of features, Y indicates the output, and S indicates the training

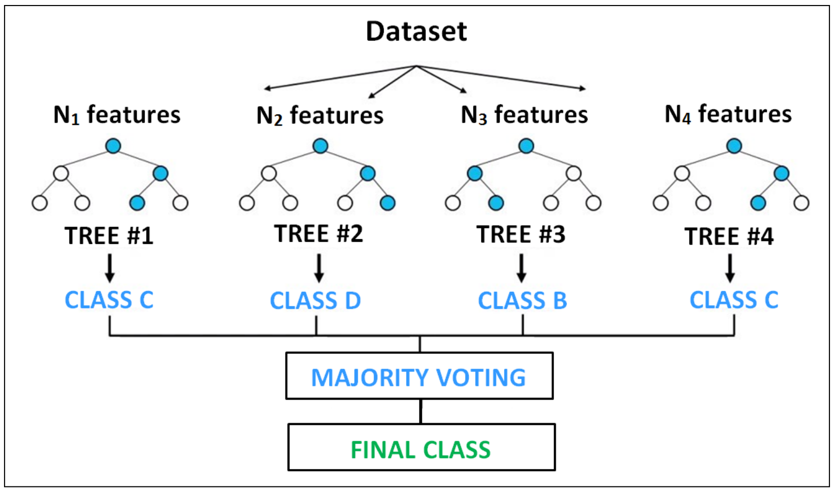


Figure 6.4 Random forest classifier

set with the couples of input features and their output. Equation (6.5) defines the purpose of the model, which is expected to predict the value of Y using the input X by applying the function p, which is representative of the tree. The building of the decision tree during the training process is based on several measures (e.g., Gini index, entropy, and information gain) for splitting nodes and identifying the best ones containing the most inference information.

$$\begin{aligned} X &= (x_1, x_2, ..., x_n)^T \\ S &= \{(X_1, Y_1), (X_2, Y_2), ..., (X_m, Y_m)\} \end{aligned} \tag{6.4}$$

$$Y = p(X) \tag{6.5}$$

6.3.2.5 Random forest classifier

The random forest is a supervised machine learning technique that comprises multiple decision tree classifiers created from the training set, as shown in Figure 6.4. It is based on the ensemble learning concept to improve the overall process performance through a precise and robust classification. Each individual tree is constructed by choosing randomly a subset of the feature space and trained to vote for a class. Then, by combining their outputs (i.e., based on the majority classification votes of all the trees), the random forest classifier infers the final classification result.

6.3.2.6 K-means clustering classifier

The K-means is an unsupervised machine learning technique. It enables solving a clustering problem by grouping unlabeled data into different clusters. For the deployment of this technique, two parameters are required, the dataset (e.g., N input data vectors $\{y_i\}_{i=1}^{N}$) and the number of clusters to be generated by the algorithm (e.g., K). The clusters are built using a centroid-based mechanism. Thus, it starts by randomly selecting K nodes from the unlabeled data wherein each node is initialized as a centroid for a cluster (e.g., c_i). Then, every unlabeled instance y_i in the dataset is assigned to one of the K clusters using the nearest distance (e.g., Manhattan, Minkowski, or Euclidean) compared to the centroid. For example, the labeling is performed through the following binary matrix L, as shown in (6.6), while the objective function for K-means clustering is defined in (6.7).

$$L_{i,j} = \begin{cases} 1 & \text{if data point } i \text{ belongs to cluster } j, \\ 0 & \text{otherwise.} \end{cases} \tag{6.6}$$

$$E(c, L) = \sum_{i,j} L_{i,j} \mid\mid y_i - c_j \mid\mid^2 \tag{6.7}$$

After all unlabeled data samples are assigned to a specific cluster, the clusters and their corresponding centroids are iteratively recalculated and updated until stable cluster convergence is achieved (i.e., no sample that can change the clusters exists).

6.3.2.7 Convolutional neural network (CNN)-based classifier

A CNN is a category of deep learning methods. It is based on a multi-layer feed-forward neural network approach. The data gets into the CNN through the input layer (e.g., feature-based image) and passes through various hidden layers before getting to the output layer producing the final result. The CNN hidden layers generally consists of three types of layers: convolution layers, pooling layers, and fully connected layers. The convolutional layers perform convolution operations for scanning and filtering raw input data with multiple local kernel filters in order

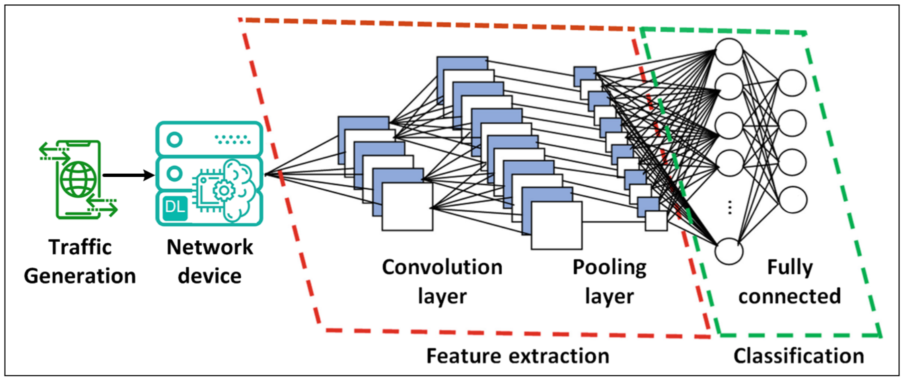


Figure 6.5 Convolutional neural network (CNN)-based system

to detect localized features in certain input divisions of the data. The pooling layers, typically following convolutional layers, perform down-sampling operations to reduce the feature size through max-pooling or average-pooling while keeping the invariance of the data. Thus, after performing feature learning via convolutional and pooling layers, the fully connected layers in a neural network conjointly fulfill the classification process, as shown in Figure 6.5.

Assuming a CNN model has a total of N layers, including the input, hidden, and output layers, Equation (6.8) shows the basic mathematical structure of a CNN model connecting between the layers, where $a^{(1)}$, $a^{(l)}$, and $a^{(l+1)}$ denotes the starting input vector, the input vector to level l, and the input vector to level $(l+1)$, respectively. Meanwhile, $z^{(l+1)}$ is the output vector of the neurons at level l, W defines the weight matrix, b is the bias matrix, and γ defines the activation function (e.g., ReLU function) that selects the output to be sent to the next layer of neurons as input. This process is iteratively repeated until the final layer N is reached and a class is chosen (i.e., the final output of the model).

$$\begin{aligned} z^{(l+1)} &= W^{(l)}a^{(l)} + b^{(l)} \\ a^{(l+1)} &= \gamma(z^{(l+1)}) \end{aligned} \tag{6.8}$$

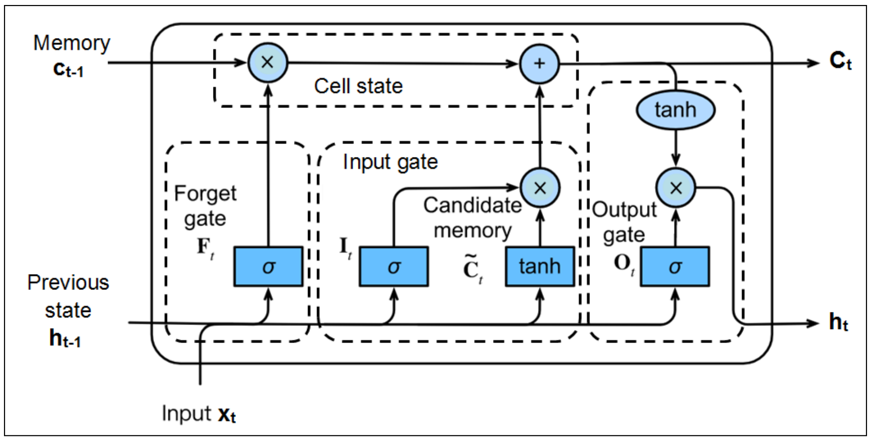


Figure 6.6 Long short-term memory (LSTM) cell diagram

6.3.2.8 Recurrent neural network (RNN)-based classifier

The RNN is a vital category of deep learning methods. It presents a sequential data-based neural network approach that uses previous outputs as inputs for the current inference. Hence, in addition to the processing of the current state, it reprocesses the outputs of the previous state, including the errors through loops, temporal layers, and internal memory to learn in a recurrent strategy. One of its most popular implementations is the Long Short-Term Memory (LSTM) variant, which solves the short-term memory problem of the standard RNN mechanism. It enables the preservation of information in memory cells for long periods of time using a set of gates that can learn which data in a sequence are important to keep and which to throw away (i.e., when information enters the memory cell, when it is output, and when it is forgotten), as shown in Figure 6.6 (Shibl *et al.*, 2020; Thapa & Duraipandian, 2021).

Hence, the forget gate controls the information to be removed (i.e., forgotten) in the corresponding LSTM cell. It uses an activation function σ applied to the weighted values of the current input x_t and the previous cell's output h_{t-1} that is re-fed into the network, in addition to a bias value, as illustrated in (6.9). The subscript f indicates that the values belong to the forget gate. The σ function provides a binary form by multiplying the produced vector with 0 or 1. Hence, if the forget gate's output is 0, the information is reset; otherwise, the information is kept in the cell state.

$$F_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{6.9}$$

The input gate controls the memory cell updates. First, the current input x_t and the previous cell's output h_{t-1} are passed into an activation function σ to filter the information to be retained, as shown in (6.10), where the subscript i indicates that the values belong to the input gate. On the other hand, the $tanh$ activation function, which has a range of [-1, 1], is also applied to the current input x_t and the previous cell's output h_{t-1} to generate the candidate memory cell, as shown in (6.11). Both of the produced vectors are multiplied to obtain the input gate's final value that is expressed as shown in (6.12) and Figure 6.6.

$$I_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{6.10}$$

$$\tilde{C}_t = tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{6.11}$$

$$I_t \circ \tilde{C}_t \tag{6.12}$$

Finally, the output gate is made from the multiplication of the outputs of another $tanh$ function applied to the generated vector of the current cell state C_t and a $sigmoid$ function applied to the weighted values of the current input x_t and the previous cell's output h_{t-1}, in addition to a bias value, as defined in (6.13) and (6.14), respectively. Hence, (6.15) enables deciding the final output value of the model.

$$C_t = F_t \circ c_{t-1} + I_t \circ \tilde{C}_t \tag{6.13}$$

$$O_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{6.14}$$

$$h_t = O_t \circ tanh(C_t) \tag{6.15}$$

6.4 Performance evaluation criteria

In this work, the performance evaluations of the different classification engines in terms of correctness and computational cost are performed through different rigorous experiments. To

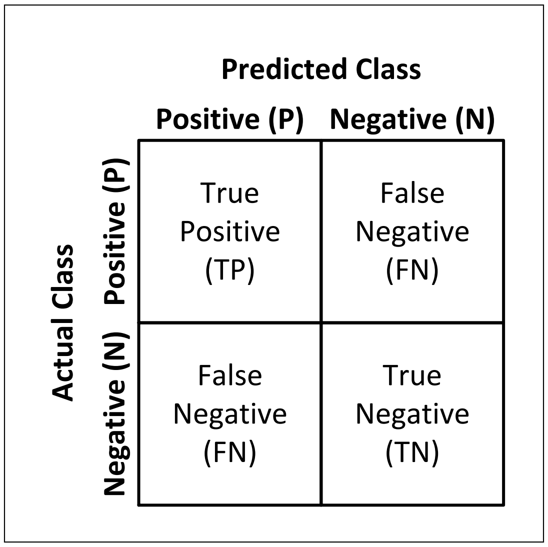


Figure 6.7 Confusion matrix for binary classification

evaluate classification correctness, we adopt the confusion matrix as a basic tool to provide direct visualization, analysis, and comparison of the different correctness measurements (Boutaba *et al.*, 2018; Finsterbusch *et al.*, 2013). Figure 6.7 depicts the confusion matrix for a binary classification. The matrix's rows refer to the actual classes, while the columns refer to the classified classes. The true positive (TP) presents the number of positive instances that are correctly classified; the false positive (FP) presents the number of negative instances that are incorrectly classified as positive samples; the true negative (TN) presents the number of negative instances that are correctly classified; and the false negative (FN) presents the number of positive instances that are incorrectly classified as negative samples.

From the confusion matrix, different evaluation metrics (e.g., accuracy, recall, precision, specificity, and F-measure) that enable achieving a deeper understanding of the classifier's performance could be concluded, where each metric is expressed as a function of the confusion matrix (i.e. TP, TN, FP, and FN), as follows:

- The overall accuracy metric is defined as the total number of all properly classified flows in all the overall classified instances.

$$Overall - Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{6.16}$$

- The recall (Rc) metric presents a per-class measure and is defined as the number of correctly classified flows of a specific class divided by the total number of flows belonging to that class. It should be noted that sensitivity and recall are the same metrics in the traffic classification technique.

$$Rc = \frac{TP}{TP + FN} \tag{6.17}$$

- The precision (Pr) metric presents a per-class measure and is defined as the ratio of the number of properly classified flows of a class over the total number of flows assigned to that class.

$$Pr = \frac{TP}{TP + FP} \tag{6.18}$$

- The F-measure (F_1) allows analyzing the trade-off between recall and precision by providing the harmonic average of those metrics, ideally 1; and it is calculated as follow:

$$F_1 = \frac{2 \,.\, Rc \,.\, Pr}{Rc + Pr} \tag{6.19}$$

6.5 Datasets for traffic classification

The appropriate network data presents a crucial factor for understanding network traffic and designing analytics engines. However, one of the main obstacles within network data analytics is the lack of sharable traces to serve as test data and ground truth (i.e., annotated flow objects used as reference) for validation due to many reasons, such as the end-user privacy concern that prevents unauthorized people from inspecting the contents of the communications.

Network traffic flow can be traced through networking devices (e.g., router, switch, and gateway) using different libraries (e.g., Tcpdump, WinDump, LibPcap, and WinPcap) and stored as captures of pcap files on external hard drives. In this work, we use three famous datasets that have already been used in different research works. Two datasets include general non-IoT traces, and the third dataset includes IoT traffic traces. Using different databases enables not only analyzing as many as possible of heterogeneous IoT and non-IoT services but also evaluating the stability performance of the classifiers. Table 6.3 shows the statistics of selected classes from the different datasets considered in this work. As mentioned above, we are considering a network flow as an unidirectional sequence of packets that share the same quintuple: (IP source, port source, IP destination, port destination, transport-level protocol). Each raw trace in the three used datasets contains packet headers and payload information providing researchers with rich data and corresponding features for different aspects of network traffic investigation.

6.5.1 UNIBS Italy, traffic traces for Non-IoT-based applications

UNIBS is an openly available traffic dataset with associated ground truths developed by Prof. F. Gringoli and his research team (UNIBS, Accessed: October 20, 2020). The traffic traces were generated by twenty workstations and collected using the Tcpdump library on the edge router of the campus network of the University of Brescia in Italy during three consecutive working days (Sept 30, Oct 1, and Oct 2, 2009). The resulting traces were stored as capture files (.pcap) on a dedicated hard drive connected to the router. The whole database is composed mainly of TCP (99%) and UDP traffic. As shown in Table 6.3, it includes the main Internet applications, such as Web, Mail (POP3, IMAP4, SMTP), Skype, and Peer-to-Peer (Bittorrent and Edonkey) protocols. The total instance and byte distributions of each application are listed in Table 6.3.

6.5.2 UPC Spain, traffic traces for Non-IoT-based applications

The Universitat Politécnica de Catalunya (UPC), Barcelona, Spain, provides a large set of published traffic traces with associated ground truths, available in (UPC, Accessed: October 20, 2020). The dataset contains over 35GB of TCP and UDP traffic with a full packet payload. In addition, it includes several types of services, such as Web browsers (e.g., Chrome, Internet Explorer, and Firefox), Peer-to-Peer (Bittorrent and Edonkey), FTP (FileZilla, CuteFTP, and

Table 6.3 Traffic traces

UNIBS traces			UPC traces			UNSW traces		
Flow classes	**No. of flows**	**No. of MB**	**Flow classes**	**No. of flows**	**No. of MB**	**Flow classes**	**No. of flows**	**No. of MB**
Web (http)	25729	107.3423	**Web (http)**	47096	8783.89	**Amazon Echo**	38779	990.03
Mail (imap)	327	0.860226	**FTP**	876	3089.06	**iHome**	57789	409.52
Mail (pop3)	2473	4.292419	**RDP**	132907	13218	**Belkin motion**	59603	1654.25
Bittorrent	3571	6.393487	**Bittorrent**	62845	2621.37	**LiFX**	36251	166.69
Edonkey	379	0.241587	**Edonkey**	176581	2823.88	**Netatmo cam**	24091	2104.40
Skype	801	0.805453	**NTP**	27786	4.03	**Hue bulb**	27738	825.94
Mail (smtp)	120	0.043566	**RTMP**	427	5907.15	**Samsung cam**	59130	2054.35

WinSCP), Remote Desktop (built-in, xrdp), and other different protocols (e.g., DNS, NTP, NETBIOS). The information about network captures in terms of numbers of instances and total byte distributions is shown in Table 6.3.

6.5.3 UNSW Sydney, traffic traces for IoT-based applications

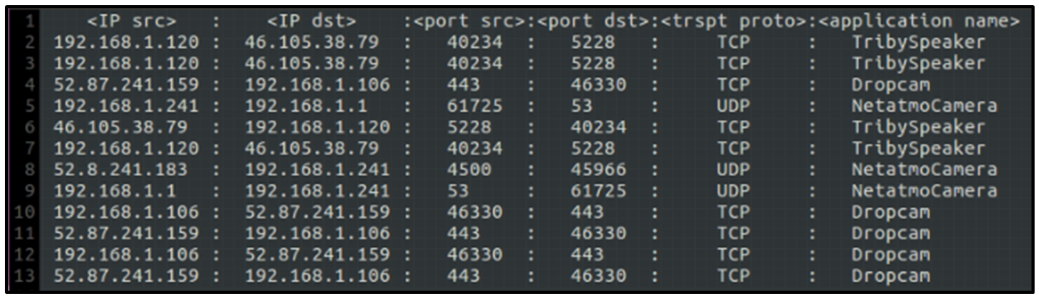
The UNSW is an openly available IoT-based dataset with associated ground truths developed by Prof. Hassan Habibi Gharakheili and his research team at the University of New South Wales (UNSW, Accessed: October 20, 2020). The smart environment was instrumented with 28 different IoT devices representing various categories, e.g., cameras, lights, plugs, motion sensors, air quality sensors, smoke sensors, appliances, and health monitors. IoT traffic traces were collected using the Tcpdump library running on the OpenWrt operating system and stored as pcap files on an external hard drive of 1 TB storage attached to the gateway. The traces collecting process was performed during six months with the size of the daily logs varying between 61 MB and 2 GB, with an average of 365 MB, and only a subset of which was released as open for the research community. The information about the considered services from the UNSW dataset in terms of numbers of flows and total byte distributions is shown in Table 6.3.

6.6 Experimental results

6.6.1 Experimental environment

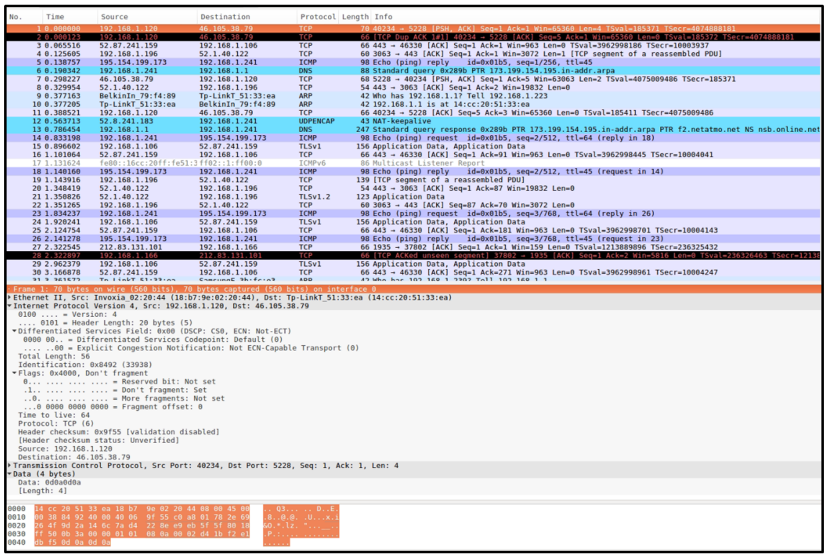
All experiments were carried out on a Linux PC running Ubuntu 16.04 with Intel Core i7 and 190GB of memory. Regarding datasets, as detailed in Section 6.5, we adopt three databases representing their IoT and non-IoT services through significant numbers of samples with standard flow object definitions, as shown in Figure 6.8. To perform online traffic classification, we use the PlayCap tool (Softpedia, Accessed: October 20, 2020), which takes as inputs a pcap file and a network interface from which the traffic will be captured. It loads the traces and regenerates them packet-by-packet through the specified interface while respecting the order and the inter-arrival time between packets. We use Python's Pcapy library (Coresecurity, Accessed: October 20, 2020) to read and investigate the traffic captures (an example is given in Figure 6.9). As mentioned in previous sections, we consider a flow as an unidirectional exchange of consecutive packets having the same quintuple: $\big($IP source, port source, IP destination, port destination, transport-level protocol$\big)$.

Regarding the preprocessing operation, to manage a large amount of data, we need to create objects that can be rapidly recognized. Thus, we define structures in the forms of packets and flows, where we infer the corresponding features (e.g., average-packet-length), as shown in Figures 6.10 to 6.12. Furthermore, we convert the features' strings into hexadecimal numbers while preserving all the string's information, i.e., each character is converted and stored with respect to its position in the corresponding object. These packet- and flow-based objects are suitable for machine learning techniques; nevertheless, they need some more adjustment to be suitable for the deep learning models, which were originally used for image recognition problems but have rarely been implemented in traffic classification. Hence, since the deep learning approach can select features by self-learning from the training data set, we built an (N×N) matrix using network traffic features $\big($e.g., 16×16 for 256 non-IoT features (Moore *et al.*, 2013)$\big)$. Then, each network flow's matrix of features is converted to a grayscale image that constitutes thereby the input layer of the deep learning model.



```
   <IP src>     :    <IP dst>     :<port src>:<port dst>:<trspt proto>:<application name>
192.168.1.120 : 46.105.38.79  :  40234   :   5228   :     TCP     :    TribySpeaker
192.168.1.120 : 46.105.38.79  :  40234   :   5228   :     TCP     :    TribySpeaker
52.87.241.159 : 192.168.1.106 :  443     :   46330  :     TCP     :    Dropcam
192.168.1.241 : 192.168.1.1   :  61725   :   53     :     UDP     :    NetatmoCamera
46.105.38.79  : 192.168.1.120 :  5228    :   40234  :     TCP     :    TribySpeaker
192.168.1.120 : 46.105.38.79  :  40234   :   5228   :     TCP     :    TribySpeaker
52.8.241.183  : 192.168.1.241 :  4500    :   45966  :     UDP     :    NetatmoCamera
192.168.1.1   : 192.168.1.241 :  53      :   61725  :     UDP     :    NetatmoCamera
192.168.1.106 : 52.87.241.159 :  46330   :   443    :     TCP     :    Dropcam
52.87.241.159 : 192.168.1.106 :  443     :   46330  :     TCP     :    Dropcam
192.168.1.106 : 52.87.241.159 :  46330   :   443    :     TCP     :    Dropcam
52.87.241.159 : 192.168.1.106 :  443     :   46330  :     TCP     :    Dropcam
```

Figure 6.8 Extract from the ground truth file of the UNSW traces



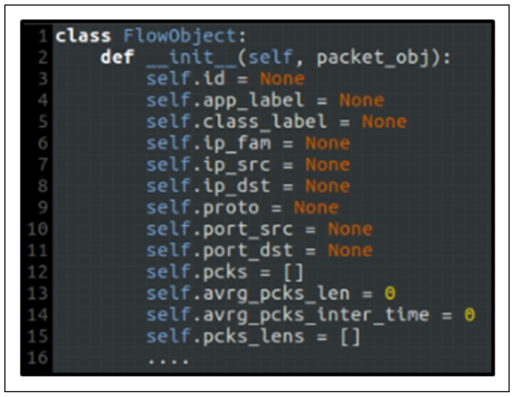
```
No.  Time      Source               Destination        Protocol  Length Info
   1 0.000000  192.168.1.120        46.105.38.79       TCP          70 40234 → 5228 [PSH, ACK] Seq=1 Ack=1 Win=65360 Len=4 TSval=185371 TSecr=4074888181
   2 0.000123  192.168.1.120        46.105.38.79       TCP          66 [TCP Dup ACK 1#1] 40234 → 5228 [ACK] Seq=5 Ack=1 Win=65360 Len=0 TSval=185372 TSecr=4074888181
   3 0.065516  52.87.241.159        192.168.1.106      TCP          66 443 → 46330 [ACK] Seq=1 Ack=1 Win=963 Len=0 TSval=3962998186 TSecr=10003937
   4 0.125605  192.168.1.196        52.1.40.122        TCP          60 3063 → 443 [ACK] Seq=1 Ack=1 Win=3072 Len=1 [TCP segment of a reassembled PDU]
   5 0.138757  195.154.199.173      192.168.1.241      ICMP         98 Echo (ping) reply    id=0x01b5, seq=1/256, ttl=45
   6 0.190342  192.168.1.241        192.168.1.1        DNS          88 Standard query 0x289b PTR 173.199.154.195.in-addr.arpa
   7 0.298227  46.105.38.79         192.168.1.120      TCP          68 5228 → 40234 [PSH, ACK] Seq=1 Ack=5 Win=63063 Len=2 TSval=4075009486 TSecr=185371
   8 0.329954  52.1.40.122          192.168.1.196      TCP          54 443 → 3063 [ACK] Seq=1 Ack=2 Win=19832 Len=0
   9 0.377163  BelkinIn_79:f4:89    Tp-LinkT_51:33:ea  ARP          42 Who has 192.168.1.1? Tell 192.168.1.223
  10 0.377205  Tp-LinkT_51:33:ea    BelkinIn_79:f4:89  ARP          42 192.168.1.1 is at 14:cc:20:51:33:ea
  11 0.388521  192.168.1.120        46.105.38.79       TCP          66 40234 → 5228 [ACK] Seq=5 Ack=3 Win=65360 Len=0 TSval=185411 TSecr=4075009486
  12 0.563713  52.8.241.183         192.168.1.241      UDPENCAP     43 NAT-keepalive
  13 0.786454  192.168.1.1          192.168.1.241      DNS         247 Standard query response 0x289b PTR 173.199.154.195.in-addr.arpa PTR f2.netatmo.net NS nsb.online.net
  14 0.833198  192.168.1.241        195.154.199.173    ICMP         98 Echo (ping) request  id=0x01b5, seq=2/512, ttl=64 (reply in 18)
  15 0.896602  192.168.1.106        52.87.241.159      TLSv1       156 Application Data, Application Data
  16 1.101064  52.87.241.159        192.168.1.106      TCP          66 443 → 46330 [ACK] Seq=1 Ack=91 Win=963 Len=0 TSval=3962998445 TSecr=10004041
  17 1.131624  fe80::16cc:20ff:fe51:3 ff02::1:ff00:0   ICMPv6       86 Multicast Listener Report
  18 1.140160  195.154.199.173      192.168.1.241      ICMP         98 Echo (ping) reply    id=0x01b5, seq=2/512, ttl=45 (request in 14)
  19 1.143916  192.168.1.196        52.1.40.122        TCP         139 [TCP segment of a reassembled PDU]
  20 1.348419  52.1.40.122          192.168.1.196      TCP          54 443 → 3063 [ACK] Seq=1 Ack=87 Win=19832 Len=0
  21 1.350826  52.1.40.122          192.168.1.196      TLSv1.2     123 Application Data
  22 1.351265  192.168.1.196        52.1.40.122        TCP          60 3063 → 443 [ACK] Seq=87 Ack=70 Win=3072 Len=0
  23 1.834237  192.168.1.241        195.154.199.173    ICMP         98 Echo (ping) request  id=0x01b5, seq=3/768, ttl=64 (reply in 26)
  24 1.920241  192.168.1.106        52.87.241.159      TLSv1       156 Application Data, Application Data
  25 2.124754  52.87.241.159        192.168.1.106      TCP          66 443 → 46330 [ACK] Seq=1 Ack=181 Win=963 Len=0 TSval=3962998701 TSecr=10004143
  26 2.141278  195.154.199.173      192.168.1.241      ICMP         98 Echo (ping) reply    id=0x01b5, seq=3/768, ttl=45 (request in 23)
  27 2.322545  212.83.131.101       192.168.1.166      TCP          66 1935 → 37802 [ACK] Seq=1 Ack=1 Win=159 Len=0 TSval=1213889896 TSecr=236325432
  28 2.322897  192.168.1.166        212.83.131.101     TCP          66 [TCP ACKed unseen segment] 37802 → 1935 [ACK] Seq=1 Ack=2 Win=5816 Len=0 TSval=236326463 TSecr=12138
  29 2.962379  192.168.1.106        52.87.241.159      TLSv1       156 Application Data, Application Data
  30 3.166878  52.87.241.159        192.168.1.106      TCP          66 443 → 46330 [ACK] Seq=1 Ack=271 Win=963 Len=0 TSval=3962998961 TSecr=10004247
  31 3.361572  Tp-LinkT_51:33:ea    SamsungE_3b:fc:e3  ARP          42 Who has 192.168.1.239? Tell 192.168.1.1

 Frame 1: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0
▸ Ethernet II, Src: Invoxia_02:20:44 (18:b7:9e:02:20:44), Dst: Tp-LinkT_51:33:ea (14:cc:20:51:33:ea)
▾ Internet Protocol Version 4, Src: 192.168.1.120, Dst: 46.105.38.79
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▾ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      0000 00.. = Differentiated Services Codepoint: Default (0)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 56
    Identification: 0x8492 (33938)
  ▾ Flags: 0x4000, Don't fragment
      0... .... .... .... = Reserved bit: Not set
      .1.. .... .... .... = Don't fragment: Set
      ..0. .... .... .... = More fragments: Not set
      ...0 0000 0000 0000 = Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x9f55 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.1.120
    Destination: 46.105.38.79
▸ Transmission Control Protocol, Src Port: 40234, Dst Port: 5228, Seq: 1, Ack: 1, Len: 4
▾ Data (4 bytes)
    Data: 0d0a0d0a
    [Length: 4]

0000  14 cc 20 51 33 ea 18 b7  9e 02 20 44 08 00 45 00   .. Q3... .. D..E.
0010  00 38 84 92 40 00 40 06  9f 55 c0 a8 01 78 2e 69   .8..@.@. .U...x.i
0020  26 4f 9d 2a 14 6c 7a d4  22 8e e9 eb 5f 5f 80 18   &O.*.lz. "...__..
0030  ff 50 0b 3a 00 00 01 01  08 0a 00 02 d4 1b f2 e1   .P.:.... ........
0040  db f5 0d 0a 0d 0a                                  ......
```

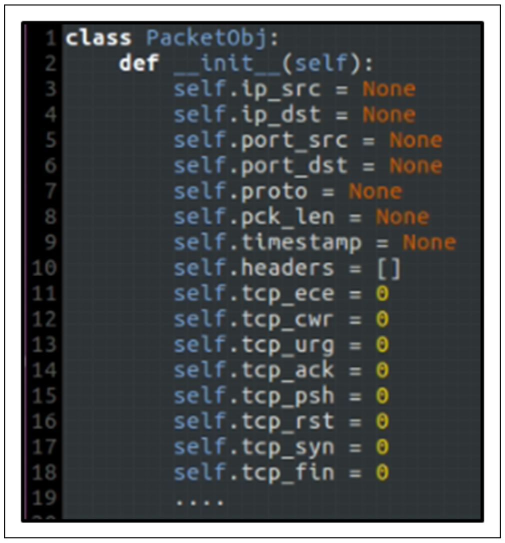
Figure 6.9 Wireshark capture from the UNSW pcap traces.

Moreover, as explained in Section 6.3.1, we perform early-stage traffic classification, i.e., we extract features only from the first few packets of a flow object without waiting until the end of the flow. This approach is very effective and requires less computational resources. Hence, we use only the first five packets, which proved to be enough (Boutaba *et al.*, 2018). During our experiments, we adopt the 10-fold cross-validation method to improve the classification correct rate. Thus, we randomly split each dataset into 10 folds while using 90% of the total samples for training and 10% for validation. We repeat this process 10 times until each of the 10-folds has served as the validation set. The average result of the cross-validation experiments serves as a performance metric for each analytical model. To avoid overfitting, we cautiously balance the



```python
class FlowObject:
    def __init__(self, packet_obj):
        self.id = None
        self.app_label = None
        self.class_label = None
        self.ip_fam = None
        self.ip_src = None
        self.ip_dst = None
        self.proto = None
        self.port_src = None
        self.port_dst = None
        self.pcks = []
        self.avrg_pcks_len = 0
        self.avrg_pcks_inter_time = 0
        self.pcks_lens = []
        ....
```

Figure 6.10 An example of capture for the flow object structure

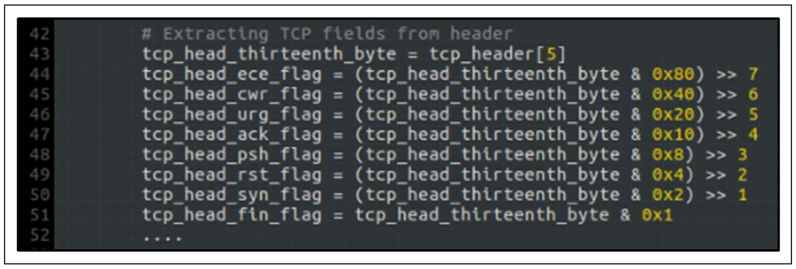


```python
class PacketObj:
    def __init__(self):
        self.ip_src = None
        self.ip_dst = None
        self.port_src = None
        self.port_dst = None
        self.proto = None
        self.pck_len = None
        self.timestamp = None
        self.headers = []
        self.tcp_ece = 0
        self.tcp_cwr = 0
        self.tcp_urg = 0
        self.tcp_ack = 0
        self.tcp_psh = 0
        self.tcp_rst = 0
        self.tcp_syn = 0
        self.tcp_fin = 0
        ....
```

Figure 6.11 An example of capture for the packet object structure

training datasets while considering that certain services require substantially more instances to be recognized.

Regarding engines, as detailed in Section 6.3.2, we consider eight different machine and deep learning classification techniques, including random forest, decision tree, naive Bayes, SVM, Kmeans, k-NN, CNN, and LSTM. We implement machine learning models using Python's Scikit-Learn library (Cournapeau, Accessed: October 20, 2020) and deep learning models with



```
        # Extracting TCP fields from header
        tcp_head_thirteenth_byte = tcp_header[5]
        tcp_head_ece_flag = (tcp_head_thirteenth_byte & 0x80) >> 7
        tcp_head_cwr_flag = (tcp_head_thirteenth_byte & 0x40) >> 6
        tcp_head_urg_flag = (tcp_head_thirteenth_byte & 0x20) >> 5
        tcp_head_ack_flag = (tcp_head_thirteenth_byte & 0x10) >> 4
        tcp_head_psh_flag = (tcp_head_thirteenth_byte & 0x8) >> 3
        tcp_head_rst_flag = (tcp_head_thirteenth_byte & 0x4) >> 2
        tcp_head_syn_flag = (tcp_head_thirteenth_byte & 0x2) >> 1
        tcp_head_fin_flag = tcp_head_thirteenth_byte & 0x1
        ....
```

Figure 6.12 An example of capture for feature extraction from a TCP header

the TensorFlow library (Google-Brain, Accessed: October 20, 2020) and its Keras APIs (Chollet, Accessed: October 20, 2020). Each of the models has various hyper-parameters. Therefore, we set up the following main parameters to configure them: We deploy the RBF kernel for the SVM classifier. For the k-NN classifier, we set *k* at 5 after testing different values between 3 and 40 to determine the optimal one that provides the best accuracy results. With K-means, we split each dataset into 8 clusters. For the random forest classifier, after testing different values, we set up 13 trees in the forest.

Regarding the deep learning models, from (Moore *et al.*, 2013) that includes 249 features and from (Sivanathan *et al.*, 2018), we manually select 60 features while eliminating the computationally challenging features (e.g., Fourier transform of packet inter-arrival times). The feature-based image's size is set to (8×8) that will be fed into the input layer of the models. We implement a CNN model with 2 convolutional layers, 32 filters, and (5×5) kernel size. On the other hand, we implement a multi-layer LSTM model with three layers, 128 output size, glorot uniform as kernel initializer, and uniform as a recurrent initializer. In addition, the softmax activation function, the adam optimizer, and the 100 batch size parameters are used with both of the models.

We carry out rigorous evaluations and comparisons through three different traffic classification scenarios applied to the adopted datasets and their corresponding selected features. Within each scenario, we compare the performance (e.g., accuracy and computational complexity) of the eight above-mentioned classification engines.

6.6.2 Result analysis

6.6.2.1 Classification accuracy

As explained in Section 6.4, to evaluate classifiers' performance in terms of accuracy, we examine the confusion matrix through different experiments. The Y-axis of each classifier's matrix presents the real services incorporated in our datasets, while the X-axis refers to the detected services of the specific classifier. The main diagonal refers to TP rates according to each actual service in the dataset. Excluding the value of the main diagonal, each row represents the FN rates classification results, while the columns represent the distribution of the FP rates of each identified service. All classification results are depicted in percentages with gradient colors from 0% to 100%. From each matrix, we can conclude the corresponding classifier's overall accuracy, recall, precision, and F-measure metrics, as presented in (6.16), (6.17), (6.18), and (6.19). The classification results of the different engines for the UNIBS, UPC, and UNSW datasets are given in Figures 6.13, 6.14, and 6.15, respectively. It is important to note that the classifier's decision can be either a service or "Unknown". The "Unknown" class is assigned to the result if the engine cannot decide within the designated classes, since the processed service is not adopted and its corresponding flow is thereby rejected (i.e., false negative).

First, regarding the general non-IoT traces and using the selected features detailed in Table 6.2's first raw, the random forest, decision tree, and LSTM classifiers achieve the highest accuracy results compared to the remaining methods, whereas among these three techniques, the random forest algorithm provides slightly better classification results. The classification results of the random forest technique, in Figures 6.13a and 6.14a, show that the overall accuracy of this technique reaches over 93% (± 1.9%) with the UNIBS dataset and over 97% (± 0.6%) with the UPC dataset. This technique can detect the different non-IoT services with a minimum precision of 88%, in both datasets, corresponding to the SMTP service, implying that this mechanism can indubitably identify heterogeneous general services with very high accuracy. These results also prove the effectiveness and the viability of the selected features for general traffic identification.

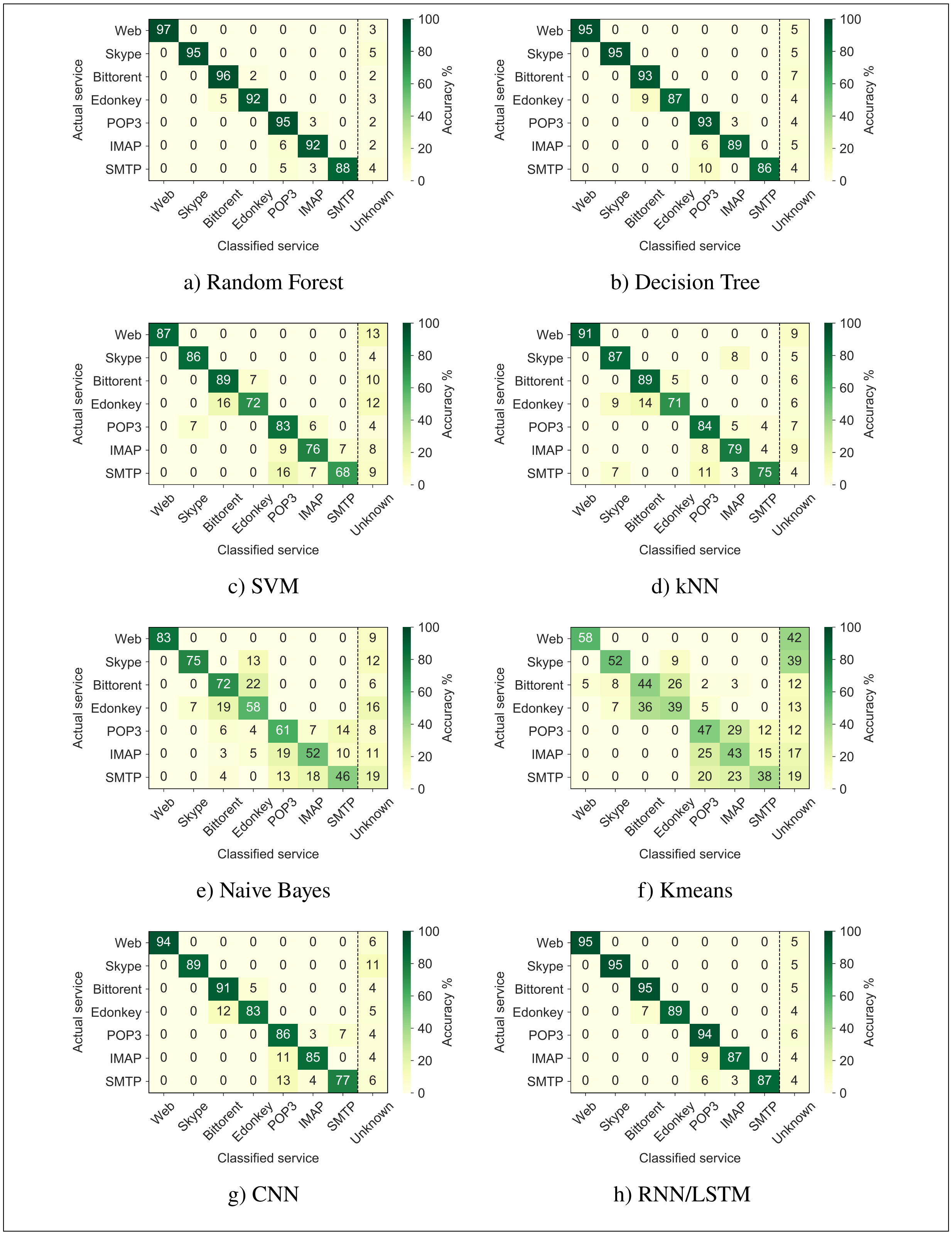


Figure 6.13 Confusion matrices of classification engines for UNIBS dataset

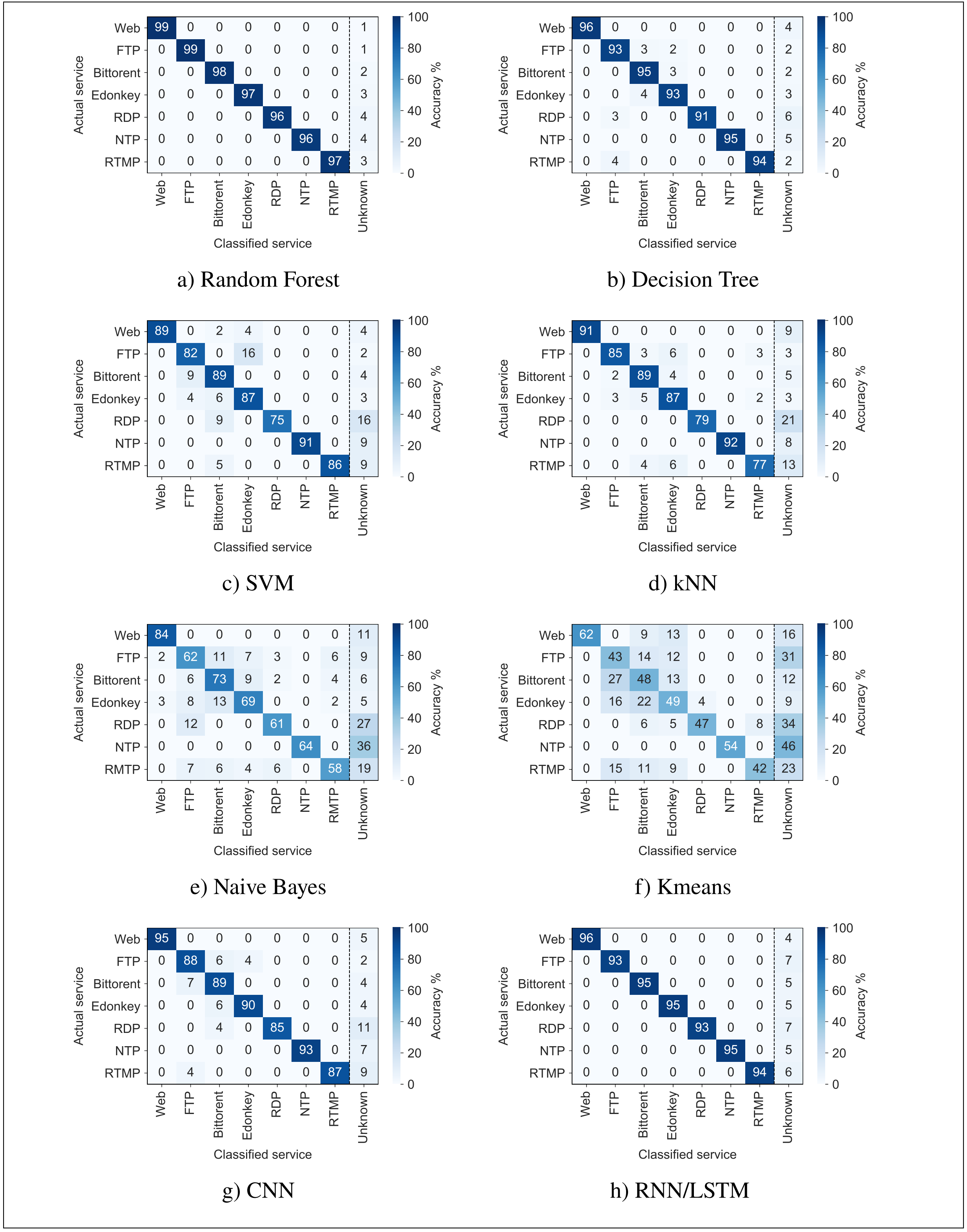


Figure 6.14 Confusion matrices of classification engines for UPC dataset

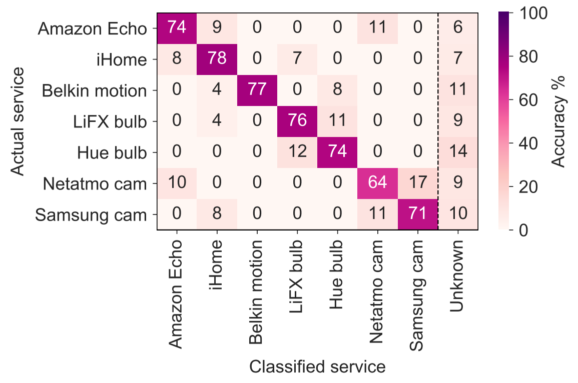
The rest of this section discusses the results of the machine learning techniques one by one, and then it addresses the deep learning methods. The decision tree is the second-best algorithm in terms of classification correctness. It provides 91.14% (± 4.6%) and 93.85% (± 3.2%) accuracy for classifying the UNIBS and UPC datasets, respectively. On the other hand, the accuracy of the SVM method is about 80.14% (± 5.3%) and 85.57% (± 4.2%), and that of the k-NN algorithm is about 82.28% (± 4.7%) and 85.71% (± 3.8%) for UNIBS and UPC, respectively. Apart from the overall accuracy of each specific classifier (e.g., SVM and k-NN), the individual accuracies of some particular services could vary, yielding a misclassification that does not consider the standard mapping between layers, such as the cases of the $\langle$POP3, IMAP, and SMTP$\rangle$ services that are all in the mail category within the UNIBS dataset, the $\langle$BitTorrent and eDonkey$\rangle$ services that belong to the P2P category, and also the P2P traffic that is misclassified with bulk transfer (i.e., FTP) in the UPC dataset. Meanwhile, the naive Bayes algorithm provides 63.85% (± 11.4%) and 67.28% (± 9.7%) accuracy with UNIBS and UPC, respectively, which presents the worst accuracy result compared to all the supervised machine learning methods. Furthermore, the accuracy performance of K-means is less effective than that of all the deployed classifiers. Figures 6.13f and 6.14f depict 45.85% (± 7.4%) and 49.28% (± 5.9%) with low precision for the large majority of the 14 classes within the UNIBS and UPC datasets, respectively.

Hence, while the application of clustering methods enables the prediction of non-labeled traffic, the experimental results prove that supervised learning techniques, which require training samples to be manually labeled in advance, have substantially higher classification accuracy than unsupervised learning algorithms in identifying known traffic. Regarding the CNN and LSTM mechanisms, while the deployment of the deep learning paradigm in network traffic classification is intricate and currently less approved, we can project from Figures 6.13 and 6.14 that their corresponding experimental results would reveal robustness comparable to the machine learning mechanisms. The CNN engine can achieve 86.42% (± 3.4%) and 89.57% (± 2.3%) accuracy for UNIBS and UPC, respectively. The LSTM classifier outperforms the CNN engine on both the considered datasets by reaching over 91.71% (± 2.7%) and 94.42% (± 1.8%) accuracy for UNIBS and UPC, respectively.

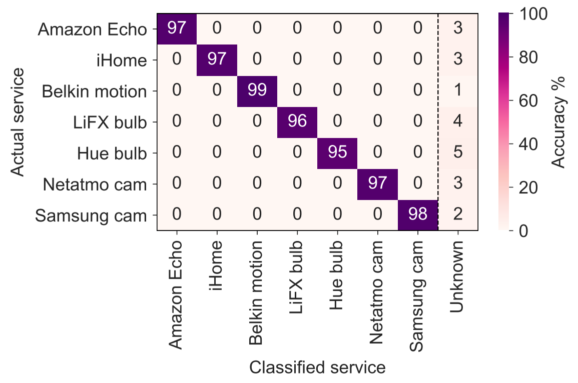
It is noteworthy that almost all the classifiers maintain relatively similar classification accuracy for the general non-IoT traffic in both datasets, with slight improvements in the UPC database, when the classifiers are affected by the size of the training dataset. These broad similarities prove the stability of the adopted classifiers, their corresponding configurations, and the effectiveness of the selected features.

For identifying IoT traffic, we consider the random forest algorithm from the machine learning techniques and the LSTM algorithm from the deep learning techniques, since they provide the highest accuracy with general Internet service differentiation. Figure 6.15 illustrates the resulting confusion matrices for the IoT-UNSW traces. By applying the random forest technique with the selected features for Internet traffic (detailed in Table 6.2's first raw), we get the results depicted in Figure 6.15a. From this figure, it is clear that the overall accuracy of the random forest technique is low ($73.42\% \pm 0.8\%$) compared to the general traffic classification results shown in Figures 6.13a and 6.14a. However, when we also consider the IoT-based features (e.g., sleep time and active time) as detailed in the second raw of Table 6.2, the random forest accuracy shows an impressive result of 97%, as indicated in Figure 6.15b.

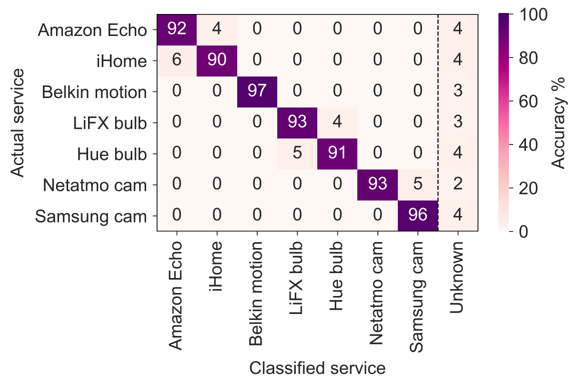
On the other hand, the LSTM algorithm applied to the IoT traces provides ($93.14\% \pm 1.7\%$) of accuracy, which presents similar inference results compared to the non-IoT traffic classification illustrated in Figures 6.13g and 6.14g. Thus, from Figure 6.15, it can be seen that machine learning algorithms such as random forest are highly dependent on the selected features. Furthermore, it is crucial to consider the type of traffic to be identified (e.g., IoT/non-IoT) when performing feature engineering. In fact, while selected discriminators for general traffic traces (e.g., packet size and packet inter-arrival time statistics) are significant for IoT service differentiation, they nevertheless, do not exclusively play the dominant role in the process; therefore, more specific features must be considered for IoT service differentiation. Meanwhile, the deep learning-based approach deploys an end-to-end multilayer model that eliminates the need for manual feature construction by automatically learning important IoT features and executes classification conjointly, enabling approximate accuracy results.



a) Random Forest with Internet traffic features

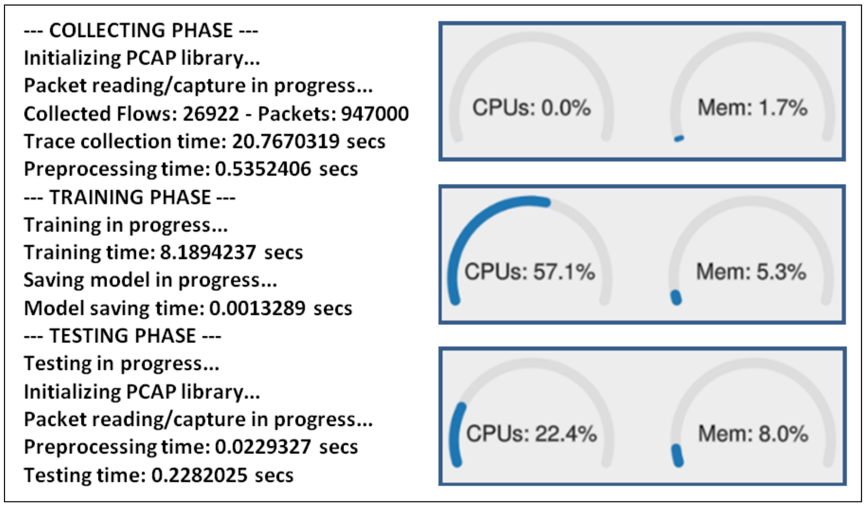


b) Random Forest with IoT traffic features



c) RNN/LSTM

Figure 6.15 Confusion matrices of classification engines for UNSW dataset



```
--- COLLECTING PHASE ---
Initializing PCAP library...
Packet reading/capture in progress...
Collected Flows: 26922 - Packets: 947000
Trace collection time: 20.7670319 secs
Preprocessing time: 0.5352406 secs
--- TRAINING PHASE ---
Training in progress...
Training time: 8.1894237 secs
Saving model in progress...
Model saving time: 0.0013289 secs
--- TESTING PHASE ---
Testing in progress...
Initializing PCAP library...
Packet reading/capture in progress...
Preprocessing time: 0.0229327 secs
Testing time: 0.2282025 secs
```

Figure 6.16 Example of resource consumption (time, memory, and CPU) for pre-processing, training, and testing operations

6.6.2.2 Time and space complexity

The analysis of large amounts of network traffic data with significant numbers of features requires many resources, such as time, memory, and CPU, as shown in Figure 6.16. Since we are considering different datasets and because data size affects processing performance and resource consumption, each classifier would perform differently on each dataset. In this section, we evaluate the engines' performance in terms of time, CPU, and memory requirements while surveying the corresponding time and space complexity approximation. Table 6.4 provides examples of values for resource consumption reached by the different engines to classify the IoT-based dataset.

Let us consider t as the number of training samples, f the number of features, and c the number of classes (labels). The naive Bayes classifier is the most time and space-efficient model compared to the remaining deployed classifiers. It has a training time complexity of $O(t.f.c)$ for computing the probability of every feature in the training set for each class. It has a training space complexity of $O(f.c)$ for storing attributes for each class. Besides, to identify an instance, its testing time and space complexity are $O(f.c)$ for managing features for each category. Thus, it is convenient for identifying latency-sensitive applications to assure real-time results.

Table 6.4 Example of resource consumption

Engine	**Metric**			
	Time (sec)		**Memory (%)**	
	Train	**Test**	**Train**	**Test**
NB	0.06	0.028	8.7	0.082
DT	14.16	0.031	6.6	1.5
RF	42.48 (mn)	0.405	27.3	9.5
SVM	134.118	0.022	8.1	8.6
kNN	0.02	45.4	18.6	3.25
Kmeans	–	3.974	–	7.4
CNN	94.82	0.039	7.4	0.6
LSTM	432.4	0.220	39.8	5.4

The decision tree algorithm is one of the fastest techniques among those deployed. It has a training time complexity of $O(f.t.\log(t))$ to sort the training data for f features, and a testing time complexity of $O(\log(t))$. Meanwhile, the training and testing space complexities are $O(number\_of\_nodes)$, where the number of nodes is counted from the root node to the leaf nodes. On the other hand, the random forest model, which consists of multiple decision tree classifiers, has a training time complexity of $O(k.f.t.\log(t))$, where k is the defined number of trees to build the forest. The same formulation is applied for the rest of the computational problems, where the decision tree complexities are multiplied by k. Hence, the use of random forest may be impractical with some real-time scenarios, mainly when the training dataset and the number of trees are significant that require the construction of many decision trees.

The SVM algorithm consumes much time and many memory resources during the training and testing phases. It has a training time and space complexity of $O(t^2)$. Its testing time complexity is $O(k.f)$, where k presents the number of support vectors; and its space complexity is $O(k)$ for storing k vectors. Thus, the computational complexity is exponentially affected by the training instance number, making the model computationally infeasible with massive datasets through extended processing time and high occupation of memory and CPU.

The k-NN algorithm is a resource-intensive computational mechanism because, for each test instance, it needs to keep track of all training data and find the neighbor nodes. It has a runtime complexity of $O\big(k.t.f\big)$ and a space complexity of $O\big(t.f\big)$, where k is the number of neighbors. It is noteworthy that during the cross-validation, we tested different values of k in situations where a large number of neighbors usually slows the classification processing time. The K-means algorithm has a runtime complexity of $O\big(t.f.K.I\big)$ and a space complexity of $O\big((t+K).f\big)$, where k is the number of clusters and I is the bound number of iterations. The main processing drawback of the K-means clustering algorithm is the required number of iterations to reach convergence (i.e., the stability of the K clusters).

Moreover, deep learning approaches have high computational costs, making resource consumption one of the biggest challenges. For example, to calculate the computational complexity of the CNN, we need to consider the total time complexity of all convolutional layers, which is estimated as $O\big(\sum_{l=1}^{d} n_{l-1}.s_l^2.n_l.m_l^2\big)$, where d is the number of convolutional layers, l is the index of the convolutional layer, n_l is the number of filters in the l^{th} layer, n_{l-1} is the number of input channel of the l^{th} layer, s_l is the size of the filter, and m_l is the size of the output feature map. The time cost of pooling layers and fully connected layers is not included in this formulation. These layers often demand 5-10% of the computational time (He & Sun, 2015; Tsironi *et al.*, 2017). Meanwhile, LSTM is local in space and time, i.e., its computational complexity per time step and weight is $O(1)$. Thus, the overall complexity of an LSTM per time step is $O(W)$, where W is the number of weights (Hochreiter & Schmidhuber, 1997).

6.6.3 Which algorithm is best?

On the basis of the comparisons and analyses performed in previous sections, we discuss the design of lightweight, fine-grained traffic identification engines. The answer to the question "Which algorithm is the best?" is: it depends on the application problem, since none of the algorithms ensure high accuracy with low computational cost, particularly with massive amounts of data.

The result analysis shows that the random forest machine learning-based approach achieves the highest accuracy for IoT and non-IoT application identification. However, it is highly dependent on careful feature extraction and selection processes. Furthermore, the random forest may be not efficient for real-time classification, specifically with delay-sensitive services, with a large training dataset, since it requires a great deal of time to build several decision trees. The naive Bayes algorithm requires the lowest computational cost but provides inaccurate results. On the other hand, the LSTM deep learning-based mechanism provides slightly inferior accuracy results in comparison to the random forest, but it eliminates the need for manual feature engineering by automatically learning important features from the raw input data. However, LSTM also requires a high computational cost.

Therefore, we recommend deploying CNN and random forest models in a cascade fashion for an accurate fine-grained classifier. The CNN technique assures an automatic feature engineering process with low computational complexity, which is a viable strategy with the evolving nature of heterogeneous network traffic and eliminates the obstacle of domain-expert-driven handcrafted features extraction required by machine learning-based architecture. Then, using the CNN extracted features, the random forest technique performs a sequential classification enabling to achieve a high level of accuracy. This whole engine should be implemented on a distributed-based architecture, such as the federated learning model that enables end-host devices (e.g., mobile phones) to learn and predict classes using their own data, thereby yielding lower latency, less resource consumption while preserving end-user privacy (Njah *et al.*, 2020; Yang *et al.*, 2021).

6.7 Conclusion and future directions

With the development of smart environments, a massive number of heterogeneous services with an evolving nature have emerged, creating challenges and pressure on network operators. Traffic analysis is a crucial step for enhancing various network management functionalities, such as QoS provisioning, anomaly detection, profiling, and resource optimization. Therefore, the network operators require accurate fine-grained flow identification. Numerous techniques have been proposed to tackle traffic classification problems while supporting the evolving nature of the networks. Various surveys, which present practical tools for studying and analyzing the existing technologies, have been proposed for the state-of-the-art. However, the existing published works focus on either recognizing IoT-based demands or identifying general-based demands but not on both, although these classes of traffic coexist and share the same network resources. Moreover, the existing studies have focused either on machine learning or deep learning approaches, while there is a great need with the evolving nature of network traffic for studying and comparing both of these approaches.

In this paper, we propose an extensible evaluation framework addressing the mentioned gaps in existing works on traffic analysis of intelligent environment. The framework is presented through a technical survey, including various machine learning and deep learning-based classifiers. First, we present a review to determine the best minimal set of characteristics for classifying IoT and non-IoT network traffic flows while considering early-stage classification. Then, we examine the most significant state-of-the-art modules relating to traffic analysis and service differentiation. The performance of the different modules is evaluated using common sets of traffic traces with corresponding ground truth and under the same hardware. Finally, we conduct rigorous evaluations and comparisons between the different engines' performances with confusion matrices and time and space complexity. This common platform can then serve as a reliable map for designing and deploying the optimal algorithm for fine-grained IoT and non-IoT service identification.

The experimental results show that the random forest machine learning-based approach achieves the highest accuracy for IoT and non-IoT application identification, but it is highly dependent on careful feature extraction and selection processes. Furthermore, the random forest may be not efficient for real-time classification with a large training dataset, particularly for delay-sensitive services. On the other hand, while the LSTM deep learning-based mechanism provides accurate results, it eliminates the need for manual feature engineering by automatically learning important features from the raw input data. However, it also requires a high computational cost.

Our future work will focus on investigating and implementing a lightweight engine for fine-grained traffic identification, including malicious traffic, in smart environments. The approach will include deep learning (CNN) and machine learning (random forest) techniques that will be combined and optimized in a cascade fashion over distributed architectures to ensure high classification accuracy with low computational complexity while eliminating domain-expert-driven handcrafted features, particularly with the evolving nature of network traffic.

CONCLUSION AND RECOMMENDATIONS

This chapter concludes this doctoral research work. It discusses the main addressed problems and the presented key contributions. Then, it opens the door for different future research directions.

7.1 Conclusions

This thesis has addressed several problems related to the deployment of reliable and efficient management of smart environments. Specifically, the following questions were investigated: *How to efficiently and optimally manage massive smart environments while ensuring reliable services? Is it possible to design generic SDN-based engines for real-time monitoring, managing, and optimizing various smart environments that have in common several technologies, functionalities, and use cases?* The introduction, research problem, and literature review chapters presented the general context and showed limitations of existing works according to different aspects, including traffic analysis and characterization, service assurance, resource optimization, and energy consumption reduction. Furthermore, various research questions have arisen in order to attain the main goal. We established five specific objectives in Chapter 3 that led to the design and development of new engines addressing the management of different smart environments (e.g., industrial and campus networks). Their related methods and contributions were presented, evaluated, discussed, and validated in Chapters 4, 5, and 6. Our contributions are summarized and discussed in the following paragraphs while highlighting their advances made in the state-of-the-art.

Generic SDN-based engines for the management of various smart environments: After studying the latest technologies that came out to support mobile devices (e.g., 5G, IoT, cloud computing, and softwarization), we investigated the design of generic SDN-based engines for real-time monitoring, managing, and optimizing various smart environments that have several common applications (e.g., real-time environment monitoring, diagnostics, and emergency detection) and technologies (e.g., wired/wireless communications systems, and sensor networks). This has led to reviewing the criteria of different environments, precisely industrial and e-learning applications, in order to deduce the common network functionalities between them. The study concluded that some engines (e.g., infrastructure recovery, traffic analysis, and service differentiation) could be generic, while others (e.g., service assurance, resource optimization, etc.) need more effort and adjustment based on the specific network type.

Service assurance and resource optimization: To evaluate network management efficiency and reliability, we have studied service assurance and resource optimization aware routing problems through different models, techniques, and algorithms proposed in the first and second methodologies. To begin with, in contrast to the existing works, we did not address just one type of service (e.g., delay-sensitive flows), but all heterogeneous network services, including bandwidth-hungry services that should be under the control of specific engineering policies to not create congestion and degrade the performance of mission-critical applications. Hence, we demonstrated how service characterization plays a crucial role in appropriate mechanism design and network performance. Within each smart environment, we formulated the optimization routing problems as multi-constrained path selection models. Then, based on the traffic type, the service-dependent QoS requirements, and the constrained network resources, we devised various routing algorithms for heterogeneous flow management and resource optimization.

We designed over centralized control systems reactive and proactive engines, in addition to different algorithms using the Lagrangian Relaxation theory to assure various QoS requirements for each flow while optimizing resource allocation. It is noteworthy that in the literature, the existing routing algorithms are mainly based on a per-flow approach that, using the current state of the network, computes optimal path solution independently for each traffic flow within a set of hundreds, even thousands of concurrent flows. However, while this per-flow technique can reach a fast routing decision, it cannot provide an optimal global solution for the whole set in the network. In our work, precisely the first and second methodologies presented in Chapters 4 and 5, we adopted two mechanisms that enable handling heterogeneous concurrent flows simultaneously.

The first proposed routing mechanism for industrial network introduces a centralized multi-program platform (using the programmability feature of SDN) composed of triplet algorithms: One algorithm has been devised to conduct a parallel routing mechanism, while a pair of routing algorithms has been designed based on the Lagrangian relaxation approach (GEN-LARAC mechanism) to manage two different classes of traffic. The building block GEN-LARAC scheme is a generalization of the LARAC algorithm, which is a polynomial-time algorithm that efficiently finds the optimal route for a specific demand in $O([l + v \log(v)]^2)$ time complexity, where v represents the number of nodes and l the number of links in the topology. Then, we extend this multi-program approach over the smart campus environment to handle a whole set of concurrent flows through a decomposed strategy, using the Lagrangian Dual Decomposition approach. The entire problem has been decomposed into per-flow sub-problems to be solved simultaneously and to allocate network flows distributively across paths having low bandwidth utilization. Accordingly, the proposed routing strategy finds an optimal solution not only for each flow independently but for the whole set of flows while performing coordination between their various QoS requirements and the available resources. Similarly to the multi-program approach, this decomposed approach could be solved in polynomial time, where the complexity of solving the problem increases significantly when the problem size scales up with the number of concurrent flows F and considered paths P. Therefore, the use of parallel computation in multi-core systems and the deployment of the cluster-based distributed controller technology are advised to alleviate this challenge.

We evaluated our proposed schemes from different aspects, including the number of simultaneous heterogeneous flows, QoS provisioning, characterization impacts, constrained resource allocation, and network scalability. Compared to the well-known benchmarks in the routing awareness problem, our experimental results with large numbers of flows showed promising performance in terms of reliability, service assurance, and resource optimization.

Energy consumption awareness: Since the intensive setup of multiple IoT devices leads to tremendous growth in the amount of generated electric data, the energy-awareness aspect presents one of the major concerns in smart environments. However, existing works in the literature focus on restrictive network criteria, i.e., either on service assurance or on reducing energy consumption, which degrades either network or service performance, respectively. Therefore, in this thesis, in addition to service assurance and resource optimization, we also considered the energy consumption reduction issue to build sustainable smart environments. Using SDN's flexibility and programmability aspects, we introduced energy-aware data transfer engines that efficiently reduce the infrastructure energy while establishing dynamic routes by aggregating flows across activated network resources. Based on the type of traffic, we supported the introduction of good trade-offs between network service assurance, resource optimization, and energy consumption reduction to enhance the overall network performance. The obtained results confirmed that the proposed scheme has significantly improved the total energy consumption reduction.

Smart environment heterogeneous service differentiation: Since flow identification presents a fundamental network functionality to support service assurance and resource optimization aware routing, we provided an overview of the different types of services inside each intelligent environment. Furthermore, considering that the number of services generated from IoT and non-IoT devices is unlimited, we defined a set of traffic classes with their appropriate QoS requirements and priority levels in the network, according to the purpose of each smart environment. To technically achieve such characterization, we proposed a comprehensive up-to-date technical survey that facilitates the selection and further development of lightweight fine-grained service differentiation engines. Hence, first of all, we provided a feature engineering review for discriminating heterogeneous IoT and non-IoT traffic, while considering early-stage identification and selecting only relevant attributes to support real-time operations, decrease computational complexity, and preserve high accuracy. Then, we deployed the most significant modules in the state-of-the-art related to traffic analysis and service differentiation, including machine and deep learning classifiers. We conducted rigorous evaluations and comparisons between the different engines using confusion matrices and computational complexity. Experimental results showed that random forest achieves the highest accuracy for IoT and non-IoT application identification. However, it is highly dependent on careful feature engineering processes and requires a high computational cost with large training dataset and high number of decision trees. On the other hand, while the LSTM model eliminates the need for handcrafted feature engineering by automatically learning important features from the raw input data, it provides significant accurate results. However, it also requires a high computational cost. Consequently, as an answer to the question: “which algorithm is the best to perform service differentiation?”, we recommended the deployment of CNN and random forest models in a cascade fashion, where the deep learning approach assures an automatic feature engineering process with low computational complexity, and random forest ensures the fine-grained classification using the CNN results. Nevertheless, the whole engine should be implemented on a distributed-based architecture (e.g., SDN-based end-host network-level proposed in Chapter 5 or edge computing) to satisfy real-time data analytics.

These main contributions are detailed in the following scientific papers:

- Yosra Njah and Mohamed Cheriet. "Parallel route optimization and service assurance in energy-efficient software-defined industrial IoT networks." IEEE Access 9 (2021): 24682-24696.

- Yosra Njah, Chuan Pham, and Mohamed Cheriet. "Service and resource aware flow management scheme for an SDN-based smart digital campus environment." IEEE Access 8 (2020): 119635-119653.
- Yosra Njah, Misha Cattaneo, Jean Hennebert, and Mohamed Cheriet. "Machine learning and deep learning-based evaluation framework for IoT and non-IoT smart network traffic analytics." submitted to IEEE Internet of Things Journal.

7.2 Future works

In this thesis, we have considered critical dimensions in our evaluation framework, and we have presented original contributions to the state-of-the-art addressing the reliable and efficient management of smart environments. Despite achieving outstanding design and performance, other dimensions would be considered in future work, and the proposed solutions could be enriched through different upgrades.

First, it is essential to mention that numerous environments still not yet very well explored and need more reviews, such as healthcare, retails, and transportation environments that could be investigated in future works.

Then, regarding the SDN-based engines for service assurance and resource optimization of various intelligent environments:

- The multi-program platform in Chapter 4 could be extended in the future through the implementation of additional algorithms for managing more specific traffic classes, such as emergency situations in smart cities (Rego *et al.*, 2018).
- The decomposed strategy in Chapter 5 could be extended in the future through the integration of more constraints in the optimization problem, such as the limited number of SDN-based rules in the forwarding devices and the energy awareness constraint (using penalty in the objective function if a large number of links are activated).

- The multi-program platform (Chapter 4) and the decomposed approach (Chapter 5) could be combined to enable the simultaneous management of each class of traffic with its corresponding set of concurrent flows using the appropriate algorithm.
- These centralized SDN-based engines for service assurance and resource optimization should be deployed over cluster-based distributed controlling paradigm (e.g., edge and fog computing), since this issue has significant impacts on the control plane's effectiveness and scalability, and thereby on the reliability and efficiency of the overall network performance. In this perspective, the management of the wired and wireless forwarding elements as well as the interaction between the different controllers (e.g., SDN and SDR) and the controller orchestrator should be technically considered to optimize and evaluate the unified fully-programmable management processes.

Regarding the SDN-based engines for heterogeneous traffic analysis and service differentiation over various smart environments:

- Based on the technical survey presented in Chapter 6, we open the door for implementing the designed fine-grained service differentiation engine that combines CNN and random forest models in a cascade fashion over a distributed-based architecture (e.g., SDN-based end-host network-level proposed in Chapter 5).
- Network security assessments and intrusion detection systems (IDSs) are directly linked to traffic analysis and service differentiation. Therefore, in addition to the normal traffic identification, attack behavior and malicious traffic inspection can be considered in future works. Two levels of the process could be studied. The first one identifies whether the observed traffic is generated by legitimate applications or malware (e.g., botnets). Then, it consists of classifying and blocking the malware type.

BIBLIOGRAPHY

Aceto, G., Ciuonzo, D., Montieri, A. & Pescapè, A. (2019a). MIMETIC: Mobile encrypted traffic classification using multimodal deep learning. *Computer Networks*, 165, 106944.

Aceto, G., Ciuonzo, D., Montieri, A. & Pescapé, A. (2019b). Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges. *IEEE Transactions on Network and Service Management*, 16(2), 445–458.

Ahmed, E., Yaqoob, I., Gani, A., Imran, M. & Guizani, M. (2016). Internet-of-things-based smart environments: state of the art, taxonomy, and open research challenges. *IEEE Wireless Communications*, 23(5), 10–16.

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M. & Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials*, 17(4), 2347–2376.

Al-Garadi, M. A., Mohamed, A., Al-Ali, A. K., Du, X., Ali, I. & Guizani, M. (2020). A survey of machine and deep learning methods for internet of things (IoT) security. *IEEE Communications Surveys & Tutorials*, 22(3), 1646–1685.

Al-Rubaye, S., Kadhum, E., Ni, Q. & Anpalagan, A. (2017). Industrial internet of things driven by SDN platform for smart grid resiliency. *IEEE Internet of Things Journal*, 6(1), 267–277.

Alalewi, A., Dayoub, I. & Cherkaoui, S. (2021). On 5G-V2X Use Cases and Enabling Technologies: A Comprehensive Survey. *IEEE Access*.

Alam, I., Sharif, K., Li, F., Latif, Z., Karim, M. M., Nour, B., Biswas, S. & Wang, Y. (2019). IoT virtualization: a survey of software definition & function virtualization techniques for internet of things. *arXiv preprint arXiv:1902.10910*.

Alberti, A. M., Santos, M. A., Souza, R., Da Silva, H. D. L., Carneiro, J. R., Figueiredo, V. A. C. & Rodrigues, J. J. (2019). Platforms for smart environments and future Internet design: A survey. *IEEE Access*, 7, 165748–165778.

AlZoman, R. M. & Alenazi, M. J. (2021). A comparative study of traffic classification techniques for smart city networks. *Sensors*, 21(14), 4677.

Andreas Wächter, C. L. (2019. Accessed: April 2, 2019). Ipopt solver in julia. Consulted at https://github.com/JuliaOpt/Ipopt.jl.

Atlam, H. F., Walters, R. J. & Wills, G. B. (2018). Fog computing and the internet of things: A review. *big data and cognitive computing*, 2(2), 10.

Auld, T., Moore, A. W. & Gull, S. F. (2007). Bayesian neural networks for internet traffic classification. *IEEE Transactions on neural networks*, 18(1), 223–239.

Baker, F., Polk, J. & Dolly, M. (2010). A Differentiated Services Code Point (DSCP) for Capacity-Admitted Traffic. *Internet Engineering Task Force (IETF).*

Bakker, J., Ng, B., Seah, W. K. & Pekar, A. (2019). Traffic classification with machine learning in a live network. *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pp. 488–493.

Begen, A. C., Altunbasak, Y., Ergun, O. & Ammar, M. H. (2005). Multi-path selection for multiple description video streaming over overlay networks. *Signal Processing: Image Communication*, 20(1), 39–60.

Bera, S., Misra, S. & Vasilakos, A. V. (2017). Software-Defined Networking for Internet of Things: A Survey. *IEEE Internet of Things Journal*, 4(6), 1994-2008.

Besard, T., Foket, C. & De Sutter, B. (2018). Effective extensible programming: Unleashing julia on gpus. *IEEE Transactions on Parallel and Distributed Systems*, 30(4), 827–841.

Beshley, M., Seliuchenko, M., Panchenko, O. & Polishuk, A. (2017). Adaptive flow routing model in SDN. *2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)*, pp. 298–302.

Bezanson, J. (Accessed: April 23, 2020). Julia Programming Language. Consulted at https://julialang.org/.

Bezanson, J., Edelman, A., Karpinski, S. & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM review*, 59(1), 65–98.

Bormann, C., Ersue, M. & Keranen, A. (2014). Terminology for constrained-node networks. *Internet Engineering Task Force (IETF): Fremont, CA, USA*, 2070–1721.

Botta, A., Dainotti, A. & Pescapé, A. (2012). A tool for the generation of realistic network workload for emerging networking scenarios. *Computer Networks*, 56(15), 3531–3547.

Boutaba, R., Salahuddin, M. A., Limam, N., Ayoubi, S., Shahriar, N., Estrada-Solano, F. & Caicedo, O. M. (2018). A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. *Journal of Internet Services and Applications*, 9(1), 1–99.

Cacheda, R. A., García, D. C., Cuevas, A., Castano, F. J. G., Sánchez, J. H., Koltsidas, G., Mancuso, V., Novella, J. I. M., Oh, S. & Pantò, A. (2007). QoS requirements for multimedia services. In *Resource management in satellite networks* (pp. 67–94). Springer.

Cai, H., Xu, B., Jiang, L. & Vasilakos, A. V. (2016). IoT-based big data storage systems in cloud computing: perspectives and challenges. *IEEE Internet of Things Journal*, 4(1), 75–87.

Campus-Management, C. (Accessed: July 7, 2019). Transform your campus. Consulted at https://www.campusmanagement.com/.

Chan, H. C. & Chan, L. (2018). Smart Library and Smart Campus. *Journal of Service Science and Management*, 11(6), 543–564.

Chen, Y., Farley, T. & Ye, N. (2004). QoS requirements of network applications on the Internet. *Information Knowledge Systems Management*, 4(1), 55–76.

Cheng, J., Chen, W., Tao, F. & Lin, C.-L. (2018). Industrial IoT in 5G environment towards smart manufacturing. *Journal of Industrial Information Integration*, 10, 10–19.

Chollet, F. (Accessed: October 20, 2020). Keras Library. Consulted at https://keras.io/.

Cisco. (Accessed: July 12, 2019). Build a Flexible Campus. Consulted at https://www.cisco.com/c/en/us/solutions/industries/education/connected-campus.html.

Cisco. (Accessed: September 12, 2021). Cisco Annual Internet Report. Consulted at https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html.

Cisco, V. N. (Accessed: April 22, 2019). Cisco visual networking index: Forecast and methodology 2015-2020. *White paper, CISCO*. Consulted at https://static1.squarespace.com/static/54496f89e4b0ad2be6456bc7/t/57b386f8e4fcb59cf4894907/1471383319341/Cisco+Forecast+2015-2020.pdf.

Cisco-CCNA. (Accessed: October 20, 2020). Computer Networking Notes. Consulted at https://www.computernetworkingnotes.com/ccna-study-guide/ospf-metric-cost-calculation-formula-explained.html.

Connor, J. T., Martin, R. D. & Atlas, L. E. (1994). Recurrent neural networks and robust time series prediction. *IEEE transactions on neural networks*, 5(2), 240–254.

Controller, F. (Accessed: October 20, 2020). Floodlight REST API. Consulted at https://floodlight.atlassian.net/wiki/x/C4BNAQ.

Coresecurity. (Accessed: October 20, 2020). Pcapy Library. Consulted at https://pypi.org/project/pcapy/.

Cournapeau, D. (Accessed: October 20, 2020). Scikit-learn Library. Consulted at https://scikit-learn.org/stable/.

Cui, L., Yu, F. R. & Yan, Q. (2016). When big data meets software-defined networking: SDN for big data and big data for SDN. *IEEE network*, 30(1), 58–65.

Curtis, A. R., Kim, W. & Yalagandula, P. (2011). Mahout: Low-overhead datacenter traffic management using end-host-based elephant detection. *Infocom*, 11, 1629–1637.

Da Xu, L., He, W. & Li, S. (2014). Internet of things in industries: A survey. *IEEE Transactions on industrial informatics*, 10(4), 2233–2243.

Daimler. (Accessed: October 20, 2020). Daimler - Industry 4.0. Consulted at https://www.daimler.com/innovation/production/.

Dainotti, A., Pescape, A. & Claffy, K. C. (2012). Issues and future directions in traffic classification. *IEEE network*, 26(1), 35–40.

Deloitte. (Accessed: July 11, 2019). Smart Campuses. Consulted at https://www2.deloitte.com/us/en/pages/consulting/solutions/next-generation-smart-campus.html.

Dong, S. (2021). Multi class SVM algorithm with active learning for network traffic classification. *Expert Systems with Applications*, 176, 114885.

Dunning, I., Huchette, J. & Lubin, M. (2017). JuMP: A modeling language for mathematical optimization. *SIAM Review*, 59(2), 295–320.

Dutra, D. L. C., Bagaa, M., Taleb, T. & Samdanis, K. (2017). Ensuring end-to-end QoS based on multi-paths routing using SDN technology. *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pp. 1–6.

Egilmez, H. E., Dane, S. T., Bagci, K. T. & Tekalp, A. M. (2012). OpenQoS: An OpenFlow controller design for multimedia delivery with end-to-end Quality of Service over Software-Defined Networks. *Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*, pp. 1–8.

Egilmez, H. E., Civanlar, S. & Tekalp, A. M. (2013). An optimization framework for QoS-enabled adaptive video streaming over OpenFlow networks. *IEEE Transactions on Multimedia*, 15(3), 710–715.

Ersue, M., Romascanu, D., Schoenwaelder, J. & Herberg, U. (2015). Management of networks with constrained devices: problem statement and requirements. *IETF RFC 7547, 2015.*

Fahad, A., Tari, Z., Khalil, I., Habib, I. & Alnuweiri, H. (2013). Toward an efficient and scalable feature selection approach for internet traffic classification. *Computer Networks*, 57(9), 2040–2057.

Faheem, M., Shah, S. B. H., Butt, R. A., Raza, B., Anwar, M., Ashraf, M. W., Ngadi, M. A. & Gungor, V. C. (2018). Smart grid communication and information technologies in the perspective of Industry 4.0: Opportunities and challenges. *Computer Science Review*, 30, 1–30.

Fan, Z. & Liu, R. (2017). Investigation of machine learning based network traffic classification. *2017 International Symposium on Wireless Communication Systems (ISWCS)*, pp. 1–6.

Fazel, M., Hindi, H. & Boyd, S. P. (2003). Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices. *Proceedings of the 2003 American Control Conference, 2003.*, 3, 2156–2162.

Fazio, M., Celesti, A., Puliafito, A. & Villari, M. (2015). Big data storage in the cloud for smart environment monitoring. *Procedia Computer Science*, 52, 500–506.

Finn, N., Thubert, P., Varga, B. & Farkas, J. (2017). Deterministic networking architecture. *draft-ietf-detnet-architecture-03 (work in progress)*, 1–43.

Finsterbusch, M., Richter, C., Rocha, E., Muller, J.-A. & Hanssgen, K. (2013). A survey of payload-based traffic classification approaches. *IEEE Communications Surveys & Tutorials*, 16(2), 1135–1156.

Floodlight, C. (Accessed: October 20, 2020). Floodlight - How to Collect Switch Statistics (and Compute Bandwidth Utilization). Consulted at https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/overview.

Foubert, B. & Mitton, N. (2020). Long-range wireless radio technologies: A survey. *Future internet*, 12(1), 13.

General-Electric. (Accessed: October 20, 2020). Putting the Industrial Internet to Work. Consulted at https://www.ge.com/digital/sites/default/files/download_assets/2019-Digital-Transformation-Playbook-GE.pdf.

General-Electric. (Accessed: October 21, 2020). Everything you need to know about the Industrial Internet of Things. Consulted at https://www.ge.com/digital/blog/everything-you-need-know-about-industrial-internet-things.

Gharakheili, H. H., Sivaraman, V., Vishwanath, A., Exton, L., Matthews, J. & Russell, C. (2016). SDN APIs and Models for Two-Sided Resource Management in Broadband Access Networks. *IEEE Trans. Network and Service Management*, 13(4), 823–834.

Google-Brain, T. (Accessed: October 20, 2020). TensorFlow Library. Consulted at https://www.tensorflow.org/.

Grandi, M., Camps-Mur, D., Betzler, A., Aleixendri, J. J. & Catalan-Cid, M. (2018). SWAM: SDN-based Wi-Fi Small Cells with Joint Access-Backhaul and Multi-Tenant Capabilities. *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, pp. 1–2.

Guck, J. W., Reisslein, M. & Kellerer, W. (2016). Function split between delay-constrained routing and resource allocation for centrally managed QoS in industrial networks. *IEEE Transactions on Industrial Informatics*, 12(6), 2050–2061.

Guck, J. W., Van Bemten, A. & Kellerer, W. (2017). DetServ: Network models for real-time QoS provisioning in SDN-based industrial environments. *IEEE Transactions on Network and Service Management*, 14(4), 1003–1017.

Guck, J. W., Van Bemten, A., Reisslein, M. & Kellerer, W. (2018). Unicast QoS routing algorithms for SDN: A comprehensive survey and performance evaluation. *IEEE Communications Surveys & Tutorials*, 20(1), 388–415.

Guo, L. & Matta, I. (2003). Search space reduction in QoS routing. *Computer Networks*, 41(1), 73–88.

Habibi Gharakheili, H., Sivaraman, V., Moors, T., Vishwanath, A., Matthews, J. & Russell, C. (2017). Enabling Fast and Slow Lanes for Content Providers Using Software Defined Networking. *IEEE/ACM Transactions on Networking (TON)*, 25(3), 1373–1385.

Hadi, M. S., Lawey, A. Q., El-Gorashi, T. E. & Elmirghani, J. M. (2018). Big data analytics for wireless and wired network design: A survey. *Computer Networks*, 132, 180–199.

Hajjaji, Y., Boulila, W., Farah, I. R., Romdhani, I. & Hussain, A. (2021). Big data and IoT-based applications in smart environments: A systematic review. *Computer Science Review*, 39, 100318.

Haseeb, K., Almogren, A., Ud Din, I., Islam, N. & Altameem, A. (2020). SASC: Secure and Authentication-Based Sensor Cloud Architecture for Intelligent Internet of Things. *Sensors*, 20(9), 2468.

He, K. & Sun, J. (2015). Convolutional neural networks at constrained time cost. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5353–5360.

Henneke, D., Wisniewski, L. & Jasperneite, J. (2016). Analysis of realizing a future industrial network by means of Software-Defined Networking (SDN). *2016 IEEE World Conference on Factory Communication Systems (WFCS)*, pp. 1–4.

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735–1780.

Huawei, T. (2015). Huawei Agile Campus Network Solution. *white paper*. Consulted at https://www.karma-group.ru/upload/iblock/496/Huawei%20Agile%20Campus%20Network%20Solution%20Brochure%20(Detailed%20Version)%20.pdf.

Huawei, T. (Accessed: December 19, 2019). SDN, a New Definition of Next-Generation Campus Network. *white paper 2013*. Consulted at https://www.huawei.com/ilink/en/download/HW_274562.

Hwang, R.-H., Peng, M.-C., Nguyen, V.-L. & Chang, Y.-L. (2019). An LSTM-based deep learning approach for classifying malicious traffic at the packet level. *Applied Sciences*, 9(16), 3414.

Jammal, M., Singh, T., Shami, A., Asal, R. & Li, Y. (2014). Software defined networking: State of the art and research challenges. *Computer Networks*, 72, 74–98.

Johansson, A. & Sandberg, O. (2018). A Comparative Study of Deep-Learning Approaches for Activity Recognition Using Sensor Data in Smart Office Environments. Malmö universitet/Teknik och samhälle.

Jurca, D. & Frossard, P. (2007). Media flow rate allocation in multipath networks. *IEEE Transactions on Multimedia*, 9(6), 1227–1240.

Juttner, A., Szviatovski, B., Mécs, I. & Rajkó, Z. (2001). Lagrange relaxation based method for the QoS routing problem. *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2, 859–868.

Karakus, M. & Durresi, A. (2017). Quality of service (qos) in software defined networking (sdn): A survey. *Journal of Network and Computer Applications*, 80, 200–218.

Kim, H., Claffy, K. C., Fomenkov, M., Barman, D., Faloutsos, M. & Lee, K. (2008). Internet traffic classification demystified: myths, caveats, and the best practices. *Proceedings of the 2008 ACM CoNEXT conference*, pp. 1–12.

Kim, J., Kim, J., Thu, H. L. T. & Kim, H. (2016). Long short term memory recurrent neural network classifier for intrusion detection. *2016 International Conference on Platform Technology and Service (PlatCon)*, pp. 1–5.

Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S. & Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1), 14–76.

Krishnan, S., Montavont, N., Njedjou, E., Veerepalli, S. & Yegin, A. (2007). Link-layer event notifications for detecting network attachments. *Internet Engineering Task Force (IETF).*

Lantz, B. (Accessed: October 20, 2020). Mininet project. Consulted at http://mininet.org/.

Li, D., Zhou, M.-T., Zeng, P., Yang, M., Zhang, Y. & Yu, H. (2016). Green and reliable software-defined industrial networks. *IEEE Communications Magazine*, 54(10), 30–37.

Li, G., Wu, J., Li, J., Wang, K. & Ye, T. (2018a). Service popularity-based smart resources partitioning for fog computing-enabled industrial Internet of Things. *IEEE Transactions on Industrial Informatics*, 14(10), 4702–4711.

Li, S., Da Xu, L. & Zhao, S. (2018b). 5G Internet of Things: A survey. *Journal of Industrial Information Integration*, 10, 1–9.

Liao, L. X., Chao, H.-C. & Chen, M.-Y. (2020). Intelligently modeling, detecting, and scheduling elephant flows in software defined energy cloud: A survey. *Journal of Parallel and Distributed Computing*, 146, 64–78.

Liao, S., Cheng, W., Liu, W., Yang, Z. & Ding, Y. (2007). Distributed optimization for utility-energy tradeoff in wireless sensor networks. *2007 IEEE International Conference on Communications*, pp. 3190–3194.

Lin, S.-W., Miller, B., Durand, J., Bleakley, G., Chigani, A., Martin, R., Murphy, B. & Crawford, M. (2017). The industrial internet of things volume G1: reference architecture. *Industrial Internet Consortium*, 10–46.

Lin, S.-C., Akyildiz, I. F., Wang, P. & Luo, M. (2016). QoS-aware adaptive routing in multi-layer hierarchical software defined networks: A reinforcement learning approach. *2016 IEEE International Conference on Services Computing (SCC)*, pp. 25–33.

Lin, X., Shroff, N. B. & Srikant, R. (2006). A tutorial on cross-layer optimization in wireless networks. *IEEE Journal on Selected areas in Communications*, 24(8), 1452–1463.

Liu, Y., Niu, D. & Li, B. (2016). Delay-optimized video traffic routing in software-defined interdatacenter networks. *IEEE Transactions on Multimedia*, 18(5), 865–878.

Long, N. B., Tran-Dang, H. & Kim, D.-S. (2018). Energy-aware real-time routing for large-scale industrial Internet of Things. *IEEE Internet of Things Journal*, 5(3), 2190–2199.

Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A. & Lloret, J. (2017). Network traffic classifier with convolutional and recurrent neural networks for Internet of Things. *IEEE Access*, 5, 18042–18050.

Lubin, M. & Dunning, I. (2015). Computing in operations research using Julia. *INFORMS Journal on Computing*, 27(2), 238–248.

Macedo, D. F., Guedes, D., Vieira, L. F., Vieira, M. A. & Nogueira, M. (2015). Programmable networks–From software-defined radio to software-defined networking. *IEEE communications surveys & tutorials*, 17(2), 1102–1125.

Marler, R. T. & Arora, J. S. (2010). The weighted sum method for multi-objective optimization: new insights. *Structural and multidisciplinary optimization*, 41(6), 853–862.

Mendiola, A., Astorga, J., Jacob, E. & Higuero, M. (2017). A survey on the contributions of software-defined networking to traffic engineering. *IEEE Communications Surveys & Tutorials*, 19(2), 918–953.

Mocnej, J., Pekar, A., Seah, W. K. & Zolotova, I. (2018). Network Traffic Characteristics of the IoT Application Use Cases. *School of Engineering and Computer Science, Victoria University, Wellington, New Zealand, Tech. Rep. ECSTR18-01*. Consulted at https://ecs.wgtn.ac.nz/foswiki/pub/Main/TechnicalReportSeries/IoT_network_technologies_embfonts.pdf.

Mohammadi, M., Al-Fuqaha, A., Sorour, S. & Guizani, M. (2018). Deep Learning for IoT Big Data and Streaming Analytics: A Survey. *IEEE Communications Surveys Tutorials*, 20(4), 2923-2960. doi: 10.1109/COMST.2018.2844341.

Mohammed, A. R., Mohammed, S. A. & Shirmohammadi, S. (2019). Machine learning and deep learning based traffic classification and prediction in software defined networking. *2019 IEEE International Symposium on Measurements & Networking (M&N)*, pp. 1–6.

Moore, A., Zuev, D. & Crogan, M. (2013). *Discriminators for use in flow-based classification*.

Moore, A. W. & Zuev, D. (2005). Internet traffic classification using bayesian analysis techniques. *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pp. 50–60.

Mu, T.-Y., Al-Fuqaha, A., Shuaib, K., Sallabi, F. M. & Qadir, J. (2018). SDN flow entry management using reinforcement learning. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 13(2), 1–23.

Naeem, F., Tariq, M. & Poor, H. V. (2020). SDN-enabled Energy-Efficient Routing Optimization Framework for Industrial Internet of Things. *IEEE Transactions on Industrial Informatics*.

Nedic, A. & Ozdaglar, A. (2009). Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1), 48.

Nguyen, T. T. & Armitage, G. (2008). A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys Tutorials*, 10(4), 56-76. doi: 10.1109/SURV.2008.080406.

Njah, Y. & Cheriet, M. (2021). Parallel route optimization and service assurance in energy-efficient software-defined industrial IoT networks. *IEEE Access*, 9, 24682–24696.

Njah, Y., Pham, C. & Cheriet, M. (2020). Service and Resource Aware Flow Management Scheme for an SDN-Based Smart Digital Campus Environment. *IEEE Access*, 8, 119635–119653.

Nokia. (Accessed: October 20, 2020). Industrial-grade Private Wireless for manufacturing. Consulted at https://www.nokia.com/networks/go-allwhere/private-wireless/manufacturing/.

Nunez, J., Baranda, J., Pascual, I. & Mangues-Bafalluy, J. (2016). WiseHAUL: An SDN-empowered Wireless Small Cell Backhaul testbed. *Procs of the 17th IEEE Int. Symp. on a World of Wireless, Mobile and Multimedia Networks (WoWMoM). IEEE, Coimbra, Portugal*.

Okay, F. Y. & Ozdemir, S. (2018). Routing in Fog-Enabled IoT Platforms: A Survey and an SDN-Based Solution. *IEEE Internet of Things Journal*, 5(6), 4871-4889. doi: 10.1109/JIOT.2018.2882781.

OpenFlow Switch, S. (2015). OpenFlow Switch Specification Version 1.3 (Protocol v. 0x06) - ONF TS-009. Consulted at https://opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.3.2.pdf.

Orlowski, S., Wessäly, R., Pióro, M. & Tomaszewski, A. (2010). SNDlib 1.0—Survivable network design library. *Networks: An International Journal*, 55(3), 276–286.

Pacheco, F., Exposito, E., Gineste, M., Baudoin, C. & Aguilar, J. (2019). Towards the Deployment of Machine Learning Solutions in Network Traffic Classification: A Systematic Survey. *IEEE Communications Surveys Tutorials*, 21(2), 1988-2014. doi: 10.1109/COMST.2018.2883147.

Palomar, D. P. & Chiang, M. (2006). A tutorial on decomposition methods for network utility maximization. *IEEE Journal on Selected Areas in Communications*, 24(8), 1439–1451.

Qi, Q. & Tao, F. (2019). A smart manufacturing service system based on edge computing, fog computing, and cloud computing. *IEEE Access*, 7, 86769–86777.

Rego, A., Garcia, L., Sendra, S. & Lloret, J. (2018). Software Defined Network-based control system for an efficient traffic management for emergency situations in smart cities. *Future Generation Computer Systems*, 88, 243–253.

Rezaee, M. & Moghaddam, M. H. Y. (2019). SDN-based quality of service networking for wide area measurement system. *IEEE Transactions on Industrial Informatics*, 16(5), 3018–3028.

Ricardo, S. (2016). A SDN controller architecture for Small Cell Wireless Backhaul using a LTE Control Channel/Ricardo Santos, Andreas Kassler. *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2016 IEEE 17th International Symposium on A.–28 July*.

Ruckus. (Accessed: July 16, 2019). Introducing the smart campus. Consulted at https://webresources.ruckuswireless.com/pdf/solution-briefs/sb-smartcampus-ebook.pdf.

Saha, N., Bera, S. & Misra, S. (2018). Sway: Traffic-Aware QoS Routing in Software-Defined IoT. *IEEE Transactions on Emerging Topics in Computing*, 1–1. doi: 10.1109/TETC.2018.2847296.

Santos, R. & Kassler, A. (2017). Small cell wireless backhaul reconfiguration using software-defined networking. *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6.

Saunders, J. D., McClure, C. R. & Mandel, L. H. (2012). Broadband applications: Categories, requirements, and future frameworks. *First Monday*, 17(11).

Schulz, P., Matthe, M., Klessig, H., Simsek, M., Fettweis, G., Ansari, J., Ashraf, S. A., Almeroth, B., Voigt, J., Riedel, I. et al. (2017). Latency critical IoT applications in 5G: Perspective on the design of radio interface and network architecture. *IEEE Communications Magazine*, 55(2), 70–78.

Seppänen, K., Kilpi, J. & Suihko, T. (2014). Integrating WMN based mobile backhaul with SDN control. *International Internet of Things Summit*, pp. 222–233.

Serpanos, D. & Wolf, M. (2017). *Internet-of-things (IoT) systems: architectures, algorithms, methodologies*. Springer.

Shafiq, M., Tian, Z., Bashir, A. K., Jolfaei, A. & Yu, X. (2020). Data mining and machine learning methods for sustainable smart cities traffic classification: A survey. *Sustainable Cities and Society*, 60, 102177.

Shantharama, P., Thyagaturu, A. S., Karakoc, N., Ferrari, L., Reisslein, M. & Scaglione, A. (2018). LayBack: SDN management of multi-access edge computing (MEC) for network access services and radio resource sharing. *IEEE Access*, 6, 57545–57561.

Shibl, M., Ismail, L. & Massoud, A. (2020). Machine learning-based management of electric vehicles charging: Towards highly-dispersed fast chargers. *Energies*, 13(20), 5429.

Shu, Z., Wan, J., Lin, J., Wang, S., Li, D., Rho, S. & Yang, C. (2016). Traffic engineering in software-defined networking: Measurement and management. *IEEE Access*, 4, 3246–3256.

Sisinni, E., Saifullah, A., Han, S., Jennehag, U. & Gidlund, M. (2018). Industrial internet of things: Challenges, opportunities, and directions. *IEEE Transactions on Industrial Informatics*, 14(11), 4724–4734.

Sivanathan, A., Sherratt, D., Gharakheili, H. H., Radford, A., Wijenayake, C., Vishwanath, A. & Sivaraman, V. (2017). Characterizing and classifying IoT traffic in smart cities and campuses. *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 559–564.

Sivanathan, A., Gharakheili, H. H., Loi, F., Radford, A., Wijenayake, C., Vishwanath, A. & Sivaraman, V. (2018). Classifying IoT devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, 18(8), 1745–1759.

Sivanathan, A., Gharakheili, H. H. & Sivaraman, V. (2019). Inferring iot device types from network behavior using unsupervised clustering. *2019 IEEE 44th Conference on Local Computer Networks (LCN)*, pp. 230–233.

Sivanathan, A., Gharakheili, H. H. & Sivaraman, V. (2020). Managing IoT cyber-security using programmable telemetry and machine learning. *IEEE Transactions on Network and Service Management*, 17(1), 60–74.

Softpedia. (Accessed: October 20, 2020). PlayCap Application. Consulted at https://www.softpedia.com/get/Multimedia/Audio/Audio-Players/PlayCap.shtml.

Staudemeyer, R. C. (2015). Applying long short-term memory recurrent neural networks to intrusion detection. *South African Computer Journal*, 56(1), 136–154.

Sun, G., Chen, T., Su, Y. & Li, C. (2018). Internet traffic classification based on incremental support vector machines. *Mobile Networks and Applications*, 23(4), 789–796.

Sutjarittham, T., Gharakheili, H. H., Kanhere, S. S. & Sivaraman, V. (2019). Experiences with IoT and AI in a Smart Campus for Optimizing Classroom Usage. *IEEE Internet of Things Journal*.

Talebkhah, M., Sali, A., Marjani, M., Gordan, M., Hashim, S. J. & Rokhani, F. Z. (2021). IoT and Big Data Applications in Smart Cities: Recent Advances, Challenges, and Critical Issues. *IEEE Access*, 9, 55465–55484.

Taylor, V. F., Spolaor, R., Conti, M. & Martinovic, I. (2017). Robust smartphone app identification via encrypted network traffic analysis. *IEEE Transactions on Information Forensics and Security*, 13(1), 63–78.

Thames-Water. (Accessed: October 20, 2020). IoT Now - how to run an IoT enabled business. Consulted at https://www.iot-now.com/tag/thames-water/.

Thapa, K. N. K. & Duraipandian, N. (2021). Malicious Traffic classification Using Long Short-Term Memory (LSTM) Model. *Wireless Personal Communications*, 1–18.

Tsironi, E., Barros, P., Weber, C. & Wermter, S. (2017). An analysis of convolutional long short-term memory recurrent neural networks for gesture recognition. *Neurocomputing*, 268, 76–86.

UNIBS, T. (Accessed: October 20, 2020). University of Brescia. Consulted at http://netweb.ing.unibs.it/~ntw/tools/traces/.

UNSW, T. (Accessed: October 20, 2020). University of New South Wales. Consulted at https://iotanalytics.unsw.edu.au/iottraces.html.

UPC, T. (Accessed: October 20, 2020). Universitat Politcnica de Catalunya. Consulted at https://cba.upc.edu/monitoring/traffic-classification.

Uskov, V., Pandey, A., Bakken, J. P. & Margapuri, V. S. (2016). Smart engineering education: The ontology of Internet-of-Things applications. *2016 IEEE Global Engineering Education Conference (EDUCON)*, pp. 476–481.

Uskov, V. L., Howlett, R. J. & Jain, L. C. (2015). *Smart education and smart e-learning*. Springer International Publishing Switzerland. Consulted at https://doi.org/10.1007/978-3-319-19875-0.

Vitturi, S., Zunino, C. & Sauter, T. (2019). Industrial communication systems and their future challenges: next-generation Ethernet, IIoT, and 5G. *Proceedings of the IEEE*, 107(6), 944–961.

vXchnge. (Accessed: September 12, 2021). Comprehensive Guide to IoT Statistics You Need to Know in 2021. Consulted at https://www.vxchnge.com/blog/iot-statistics.

Wan, J., Tang, S., Shu, Z., Li, D., Wang, S., Imran, M. & Vasilakos, A. V. (2016). Software-defined industrial internet of things in the context of industry 4.0. *IEEE Sensors Journal*, 16(20), 7373–7380.

Wan, J., Tang, S., Li, D., Wang, S., Liu, C., Abbas, H. & Vasilakos, A. V. (2017). A manufacturing big data solution for active preventive maintenance. *IEEE Transactions on Industrial Informatics*, 13(4), 2039–2047.

Wang, K., Wang, Y., Sun, Y., Guo, S. & Wu, J. (2016). Green industrial Internet of Things architecture: An energy-efficient perspective. *IEEE Communications Magazine*, 54(12), 48–54.

Wang, Z. (2001). *Internet QoS: architectures and mechanisms for quality of service*. Morgan Kaufmann.

Wollschlaeger, M., Sauter, T. & Jasperneite, J. (2017). The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0. *IEEE industrial electronics magazine*, 11(1), 17–27.

Wu, J., Dong, M., Ota, K., Li, J., Yang, W. & Wang, M. (2019). Fog-computing-enabled cognitive network function virtualization for an information-centric future Internet. *IEEE Communications Magazine*, 57(7), 48–54.

Xia, W., Wen, Y., Foh, C. H., Niyato, D. & Xie, H. (2015). A survey on software-defined networking. *IEEE Communications Surveys & Tutorials*, 17(1), 27–51.

Xiang, W., Wang, N. & Zhou, Y. (2016). An energy-efficient routing algorithm for software-defined wireless sensor networks. *IEEE Sensors Journal*, 16(20), 7393–7400.

Xiao, Y., Thulasiraman, K., Xue, G. & Yadav, M. (2016). QoS Routing Under Multiple Additive Constraints: A Generalization of the LARAC Algorithm. *IEEE Transactions on Emerging Topics in Computing*, 4(2), 242-251.

Xu, H., Yu, W., Griffith, D. & Golmie, N. (2018). A survey on industrial Internet of Things: A cyber-physical systems perspective. *IEEE Access*, 6, 78238–78259.

Yamansavascilar, B., Guvensan, M. A., Yavuz, A. G. & Karsligil, M. E. (2017). Application identification via network traffic classification. *2017 International Conference on Computing, Networking and Communications (ICNC)*, pp. 843–848.

Yang, A.-M., Li, S.-S., Ren, C.-H., Liu, H.-X., Han, Y. & Liu, L. (2018). Situational awareness system in the smart campus. *IEEE Access*, 6, 63976–63986.

Yang, H., Yuan, J., Li, C., Zhao, G., Sun, Z., Yao, Q., Bao, B., Vasilakos, A. V. & Zhang, J. (2021). BrainIoT: Brain-Like Productive Services Provisioning with Federated Learning in Industrial IoT. *IEEE Internet of Things Journal*, 1-1. doi: 10.1109/JIOT.2021.3089334.

Yen, J. Y. (Accessed: October 20, 2020). Yens K-shortest paths algorithm. Consulted at https://github.com/guilhermemm/k-shortest-path.

Yu, T.-F., Wang, K. & Hsu, Y.-H. (2015). Adaptive routing for video streaming with QoS support over SDN networks. *Information Networking (ICOIN), 2015 International Conference on*, pp. 318–323.

Zanella, A., Bui, N., Castellani, A., Vangelista, L. & Zorzi, M. (2014). Internet of things for smart cities. *IEEE Internet of Things journal*, 1(1), 22–32.

Zhang, C. & Ji, W. (2019). Big Data Analysis Approach for Real-Time Carbon Efficiency Evaluation of Discrete Manufacturing Workshops. *IEEE Access*, 7, 107730–107743.

Zhu, J., Hua, J., Liu, M., Li, Y. & Cao, K. (2020). TRUS: Towards the Real-Time Route Update Scheduling in SDN for Data Centers. *IEEE Access*, 8, 68682-68694.

Zoppi, S., Van Bemten, A., Gürsu, H. M., Vilgelm, M., Guck, J. & Kellerer, W. (2018). Achieving hybrid wired/wireless industrial networks with WDetServ: Reliability-based scheduling for delay guarantees. *IEEE Transactions on Industrial Informatics*, 14(5), 2307–2319.