Multi-Dimensional Network Slicing: Algorithmic and Learning-based Methods

Inaugural Dissertation
of the Faculty of Science,
University of Bern

presented by

Jesutofunmi Ademiposi Ajayi

from Ondo, Nigeria

Supervisor

Prof. Dr. Torsten Braun
Institute of Computer Science
Faculty of Science of the University of Bern, Switzerland

Multi-Dimensional Network Slicing: Algorithmic and Learning-based Methods

Inaugural Dissertation
of the Faculty of Science,
University of Bern

presented by

Jesutofunmi Ademiposi Ajayi

from Ondo, Nigeria

Supervisor

Prof. Dr. Torsten Braun
Institute of Computer Science
Faculty of Science of the University of Bern, Switzerland

Accepted by the Faculty of Science.

Bern, May 2025

The Dean:
Prof. Dr. Jean-Louis Reymond

Soli Deo gloria.

Acknowledgements

I would like to express my gratitude to several persons and organizations that assisted in one way or another during this challenging, but rewarding, educational journey.

First, I would like to thank Prof. Dr. Torsten Braun, my *Doktorvater*, who took a great interest in my educational development and entrusted me to challenge myself personally and academically by carrying out my Ph.D. research in the Communication and Distributed Systems (CDS) Group at the University of Bern. His constant academic guidance, patience, support, and belief during my research and thesis writing were crucial to the completion of this dissertation and are very much appreciated. I am forever indebted to him for everything he's done for me since being a Master's student with him till now - thank you, Sir. I am also grateful to Prof. Marilia Curado for agreeing to be my second Ph.D. supervisor and for her feedback on the thesis.

I would like to thank all my colleagues who accompanied me at the University of Bern. Through discussions, interactions, and advice, they helped to shape my academic journey for the better.

In no particular order, I would like to thank: Eryk Schiller, Antonio Di Maio, Eric Samikwa, Dimitris Xenakis, Alisson Medeiros, Lucas Pacheco, Diego Oliveira, Hugo Santos, Daniela Schroth, Priska Grunder, Bettina Choffat, Stefanie Feuz and Prof. Dr. Burkhard Stiller, for their invaluable support at one stage or another of my educational journey. Special mention goes out to Prof. Dr. Peter Kropf and Prof. Dr. Zoltan Balogh for their crucial support during my studies. I would also like to extend my appreciation to present and former members of the CDS group that I had to the pleasure to meet or collaborate with during the duration of my Ph.D.

A special appreciation is reserved for the Zysset family in Rubigen, Hegg family in Ostermundigen and Haldimann family in Zurich for their never-ending hospitality. I also would like to thank my religious and social communities in Bern, namely, Riverlife Church and International Students Bern (ISB).

I also say «Thank you» to several other individuals who assisted in one way or the other but who are too numerous to be named individually.

Most importantly, I am grateful to my family, Oluyede, Adeola, Oluwasooto, Oluwatorole, and Emmanuella, to whom I owe a huge debt of gratitude for their constant support and sacrificial love throughout my Ph.D. journey. They never stopped encouraging me, challenging me, and

believing in me, and to them I say: E ṣeun, Merci, Zikomo, Asante, Bedankt, Danke - Thank you. My extended family of uncles, aunts, and cousins has also provided me with a network of support and love during this time, thank you all.

Abstract

Next-generation mobile networks are expected to support a wide range of demanding applications and services that have strict and varying performance requirements. To meet these requirements, Network Slicing (NS) has emerged as a powerful technique to enable cost-effective, multi-tenant communications and services over a shared physical mobile network infrastructure. However, the effective realization of the NS paradigm hinges on the ability to manage the end-to-end lifecycle of network slices in a dynamic, efficient, and automated manner. This challenge is exacerbated by the multi-dimensional slice requirements such as bandwidth, latency, and CPU and the unpredictable, online arrival of slice requests.

To address the challenges of managing the end-to-end lifecycle of network slices, this thesis proposes online, data-driven solutions that are capable of adapting to dynamic conditions in mobile networks by optimizing multi-dimensional resource allocations, enabling scalable, real-time slice orchestration across heterogeneous infrastructures and proactively optimizing network slice performance to ensure service-level compliance.

First, we investigate the slice admission control problem, focusing on the setting where slice requests arrive sequentially and must be admitted or rejected in real-time without prior knowledge of future resource demands. We propose an online algorithm that dynamically incorporates system resource utilization to guide admission decisions, and therefore resource allocations, with the aim of maximizing the long-term revenue of infrastructure providers.

Second, building on this foundation, we address the problem of online policy selection under non-stationary network conditions. By modeling the policy selection task as a multi-armed bandit problem, we propose a data-driven solution that learns to select the most effective admission policy across time-varying network conditions. This approach balances exploration and exploitation while detecting and reacting to changes in environmental dynamics.

Third, we address the problem of scalable slice provisioning in large-scale, distributed networks, and propose a hierarchical solution for the online network slice provisioning problem in which a service function chain must be effectively mapped onto the network infrastructure, while optimizing for multiple objectives.

Finally, we propose a proactive optimization framework for the problem of allocating resources to heterogeneous network slices. The proposed framework aims to learn an effective resource allocation strategy in virtual radio access network environments by anticipating traffic fluctuations and proactively adjusting network slice resources for optimal performance.

List of Figures

2.1 Service-Based Architecture in 5G Networks. 17
2.2 The lifecycle of a Network Slice Instance 19
2.3 Architectures for Addressing Dynamic Resource Allocation under Uncertainty (Adapted from [156]). 30

3.1 System Overview 38
3.2 Relative Gain for Linear Reservation Policy (LinRP) and Exponential Reservation Policy (ExpRP) against First Come First Serve (FCFS) for different values of ω and $\sigma = 10$. 45

4.1 Policy Selection Problem for Slice Admission Control (SAC) 49
4.2 Drift-AwaRe upper confIdence bOund (DARIO) Workflow 54
4.3 Average Revenue Relative Gain for DARIO against the Static and Adaptive baselines. 59
4.4 Acceptance Ration Relative Gain for DARIO against the Static and Adaptive baselines. 59
4.5 Average Resource Utilization Relative Gain for DARIO against the Static and Adaptive baselines. 60

5.1 System Model 65
5.2 HELIOS Architecture. 69
5.3 Example Clusters in an Abilene Topology. 70
5.4 Request Acceptance Ratio over the (A) GEANT and (B) DT2 Topologies when $\lambda = 2$, $\lambda = 5$ and $\zeta = 5$. 78
5.5 Average Resource Utilization and Cluster-based Performance over the GEANT topology. 79
5.6 Average Resource Utilization and Cluster-based Performance over the DT2 topology. 79

6.1 O-RAN System Model 86
6.2 Per slice aggregate traffic demands 90
6.3 Training and Validation Loss 93
6.4 Average Reward: Proactive Resource Optimization for Heterogeneous nETwork slices (PROPHET) vs Standard PPO 94

List of Tables

2.1 Related Works: Policy Selection for SAC 26
2.2 Related Works: Network Slice Provisioning 30

3.1 Summary of Notations 36

4.1 Summary of Notations 50

5.1 Simulation Parameter Settings 76
5.2 Avg. Execution Times of Baseline Algorithms and Hierarchical nEtwork sLIce prOviSioning (HELIOS) [s] 80

6.1 Dataset Features and Output 90
6.2 Attention-LSTM Parameters 91
6.3 PPO Hyperparameters 91
6.4 Forecasting Performance Metrics 92

List of Acronyms

AC Admission Control.

AI Artificial Intelligence.

AR Acceptance Ratio.

C-MAB Contextual Multi-Armed Bandit.

CB Contextual Bandit.

CD Concept Drift.

CN Core Network.

COMO-MAB Combinatorial Multi-Objective Multi-Armed Bandit.

COMO-UCB Combinatorial Multi-Objective Upper-Confidence Bound.

DARIO Drift-AwaRe upper confIdence bOund.

DL Deep Learning.

DNN Deep Neural Network.

DRL Deep Reinforcement Learning.

E2E End-to-End.

EC Edge Computing.

eMBB enhanced Mobile Broadband.

EPC Evolved Packet Core.

ExpRP Exponential Reservation Policy.

FCFS First Come First Serve.

GGF Generalized Gini Function.

GGI Generalized Gini Index.

HLA High-Level Agent.

HMAB Hierarchical Multi-Armed Bandit.

ILP Integer Linear Program.

InP Infrastructure Provider.

IoT Internet-of-Things.

LLA Low-Level Agent.

LSTM Long Short-Term Memory.

MAB Multi-Armed Bandit.

MANO Management & Orchestration.

MDP Markov Decision Process.

MEC Mobile Edge Computing.

ML Machine Learning.

mMTC massive Machine Type Communication.

MO-MAB Multiple-Objective Multi-Armed Bandit.

MOO Multiple-Objective Optimization.

NaaS Network-as-a-Service.

NF Network Function.

NFV Network Function Virtualization.

NGMN Next-Generation Mobile Network.

NN Neural Network.

NS Network Slicing.

NSaaS Network Slice-as-a-Service.

NSL Network Slice.

NSP Network Slice Provisioning.

NSR Network Slice Request.

O-RAN Open Radio Access Network.

OCO Online Convex Optimization.

OGA Online Gradient Ascent.

OL Online Learning.

OMdKP Online Multidimensional Knapsack Problem.

ONSP Online Network Slice Provisioning.

OPEX Operating Expenditure.

OSAC Online Slice Admission Control.

OWA Ordered Weighted Averaging.

PF Pareto Front.

PNF Physical Network Function.

PRB Physical Resource Block.

QoE Quality of Experience.

QoS Quality of Service.

RAN Radio Access Network.

RL Reinforcement Learning.

SAC Slice Admission Control.

SACPS Slice Admission Control Policy Selection.

SB Single Best.

SBA Service-Based Architecture.

SDN Software Defined Networking.

SFC Service Function Chain.

SLA Service Level Agreement.

SoTA State-of-The-Art.

ST Slice Tenant.

SVR Support Vector Regression.

SW Sliding Window.

SW-UCB Sliding-Window Upper Confidence Bound.

UCB Upper Confidence Bound.

UE User Equipment.

uRLLC ultra-Reliable and Low-Latency Communication.

UV Unit Value.

V2X Vehicle-to-Everything.

VM Virtual Machine.

VNF Virtual Network Function.

VR Virtual Reality.

vRAN Virtual Radio Access Network.

WTPR Willingness-To-Pay-Ratio.

XAI eXplainable Artificial Intelligence.

Contents

Acknowledgements **i**

List of Figures **iv**

List of Tables **v**

List of Acronyms **vi**

1 Introduction **1**
1.1 Overview 1
1.2 Motivation 2
1.3 Problem Statement 5
1.4 Thesis Contributions 9
1.5 Thesis Outline 14

2 Background and Related Works **15**
2.1 Background 15
2.2 Related Works 24
2.3 Chapter Conclusions 32

3 Multi-dimensional Slice Admission Control **34**
3.1 Introduction 34
3.2 System Model and Problem Formulation 36
3.3 Online Slice Admission Control 38
3.4 Performance Evaluation 42
3.5 Chapter Conclusions 46

4 Data-Driven Online Policy Selection **47**
4.1 Introduction 47
4.2 System Model and Problem Formulation 49
4.3 Drift-Aware Policy Selection 51
4.4 Performance Evaluation 57
4.5 Chapter Conclusions 61

5 Hierarchical Network Slice Provisioning **62**

5.1 Introduction . . . 62
5.2 System Model and Problem Formulation . . . 64
5.3 Hierarchical Placement Learning . . . 68
5.4 Performance Evaluation . . . 76
5.5 Chapter Conclusions . . . 81

6 SLA-Driven Proactive Slice Optimization **82**
6.1 Introduction . . . 82
6.2 System Model and Problem Formulation . . . 84
6.3 Proactive Slice Optimization . . . 88
6.4 Performance Evaluation . . . 89
6.5 Chapter Conclusions . . . 93

7 Conclusions and Future Work **95**
7.1 Summary of Contributions . . . 95
7.2 Future Work . . . 100

List of Publications **104**

Bibliography **106**

Chapter 1

Introduction

1.1 Overview

While third generation (3G) and fourth generation (4G) cellular networks were primarily driven by the need for high data-rates and better network coverage, Next-Generation Mobile Networks (NGMNs) (i.e. Fifth Generation - 5G - and beyond 5G) will be required to provide enhanced support for applications and services that have stringent Quality of Service (QoS) and Quality of Experience (QoE) requirements [1], [2]. Towards this, the main focus in the ongoing development of 5G networks has been on increasing the network capacity and availability, improving the connection density, enhancing network reliability, and reducing the overall end-to-end (E2E) network latency [3]. The adoption of Cloud, Edge and Fog computing into the network architecture, as well as recent advancements in Network Function Virtualization (NFV) and Software Defined Networking (SDN), have also stood out as essential developments that can be used to create new opportunities that meet current and future requirements of novel applications and services. More specifically, the combination of heterogeneous technologies enables the realization of Network Slicing (NS) in NGMNs, where the vision is to facilitate the on-demand creation and instantiation of multiple logical networks that share a common network infrastructure, and where each logical network (i.e., a Network Slice (NSL)) is tailored towards a particular use case. Network Slicing technology enables network resources to be dynamically allocated and shared across a diverse range of services, applications and network operators [4]. However, due to the unpredictable nature of mobile traffic, slice requests and service requirements, especially at the edge, there is a need to develop novel solutions that can overcome the limitations of traditional approaches used for network optimization, which typically assume full and static knowledge about the behavior of the network environment, while ensuring that the diverse objectives in provisioning a NSL are met.

Over the past decade, advancements in Machine Learning (ML) and Artificial Intelligence (AI) have rapidly evolved and found applications in a wide range of domains, including mobile communication networks, such as 5G. As a result, data-driven techniques such as Deep Learning (DL), Reinforcement Learning (RL), Random Forests (RFs), Decision Trees (DTs), XGBoost, and Support Vector Machines (SVMs) have been applied to address challenges in

mobile network management, including network optimization, traffic forecasting, mobility management, intrusion and anomaly detection, and cognitive network management, thereby improving network performance and the user experience [5]. As mobile networks evolve from 5G to 6G, such intelligent techniques will likely become an omnipresent feature of such networks, enabling ubiquitous intelligence by penetrating every aspect of the network fabric to fulfill the diverse requirements of novel applications and services [6].

Despite the general success of Machine Learning (ML)-based solutions in networking related problems, such as enhanced transmission reliability and better responses to changing network conditions and demands, the inherent uncertainty of ML-based algorithms, especially in complex and dynamically changing network environments, raises concerns about their reliability and suitability in autonomously managing or operating future networks [5]. Moreover, many ML-based techniques require extensive training on large datasets to learn management or orchestration policies, thereby hindering their adoption in real-time network management due to long training times and a lack of representative datasets. Overcoming these limitations is essential for enabling fully autonomous, intelligent networks that adapt to environmental dynamics.

1.2 Motivation

Fifth Generation mobile networks are now the de-facto Radio Access Technology (RAT) for cellular communications. However, adequately supporting the wide range of upcoming 5G-enabled services and applications over the same network infrastructure will be a significant challenge [7], and requires enhancements to the current network architecture to enable greater flexibility and ubiquity towards the realization of fully programmable networks [8]. Consequently, network slicing has been proposed as a virtualization-enabled and software-defined technique in which physical network resources (*i.e.,* compute, storage, bandwidth, or physical resource blocks) can be virtualized and deployed to enable multi-tenancy, greater customizability, and better overall network performance, while reducing Operating Expenditure (OPEX) and Capital Expenditure (CAPEX) for infrastructure providers. The network slicing paradigm follows the Service-Based Architecture (SBA) of softwarized 5G network designs and can be delivered as part of a Network-as-a-Service (NaaS) computing model - Network Slice-as-a-Service (NSaaS) - as it provides a low-cost and logically isolated network over the physical infrastructure by splitting it into multiple instances, where each Network Slice Instance (NSI) is tailored towards a particular service based on the demands of a Slice Tenant (ST).

Building on this paradigm, the integration of AI/ML into mobile network optimization is a key enabler of efficient, adaptive, and automated network-slice lifecycle management. AI-driven approaches leverage data and advanced learning methods to optimize resource allocation, forecast network demand, and dynamically reconfigure slices to meet diverse application and service requirements [9]. Within the NSaaS paradigm, AI enables Infrastructure Providers (InPs) to make more informed decisions, proactively optimize resources, and

improve QoS, thereby delivering a more resilient and cost-effective multidimensional network-slicing framework

Multi-Dimensional Resource Management. Network slicing in NGMNs introduces substantial complexity in resource management because multiple resource dimensions must be optimized concurrently across network domains and protocol layers. A NSL may require reserving and isolating specific computing resources (Central Processing Unit (CPU), Graphic Processing Unit (GPU), Tensor Processing Unit (TPU)), memory, storage capacity, physical resource blocks, bandwidth, and specialized network functions to satisfy stringent latency and reliability requirements [10], [11]. The resulting multi-dimensional resource landscape creates a high-dimensional, tightly constrained optimization space [12], where decisions must account for intricate inter-dependencies among heterogeneous resource types, rendering classical optimization approaches inadequate and slow for operational timescales. For example, in admission control for network slicing, classical optimization methods struggle to jointly optimize CPU, memory, storage, and bandwidth in real time; the computational burden grows exponentially with each added dimension, highlighting the need for efficient online algorithms that handle multi-dimensional constraints while enabling rapid decision making [13]. The challenge is further compounded by infrastructure heterogeneity, as varying hardware capabilities and capacity limits must be considered in resource allocation and SAC decisions.

In NGMNs, Network Slice Requests (NSRs) arrive in an online manner with heterogeneous resource demands, revenue profiles, and Service Level Agreements (SLAs), necessitating real-time decisions that balance immediate resource reservation or allocation against preserving scarce resource capacity for future demand [14]. The challenge intensifies when NSRs differ in duration or exhibit elastic scaling requirements, requiring admission-control mechanisms that can commit resources promptly while safeguarding capacity for potentially higher-value future requests [15]. As network slicing becomes central to supporting diverse applications, services, and industry verticals in next-generation mobile networks, the design of efficient multi-dimensional resource-reservation and allocation strategies remains a critical research priority.

Non-Stationary Networks and Uncertainty. The provisioning of network slices in NGMNs faces several challenges that are a result of non-stationary conditions such as fluctuating traffic patterns, unpredictable resource availability, and dynamic user mobility. The dynamic nature of such networks makes it difficult to obtain optimal resource allocation decisions or maintain consistent and guaranteed performance across NSs [16], [17]. This non-stationarity introduces significant uncertainty that challenges traditional optimization approaches which assume static or predictable network conditions. In the NS context, where multiple virtual networks share the same physical infrastructure, these uncertainties are magnified as changes in one slice can quickly and easily be propagated and affect the resource availability for co-deployed NSLs. The problem is exacerbated when SLAs must be maintained despite these fluctuating conditions, requiring adaptive resource allocation and admission control mechanisms that can respond to both gradual trends and sudden network changes in the network states without prior knowledge of future conditions.

To address these challenges, frameworks that integrate predictive analytics, robust optimization, and machine learning are required. Methods from Stochastic Optimization (SO) [18]–[20], Online Convex Optimization (OCO) [21], [22], Bayesian Optimization (BO) [23], [24], and RL [25]–[27], often augmented with uncertainty estimation [28]–[30], have been leveraged to develop robust, resilient resource-optimization frameworks for NS. These approaches optimize under current network conditions while explicitly accounting for distributions over possible future states, yielding strategies that remain effective across a broad range of scenarios. Ultimately, the goal is to realize self-organizing NSLs that adapt rapidly and autonomously to non-stationary conditions while maintaining SLA guarantees, ensuring efficient resource utilization, and promoting fair allocation among co-deployed slices — capabilities that are critical for NGMNs to support diverse applications and services.

Scalability Challenges in Network Slicing. The provisioning of NSLs faces critical scalability challenges, as InP attempt to instantiate, configure, and deploy NSLs at the pace demanded by diverse vertical industries and use cases. The time-sensitive nature of network slice provisioning - which involves resource reservation, Virtual Network Function (VNF) placement, Service Function Chain (SFC) composition, and policy configuration - creates significant operational challenges when scaled to support hundreds or thousands of concurrent flows from diverse NSRs [31]–[33]. In particular, as the granularity of concurrent flows, such as provisioning resources at the per-user or per-application level, increases, the complexity and scale of operational tasks grow, which further increases the challenges involved in real-time network slice provisioning. Traditional provisioning workflows often rely on manual verification steps and sequential processes that introduce unacceptable delays in service activation, particularly for dynamic use cases, requiring the establishment of NSLs in near-real-time. These inefficiencies become more pronounced as NSRs increase in volume and complexity, creating a fundamental tension between provisioning speed and the assurance of NSL isolation and performance guarantees. Moreover, the process of provisioning NSLs requires the consideration of multiple potentially conflicting objectives [34], [35], including: Maximizing the revenue gained from provisioning the requests and ensuring that the QoS requirements of the requests are met during their lifetimes in the network. These objectives can typically be achieved by maximizing the number of accepted NSRs, maximizing or minimizing the resource utilization rate in the network, reducing the provisioning times, and minimizing the number of migrations that occur as a result of accepting a request [35].

The introduction of edge computing into the mobile network infrastructure, characterized by distributed, resource-constrained devices with heterogeneous capabilities, creates unique slice provisioning challenges that render traditional centralized approaches ineffective [36]. More specifically, the distributed nature of edge networks further complicates the NSL provisioning task at scale, requiring complex orchestration across multiple technology and administrative domains and systems, where each network domain could introduce its own provisioning latency, resource discovery mechanisms, and configuration interfaces. This creates cumulative delays that impact the end-to-end slice deployment time. Hence, achieving scalable slice provisioning remains a prerequisite for realizing the commercial potential of NS in NGMNs, particularly for

supporting emerging applications requiring rapid service instantiation and reconfiguration in response to changing business requirements or network conditions.

Ubiquitous Network Intelligence. The intelligent management of mobile networks represents a paradigm shift in how NSLs are orchestrated, optimized, and maintained across increasingly complex network infrastructures. As NSLs are deployed to support diverse vertical applications and services with heterogeneous requirements, traditional rule-based management approaches become overwhelmingly complex and inefficient, creating the need for AI-driven automation approaches. This is further emphasized by the goal of ubiquitous intelligence as one of the most important features of NGMNs, as it will be able to penetrate various parts of NS systems [37] to enable the simultaneous monitoring of performance metrics across multiple NSLs which would identify complex inter-slice dependencies, detect anomalous behaviors, and predict potential service degradations before they impact end-users. Through intelligent monitoring of the ongoing network conditions, proactive resource allocation [38] and slice capacity adjustments can be made to ensure that the SLAs of NSL requests are being continuously met and ensure that networks are able to autonomously predict, detect, and respond to complex operational challenges without strict human intervention [39], [40].

The ongoing evolution toward Zero-Touch Management (ZTM) through intelligent systems creates opportunities for significant operational cost reductions and enhanced service reliability. However, the lack of 5G/6G specific datasets creates challenges regarding the evaluation of intelligent techniques for ZTM [38] in such networks. Hence, there's a need to build frameworks that integrate AI for more accurate prediction, with dynamic optimization of slice resources by leveraging realistic networking datasets to evaluate the performance of such solutions.

1.3 Problem Statement

The increasing demand for NS in 5G and beyond requires the development of intelligent and efficient resource allocation strategies to ensure service quality and optimize resource efficiency. However, the dynamic nature of network slice requests, coupled with the multi-dimensional resource constraints in mobile network and time-varying service demands, presents significant challenges in efficient slice management. While traditional static optimization approaches have typically been used in the previous generations of mobile networks (i.e., 3G and 4G), the dynamic nature of NGMNs, such as 5G mobile networks, means that such optimization techniques often struggle to adapt to changing network conditions and cannot produce optimal solutions to various network slice management problems within an acceptable time, which leads to suboptimal resource allocation and efficiency. Furthermore, the complexity of managing multiple resource dimensions simultaneously, such as CPU, GPUs, TPUs, storage, and bandwidth, makes it challenging to make real-time admission, placement and configuration decisions while maintaining the QoS guarantees of NSs.

In this thesis, we seek to develop novel strategies that address these challenges by focusing on the need for solutions that lead to timely decisions in dynamic and uncertain environments, such as the network edge. Specifically, we focus on challenges relating to the admission of

network slice requests, the placement of such requests in distributed edge networks to meet multiple provisioning objectives and the need for predictive techniques to continually optimize the capacity of the deployed network slices to satisfy slice SLAs.

In the following subsections, we provide a detailed description of the main challenges raised by dynamic network slicing and present the research questions that we have designed to guide our research.

1.3.1 Online Resource Reservation for Network Slice Requests

Online resource reservation in NS systems presents a fundamental challenge of decision-making under uncertainty, where InPs must *commit* resources to NSLs without complete knowledge of future requests and their demands. This problem is defined by the inherent trade-off between proactive resource reservation to ensure service availability and reactive resource allocation to optimize for resource efficiency. InPs face the challenging task of determining optimal reservation strategies across multiple resource dimensions (e.g., computing, storage, bandwidth) while potentially accounting for temporal demand variations, service priority levels, and economic considerations. The over reservation of resources leads to poor resource efficiency and increased OPEX, while under reserving resources risks service degradation and SLA violations during demand spikes. The challenge is further complicated by the heterogeneity of NSL requirements, ranging from ultra-Reliable and Low-Latency Communication (uRLLC) services which demand stringent resource guarantees to massive Machine Type Communication (mMTC) applications where statistical multiplexing might be more appropriate.

To address the above issues, we formulate the following research questions, which focus on designing an online admission control solution that jointly optimizes resource efficiency, long-term revenue, and fairness under heterogeneous and uncertain network slicing demands:

RQ 1.1: How can admission control policies effectively manage the **multi-dimensional** resource demands of network slices while maintaining high resource efficiency in dynamic mobile networks?

RQ 1.2 How can online admission control policies ensure long-term **revenue optimization** in network slicing, while accommodating heterogeneous slice requirements and uncertain demand?

RQ 1.3: How does **economic disparity** among slice tenants influence admission control decisions, and how can admission control policies balance revenue optimization with fairness?

1.3.2 Data-Driven Policy Selection in Network Slicing

Network management in the context of network slicing must operate in highly dynamic environments characterized by continuous and unpredictable changes in resource utilization patterns, user demands, and operational conditions - a phenomenon that is captured by Concept Drift (CD). Concept drift refers to the change in the statistical properties of the target variable or the underlying data distribution over time, which can degrade the performance of predictive models if not properly addressed [41]. The existance of such a phenomenon in mobile

networks renders static resource allocation and admission control policies inherently ineffective, as they fail to capture the temporal dynamics of network behavior, leading to suboptimal resource utilization and potential service degradation when conditions inevitably change. In the context of network slice management, the existence of concept drift may arise from evolving traffic patterns, shifting service-level requirements, or the introduction of new applications and devices, necessitating adaptive policies and algorithms that can detect and respond to these changes in real-time to maintain efficient and reliable operation. With increasing network complexity and the proliferation of heterogeneous services, there is a critical need for intelligent, adaptive solutions that can monitor distributional changes in traffic and service demands and environmental conditions to continuously evolve their resource management strategies to reflect current network states. While traditional machine learning approaches have shown promising results for network management tasks such as traffic prediction and fault detection, they typically rely on historical datasets with assumed stationary distributions, which is an assumption that rarely holds in real-world production environments. The development of adaptive policies introduces additional challenges related to computational overhead, system stability preservation, and the fundamental exploration-exploitation dilemma, where systems must balance discovering better policy parameters against leveraging known effective strategies, based on their historical performance. Modern approaches increasingly focus on developing data-driven, Online Learning (OL) frameworks capable of adapting to non-stationary conditions while simultaneously maintaining strict service guarantees across diverse slice types - a capability essential for enabling resilient, self-organizing network slicing platforms that can support dynamic slice requests without manual intervention or performance degradation during environmental transitions in the network.

To address the above issues, we formulate the following research questions, which focus on developing adaptive admission control strategies that remain robust to shifting slice request patterns:

RQ 2.1: How can network slice admission control policies be **dynamically adapted** to cope with evolving network slice request patterns in 5G networks?

RQ 2.2: What is the effect of **temporal shifts** in the distribution of network slice request characteristics on the performance and robustness of slice admission control policies?

RQ 2.3: How can online learning techniques be integrated with change detection mechanisms to enable **continuous selection and adjustment** of admission control policy parameters for network slicing in dynamic environments?

1.3.3 Hierarchical Decision Framework for Network Slice Placement

The goal of allocating resources to diverse services and applications in NGMNs introduces significant complexity in resource management due to the need to optimize multiple resource dimensions (e.g., CPU, memory, storage, bandwidth) across various network domains and layers. Each NSL may require specific resources to meet strict latency and reliability requirements, while also maintaining isolation between slices. The multi-dimensional nature

of these resources, along with intricate dependencies and constraints, makes traditional optimization approaches inadequate and time-consuming. For example, in a large-scale distributed network, determining where to efficiently deploy sequentially arriving NSRs in real-time, presents computational challenges that grow exponentially with the number of arriving requests and the possible deployment options. This complexity is further compounded by the heterogeneous nature of network infrastructure due to integration of Edge Computing (EC) capabilities and the dynamically arriving NSRs with varying resource needs, SLAs, and lifetime durations. Optimizing the decisions relating to the real-time provisioning of NSLs must seek to balance immediate resource allocation with long-term availability, while ensuring that future NSRs, can also be deployed in the network. Therefore, scalable and data-driven NSL provisioning solutions are critical for the effective deployment of slice requests in edge-enabled NGMNs.

To address these challenges, the following research questions are formulated, which focus on designing a scalable network slice provisioning solution through hierarchical decision-making and multi-objective, fairness-aware performance evaluation:

RQ 3.1: How can we increase the rate of admitted NSLs while minimizing the amount of allocated network resources, especially in **large-scale** networks with highly **dynamic** resource requirements?

RQ 3.2: How can a **hierarchical model** be devised so that agents in different network subdomains make their own placement decisions?

RQ 3.3: How can we design a NS-provisioning performance metric that considers **multiple objectives** and enables a network policymaker to specify the objectives' relative **importance** and mutual **fairness**?

1.3.4 Proactive Resource Optimization under Traffic Uncertainty

In modern mobile networks, such as 5G, NS plays a crucial role in supporting diverse applications and services with varying resource demands and performance requirements. However, the challenge lies in the uncertainty of future demand for each slice, which can fluctuate due to changing traffic patterns, user behaviors, and network conditions. Traditional reactive approaches, which allocate resources only after demand manifests, often lead to inefficiencies, such as over-provisioning or under-provisioning resources, which can degrade QoS and lead to higher operating expenses due to unnecessary reconfiguration costs.

The proactive optimization of NSLs aims to address this challenge by anticipating future demand and allocating resources ahead of time to ensure optimal performance, while minimizing the amount of resources under-utilized. The main problem involves predicting future resource requirements or traffic volumes for each deployed NSL with a high accuracy, despite the inherent uncertainty in demand. This requires developing advanced forecasting models that can account for dynamic and unpredictable factors influencing network traffic. Furthermore, proactive optimization needs to balance the trade-off between minimizing the amount of resources unutilized (due to over-provisioning) and ensuring sufficient resources are

available to meet SLAs and latency constraints (due to under-provisioning). The proactive optimization of NSLs must consider the inter-dependencies between different network slices, as resource allocation in one slice can impact the performance of others that are co-deployed with it and share the same underlying infrastructure. Therefore, the development of efficient proactive slice optimization algorithms that can leverage predicted NSL demand to dynamically allocate resources and maintain service guarantees in the face of uncertainty is critical for achieving the goals of NS, ensuring resource efficiency, and maintaining QoS in NGMNs.

Based on the above observations, we formulate the following research questions, which focus on integrating traffic prediction with reinforcement learning to enable proactive and SLA-aware resource allocation in heterogeneous 5G slicing environments:

RQ 4.1: How can ML-based traffic **prediction** be integrated with **reinforcement learning** to enable **proactive resource allocation** in heterogeneous 5G network slicing environments?

RQ 4.2: To what extent does **forecasting accuracy** and prediction horizon impact the **effectiveness** of proactive resource allocation strategies in maintaining SLA while maximizing resource efficiency?

1.4 Thesis Contributions

The main contributions of this thesis are to develop algorithmic approaches and design learning-based frameworks that would enable quick, near-optimal decision-making and policy learning for network management tasks, within next-generation mobile networks. A typical trend within this thesis is to focus on the real-time nature of mobile networks and the associated uncertainty they embody due to inherent dynamism. Primarily, the objective of this thesis is to design solutions for common network slice provisioning problems such as, admission control, dynamic policy selection, network slice placement, and slice dimension optimization. We seek to address the previous research questions in Section 1.3 by designing novel algorithms and frameworks to improve the provisioning of NSLs, based on optimizing different objectives such as the number of accepted NSRs, the utilization of resources, the revenue gained from NSL, and the SLAs violated. The contributions of the presented works can be summarized as follows:

- Online Slice Admission Control
- Drift-Aware Policy Selection for Slice Admission Control
- Hierarchical Placement Learning for Network Slice Provisioning
- Proactive Slice Optimization for Heterogeneous Network Traffic

Through these contributions, this thesis provides a robust set of solutions that seek to address the algorithmic and modeling challenges commonly seen in NS problems, and by doing so, lays the foundation for the development of advanced, AI-driven mobile networks that can support a diverse range of applications and services.

1.4.1 Online Slice Admission Control

In our first contribution, we propose an Online Slice Admission Control (OSAC) approach that leverages online reservation policies for multi-dimensional resource evaluation in heterogeneous mobile networks. This approach maximizes the InP revenue despite the uncertainty around the requirements, value and duration of future NSRs. We demonstrate its effectiveness compared to other solutions, showing that it consistently outperforms benchmarks in key metrics like revenue maximization and resource efficiency, regardless of the chosen reservation function.

To address ***RQ 1.1*** and ***RQ 1.2***, we model the SAC problem as an Online Multidimensional Knapsack Problem (OMdKP) and implement two reservation policies: Linear Reservation Policy (LinRP) and Exponential Reservation Policy (ExpRP). These policies dynamically adjust admission thresholds based on current resource utilization, accounting for the heterogeneous resource dimensions in the network. By considering the scarcity in each resource dimension, the introduced policies ensure slice requests are only admitted if sufficient capacity exists and their value meets or exceeds resource costs. This approach enables our OSAC approach to maximize long-term revenue from admitted requests.

For ***RQ 1.3***, we model economic inequality with a parameter ω in a Beta distribution, where low ω values indicate high inequality. Our evaluation shows that, in high-inequality scenarios, OSAC selectively rejects lower-value requests to reserve capacity for higher-value ones. While this reduces acceptance ratios compared to greedy algorithms, it significantly boosts revenue. Economic inequality thus enables strategic admission control, reducing average resource utilization and improving allocation efficiency under network uncertainty.

By using online reservation policies, our OSAC algorithm reserves scarce network resources by dynamically updating the admission threshold for incoming slice requests. This results in a 12% increase in InP revenue and a 1.7% reduction in resource utilization. The formulated SAC problem, the description of the reservation policies and how they're leveraged in our algorithm are described thoroughly in Chapter 3.

1.4.2 Drift-Aware Policy Selection for Slice Admission Control

As the second contribution, this work presents the Drift-AwaRe upper confIdence bOund (DARIO) framework, which adaptively learns from historical request patterns to select near-optimal, online slice admission-control policies in real time. By framing Slice Admission Control Policy Selection (SACPS) as a Multi-Armed Bandit (MAB) problem, DARIO learns a policy that prioritizes high-value NSRs, which significantly boosts InP revenue. This contribution addresses the research questions in Section 1.3 by presenting a novel online-learning framework for SAC that adapts admission thresholds via drift detection on historical slice request patterns.

We address ***RQ 2.1*** by framing SACPS as a MAB problem. In this formulation, the bandit agent determines which admission control policy to apply for incoming network slice requests. We propose a Sliding-Window Upper Confidence Bound (SW-UCB)-based algorithm for adaptively selecting between policies, enabling online learning of their performance.

RQ 2.2 is addressed by analyzing how the statistical features of slice requests impact admission control policy performance. Based on these insights, we design an ADaptive WINdowing (ADWIN)-based change detection mechanism to identify shifts in statistical features of slice requests and trigger adaptive threshold adjustments.

For ***RQ 2.3***, we integrate the SW-UCB algorithm with the change detection mechanism to create a data-driven online framework. This framework learns from historical performance data while remaining responsive to environmental changes (e.g., fluctuations in the Willingness-To-Pay-Ratio (WTPR) - θ). It balances exploring different policies for performance information with exploiting known high-performing ones, especially when the request distribution is non-stationary.

By combining OL with Drift Detection (DD), our framework adapts to changing network conditions, captured by the characteristics of incoming slice requests. We compare DARIO's performance to baseline solutions, including LinRP, ExpRP, greedy FCFS, and a basic Upper Confidence Bound (UCB) algorithm without change detection. Our results show that DARIO outperforms all benchmarks, achieving a 4.5% higher revenue gain while maintaining efficient resource utilization. This success essentially stems from DARIO's ability to detect deviations in historical request characteristics and adapt by exploiting or exploring policies for better performance. The description of the considered problem, its formulation and the details about the framework are described in greater detail in Chapter 4.

1.4.3 Hierarchical Placement Learning for Network Slice Provisioning

In our third contribution, we address the challenge of NSL provisioning in distributed edge environments by developing the Hierarchical nEtwork sLIce prOviSioning (HELIOS) framework, a novel hierarchical learning approach built on the Hierarchical Multi-Armed Bandit (HMAB) model. By partitioning the network into sub-domains based on connected communities, we develop a hierarchical framework of bandit agents that are distributed across the network in each sub-domain to solve the slice provisioning problem. The proposed

framework learns a mapping between slice request requirements and their likelihood of acceptance, along with the expected resource utilization, across different network sub-domains.

We addressed ***RQ 3.1*** by demonstrating how dividing the network into clusters using community detection (Louvain algorithm) and implementing a two-tier agent structure (High-Level Agent (HLA) and Low-Level Agent (LLA)) provides an effective slice provisioning solution. Our experimental results demonstrated that the hierarchical approach outperforms centralized baselines across multiple network topologies with heterogeneous resources by admitting a higher number of requests and having marginally higher utilization than the baselines.

RQ 3.2 is addressed by proposing Hierarchical nEtwork sLIce prOviSioning (HELIOS), a novel two-level hierarchical learning system to solve the online Network Slice Provisioning (NSP) problem by jointly placing the VNFs of a SFC in the network. At HELIOS' high level, a contextual bandit agent directs each slice request to a specific region in the network, depending on the measured resource state and slice features. At HELIOS' low level, a combinatorial bandit agent determines the nodes on which the VNFs will be placed. HELIOS is designed to learn a placement policy that scales with network size.

To address ***RQ 3.3***, we leverage the Generalized Gini Index (GGI) aggregation function which scalarizes and balances multiple provisioning objectives, together. By maximizing this function, we aim to find a point on the Pareto front of the multi-objective optimization problem. This allows for a direct optimization of the different provisioning objectives, enabling trade-offs based on the chosen weights.

By designing a hierarchical decision framework which enables independent and distributed bandit agents to sequentially solve the NSP problem at different network locations, we achieve scalable slice provisioning in distributed, heterogeneous edge environments. HELIOS achieves higher acceptance rates and lower resource utilization compared to both myopic and intelligent benchmark solutions. The hierarchical strategy learned by HELIOS effectively manages the complexity of distributed edge networks, providing a scalable approach to slice provisioning. Our findings contribute to the design of hierarchical learning-based frameworks that combine bandit learning efficiency with structural network awareness, enabling multi-objective optimization in resource-constrained edge environments without needing complete prior knowledge of request patterns or infrastructure capabilities. We describe the framework and its hierarchical learning capabilities in more detail in Chapter 5.

1.4.4 Proactive Resource Optimization for Heterogeneous Network Slices

In our fourth and final contribution, we address the challenge of learning a policy to proactively allocate resources to heterogeneous NSLs, with the goal of meeting their varying SLAs. By predicting the upcoming traffic of different NSLs, we design a framework, PROPHET, that leverages Proximal Policy Optimization (PPO), a Deep Reinforcement Learning (DRL) algorithm, for on-policy learning. We demonstrate that PROPHET is able to optimize the

performance of NSLs by proactively allocating them adequate resources to meet the SLAs of their services.

We address ***RQ 4.1*** by designing a forecaster to predict the upcoming traffic in heterogeneous NSLs. The forecaster is based on an Long Short-Term Memory (LSTM)-attention hybrid architecture and is able to show relatively good predictive performance on a real-world User Equipment (UE) network traffic time-series dataset.

We address ***RQ 4.2*** by designing a DRL framework that integrates the predicted output of the traffic forecaster into the resource allocation policy search problem. Our results show that by incorporating predicted traffic of NSL into the state space of the DRL agent, we're able to learn a policy for proactively allocating resources based on predicted traffic, and thus, minimize the SLA violations of heterogeneous NSLs. Specifically, we show that the average cumulative reward of incorporating predicted traffic is dependent on the prediction horizon, with longer predictions leading to lower average performance for the resource allocation problem.

By combining network traffic prediction and resource optimization through on-policy learning, our proposed framework demonstrates the ability to predictably optimize network resource efficiency and minimize SLA violations in NGMNs, through proactive resource allocation.

The considered problem, its formulation and the detail about the PROPHET framework are described in greater detail in Chapter 6.

1.5 Thesis Outline

This section provides a brief overview of the thesis structure. The rest of the thesis is structured as follows.

Chapter 2 provides the foundational context by defining key concepts and reviewing contemporary research relevant to this thesis. Through a comprehensive examination of literature on network softwarization, edge computing, network slicing, admission control, and online learning, it establishes the basis for the novel contributions presented in subsequent chapters. It also highlights critical gaps in existing work that the thesis seeks to fill.

Chapter 3 introduces a novel online slice admission control framework that evaluates incoming requests through reservation policies that are based on multi-dimensional resource utilization. It shows that reserving scarce network resources can maximize InP revenue under uncertainty about future slice demand. The chapter further shows that, despite employing conservative reservation policies, the proposed algorithm consistently outperforms benchmark methods across various performance metrics.

Chapter 4 introduces the DARIO framework, which adaptively selects near-optimal online slice admission control policies based on historical request patterns. The chapter re-frames slice admission control policy selection as an online learning problem and presents a method that continually estimates and compares policy performance across non-stationary scenarios. By prioritizing high-value requests, the framework increases InP revenue while maintaining responsiveness to evolving demand.

Chapter 5 presents a Hierarchical Decision Framework, HELIOS, that enables scalable slice provisioning across distributed edge networks through hierarchical bandit learning. It explains how we partition the slice provisioning problem's search space to deploy intelligent agents capable of learning effective deployment policies within specific network sub-domains. The evaluation and analysis of the approach demonstrates how the learned policy achieves higher acceptance rates while maintaining lower average resource utilization compared to benchmark approaches.

Chapter 6 discusses the PROPHET approach, which is a proactive resource optimization solution for heterogeneous NSLs. In this chapter, we formulate the problem of continuously allocating network resources to slices with different traffic and SLA requirements. The efficacy of the proposed solution is evaluated based on a real dataset of UE network traffic and on a custom-built OpenAI Gym environment for NS.

Finally, Chapter 7 summarizes the contributions of the thesis by highlighting the key findings and achievements of each research area studied. More specifically, it describes how the different solutions and frameworks proposed by this thesis bring forward the integration of AI in networking for efficiency in resource allocation, service placement and real-time slice dimension adjustments based on usage patterns and demand forecasts. It also describes various avenues for future research which could further enhance network performance and intelligence through AI-native network slicing.

Chapter 2

Background and Related Works

2.1 Background

2.1.1 Network Softwarization

Next-Generation Mobile Networks represent a remarkable technological leap over the previous network generations (i.e., 4G) by introducing significant hardware and software innovations. More specifically, 5G and beyond networks consolidate a major process of softwarization that stands out due to the adoption of cloud-based systems and technologies such as Network Function Virtualization (NFV), Software Defined Networking (SDN), Network Slicing (NS), and a Service-Based Architecture (SBA) [42] (Figure 2.1). Network softwarization enables the design, implementation, building, management and maintenance of equipment, services, and components of mobile networks through programmable interfaces [43]. In turn, this enables the network to be flexible, adaptive, agile and dynamically reconfigurable in the presence of evolving network conditions. The driving force behind the softwarization of the network is the need to reduce Capital Expenditure (CAPEX) and Operating Expenditure (OPEX) while improving operational and lifecycle management of next-generation mobile networks [44]. However, the transition from the current centralized hardware-centric and purpose-built mobile network architecture with tightly coupled control and data planes to a softwarized, disaggregated mobile networks, brings about several challenges.

The main impetus for network softwarization stems from the increasing demands for network agility, scalability, and cost-effectiveness in the face of explosive growth in data traffic, cloud computing, and diverse application requirements, which are largely powered by online activity such as large-scale analytics, remote conferencing, e-commerce, digital communications, and video streaming that amount to hundreds of billions of dollars in market size [14]. By abstracting the availability and discovery of network resources through Application Programming Interfaces (APIs) and orchestrating them through centralized controllers, softwarized networks enable unprecedented levels of automation, service innovation, and resource optimization. This architectural transformation has profound implications across the telco ecosystem, as it influences everything from operator business models and service delivery

paradigms to infrastructure deployment strategies and standardization efforts within bodies such as the Third Generation Partnership Project (3GPP), the ITU Telecommunication Standardization Sector (ITU-T), European Telecommunications Standards Institute (ETSI), the Internet Engineering Task Force (IETF) and the Open Networking Foundation (ONF). As 5G networks mature and research into 6G accelerates, network softwarization continues to evolve, incorporating emerging technologies like NS, Edge Computing (EC), and Artificial Intelligence (AI)-driven network management.

Software Defined Networking

SDN aims to simplify network management and enable the development of new network services by decoupling the control plane from the data plane, enabling centralized and programmable management that enhances scalability and network operational efficiency [45]. The decoupling of control plane functionality from the data plane introduces a novel layer of abstraction and provides opportunities for flexibility and adaptation [46], [47], where the data plane, or infrastructure plane, is the lowest layer in SDN architecture, consisting of forwarding devices such as physical and virtual switches. Through SDN, it is possible to create and separate the network control plane(s) that traverse hardware devices, communicate with network devices (*i.e.,* switches) through southbound APIs and implement network-wide policies to handle various types of traffic flows at a central entity (*i.e.,* controller). The global view of the network enabled by SDN provides a means to detect changes in the network state (such as link loads and link failures) in order to dynamically react and maintain the high-level policies of the network. Furthermore, the complexity of current wireless and mobile networks is also being acknowledged in the SDN domain by making network management simpler and more scalable. The growing consensus is that future Software-Defined Wireless and Mobile Networks, through the integration with SDN, will follow an AI-native approach [38].

Network Function Virtualization

The ongoing trend of NFV reflects a growing trend of softwarization in mobile networks to facilitate flexible, scalable, and adaptive network management [8]. NFV can be considered as a complementary technology to SDN [48] as it also enables scalable network operations through the flexible allocation of resources. This is due to the decoupling of hardware from software functionality, which enables network functions to be implemented as VNFs on General Purpose Processors (GPPs) that could form part of the resources at the edge. Network Function Virtualization facilitates the transition from hardware-based networking platforms towards more off-the-shelf softwarized networks, which would enable the deployment of services in virtualized environments, such as Virtual Machines (VMs) or containers, and more recently, as serverless functions [49]. The advances in NFV have seen the design and implementation of various Physical Network Functions (PNFs) such as load balancers, firewalls, and even Radio Access Network (RAN) functions, as Virtual Network Functions. More specifically, the virtualization of RAN functions towards providing virtualized Radio Access Networks (vRANs) which could be deployed at central locations, such as data centers, is quite revolutionary,

as it reduces the cost of network deployment, as well as greatly improves the ability of the network to scale to varying loads. This results in networks that enable Service Providers (SPs) to instantiate logically isolated entities, also know as Virtual Networks (VNs), on top of a Substrate Network (SN) [50].

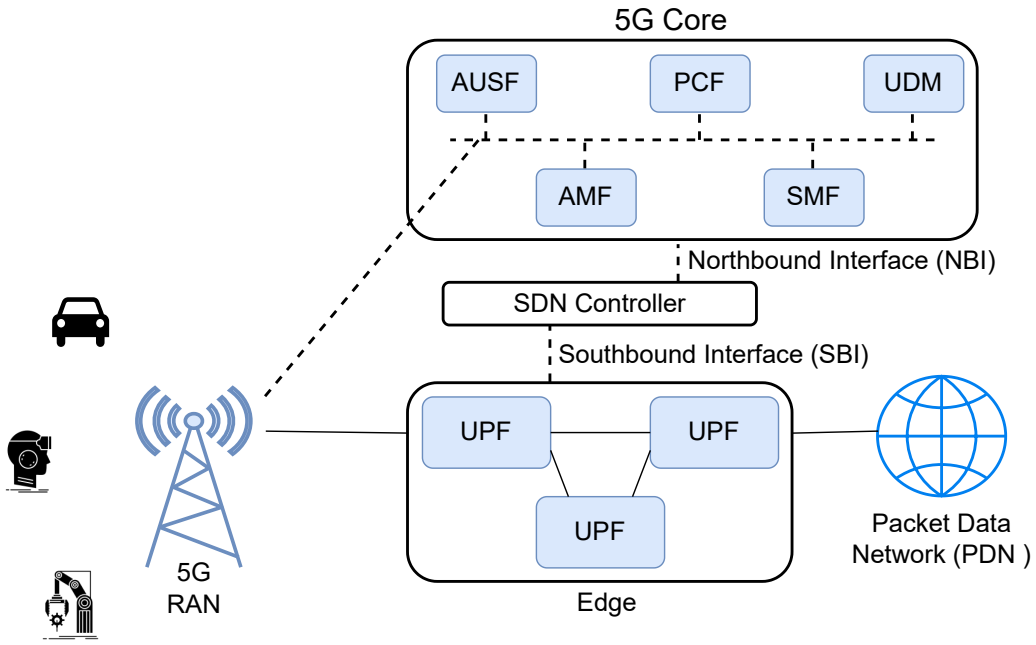


FIGURE 2.1: Service-Based Architecture in 5G Networks.

In order to facilitate the development of flexible wireless networks that can respond quicker to changes in demand, there are efforts being made towards designing service-based networks that can support network virtualization, softwarization, NS, as well as providing an infrastructure for the development of stateful/stateless services, in a novel paradigm known as Network-as-a-Service (NaaS). To achieve this, network elements such as the Evolved Packet Core (EPC) and RAN are remodeled as VNFs that run on GPPs such those found in cloud datacenter environments. Following this trend, Radio Access as a Service (RANaaS) has emerged as a possible new cloud computing paradigm in which an access network is delivered as a pay-as-you-go service that can be instantiated on top of a cloud infrastructure [51]. This allows network operators to respond quickly to changes in network load (e.g., increased demand at a particular cell) by instantiating new instances of these Network Functions (NFs) and distributing the load to them through load balancing. Due to virtualization of the RAN functions such as Radio Resource Management (RRM), radio bearer control, scheduling and transmission, and Core Network (CN) functions such the User Plane Function (UPF), Access and Mobility Management Function (AMF) and Session Management Function (SMF), it is possible reduce resource consumption and meet the demands for heavier cell loads.

As a result of ongoing virtualization efforts in NGMNs, Open Radio Access Networks (O-RANs) have emerged as a critical cloud-based and disaggregated network architecture that revolutionizes conventional cellular networks. By disaggregating Base Stations (BSs) into

virtualized components that communicate through open and standardized interfaces, O-RAN replaces the traditional black-box and monolithic RAN architectures of previous generations (e.g., 4G). This flexible, multi-vendor approach not only enables efficient resource utilization, data-driven optimization, closed-loop control, and automation but also supports flexible network slice management to meet diverse SLA requirements such as latency, throughput, and reliability [52]–[54]. To further enhance flexibility, the O-RAN architecture allows fine-grained control enabled by advanced AI and ML techniques for tasks such as real-time online resource orchestration [55]. However, despite these benefits, the widespread adoption of O-RAN still faces significant challenges due to the complexity of configuration choices, which can profoundly affect both network performance and energy efficiency [56].

In essence, NFV and SDN have stood out as complementary technologies that can be used to create new opportunities that will meet the ever-growing computational and networking requirements for 5G and beyond mobile networks.

2.1.2 Edge and Fog Computing

Edge and Fog computing offer storage, computational, and networking capabilities within a single or multiple domains to enable the deployment of novel applications and services [57]. As a result, they provide an environment in which applications and services can make use of the cloud computing and IT capabilities of devices that are closer to the network edge, thereby reducing the physical and logical distance between application or service path endpoints. As the processing and storage resources (*i.e.,* servers) are closer to the edge, users that require low-latency and high bandwidth capabilities can be supported by the servers that are in close proximity to them compared to cloud data centers. Applications that typically benefit from such capabilities of the network are those that required computational offloading or collaboration (*i.e.,* Vehicle-to-Everything (V2X)) and video content delivery [48] (*i.e.,* Virtual Reality (VR) and Augmented Reality (AR)), where long delays caused by the transmission distance could affect the QoS and user-experienced QoE.

Recent proposals[1] have suggested and considered the co-deployment of edge servers at the radio access network edge in order to drive MEC adoption in 5G systems, and improve the performance of edge-hosted services by enabling access to real-time radio and network information. This would enable context-awareness for mobile services & edge applications to adequately respond to varying channel conditions, for example, by reducing video quality transmitted when the channel is congested.

2.1.3 Network Slicing

The introduction of Network Slicing into the 5G system architecture equips such networks with the ability to meet the heterogeneous requirements of different industry verticals, including Robotics, Factory Automation, VR, and V2X communication [58]. To differentiate between the requirements of the diverse range of applications and services that will need to be supported

[1] https://www.etsi.org/images/files/etsiwhitepapers/etsi_wp23_mec_and_cran_ed1_final.pdf

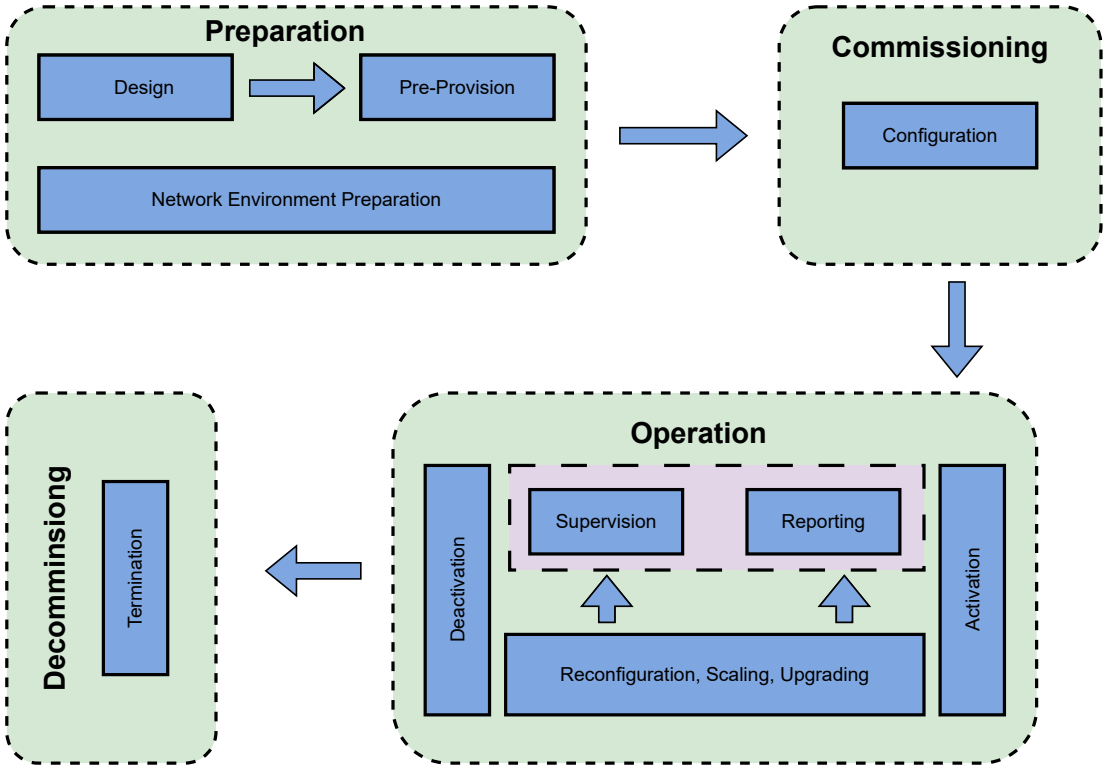


FIGURE 2.2: The lifecycle of a Network Slice Instance

in 5G systems, three major service categories have been suggested which would each require the deployment of Network Slices (NSLs) [59]:

- enhanced Mobile Broadband (eMBB): in this service category, the aim is to have a NSL that can cater to high-throughput (i.e., >10 Gbps) or high-bandwidth services such as video streaming services or immersive multimedia content and provide improved spectral efficiency
- massive Machine Type Communication (mMTC): this service category provides NSLs for that act as a medium for machine-to-machine type communication for devices that transmit smaller amounts of data and can tolerate higher delays
- ultra-Reliable and Low-Latency Communication (uRLLC): with this service category, the NSL is expected to enable mission-critical communications that require very low latencies (1 ms-10 ms)

By efficiently allocating network and compute resources to each Network Slice in the network, the diverse QoS requirements of the services required by each Network Slice can be ensured in order to provide service users with seamless, and high-quality internet experiences [60].

Each NSL can invoke VNFs that run on the common infrastructure, and tailor them to meet its specific application and service requirements in order to ensure NSLs are customized to support specific mobile services, which provides greater flexibility than RAN sharing approaches in previous generation mobiles networks such as 5G [61].

Network Slicing enables cost-effective, multi-tenant communications over a shared physical infrastructure in wireless networks. This is important to support the heterogeneous and diverse services that will need to be supported in 5G and beyond wireless networks. In order to enable personalized services over the same infrastructure, a combination of NFV and SDN approaches is crucial as they could be used to support a tight coordination for VNF allocation and service provisioning at the edge-cloud continuum, allowing for true and flexible service control [4]. Network Slicing can provide an efficient allocation of resources (i.e. Radio, Compute, Network) to the different tenants of the network based on the requirements of the services the tenant serves. Allocating resources to network slices is also vital to providing load balancing, enhancing resource utilization, and improving network performance, which are all important factors in providing a viable network infrastructure for latency-sensitive services.

Introducing the slicing concept into wireless networks requires thorough study. Novel slicing architectures which can provide integrated 5G communication stacks to create QoS-tailored slices, as in [62], will need to developed and evaluated to determine their suitability for latency-sensitive applications and services. A slicing-enabled 5G architecture that is able to efficiently capture the need for integrated network programmability and control, support service orchestration, as well as provision for important concepts such as Central RAN and MEC is the aim of many researchers.

To efficiently manage an instance of a NSL, the 3rd Generation Partnership Project (3GPP) has defined a four-phase lifecycle for network slices [63], [64], namely: 1.) Preparation, 2.) Commissioning, 3.) Operation, and 4.) Decommissioning, as illustrated in Figure 2.2. The preparation phase occurs before creating a network slice instance in the network and involves pre-planning, designing the slice topology, configuring slice parameters, evaluating performance, and negotiation of service attributes, among other tasks. During the commissioning phase, NSLs with varying resource requirements are created and placed onto the underlying network infrastructure and allocated or configured with the resources to meet the requirements of the NSL. In the operation phase, run-time operations are enabled which include activation, supervision, reporting, modification, and deactivation of the deployed NSI. This is achieved through continuous monitoring to ensure the optimal performance of the NSL and adapt to evolving demands. Finally, the decommissioning phase occurs when the NSL reaches the end of its lifecycle or is no longer needed by the ST. In this case, the allocated resources are released, making them available for other future NSLs or VNF.

While each phase has unique challenges that require carefully designed solutions, in this thesis, we primarily focus on designing and developing novel algorithmic and learning-based frameworks that can be used for the management of slice-enabled mobile networks by leveraging online and data-driven optimization approaches to dynamically allocate, orchestrate and optimize network resources in Next-Generation Mobile Networks.

Currently, most NS solution approaches in the literature consider the challenge of resource allocation at the RAN or CN to guarantee slice isolation whilst meeting QoS-requirements of various services. However, in increasingly virtualized mobile networks, the resources in other

network domains, such as the network edge, will form part of the network description and slice components, and thus will need to be allocated and orchestrated, in an optimal manner, to support the co-location of heterogeneous services over a common infrastructure. This is underlined by the introduction of Mobile Edge Computing (MEC) into the network architecture as a way to bring computational power closer to the edge through small but powerful cloud infrastructures [65], in order to reduce the delay for latency-sensitive services by lowering the transmission and processing time between where the data is generated and where it is processed. MEC enables support for a new range of 5G-native mission-critical and real-time applications which could require strict communication and computation guarantees. It also provides the network with an infrastructure to exploit context-based information available at the edge (*i.e.,* for a given geographical area) [66] in real-time, to dynamically alter the performance of edge-hosted applications and slice-specific Virtual Network Functions (VNFs) to meet QoS requirements. While the introduction of MEC will undoubtedly improve service delivery, the co-deployment of edge-hosted network functions and applications with RAN components over the same infrastructure could lead to higher contention of the underlying resources, ultimately leading to performance degradation. This underline the requirement for strict reservation and isolation of network resources to avoid QoS disruptions during the provisioning of applications and services in NSLs at the edge.

Network Slicing typically involves complex decision-making processes that span from admission control and resource allocation to dynamic resource reconfiguration across multiple administrative domains [61], [67]. One of such domains is the network edge, which contains constrained computing resources to support low-latency applications and services. Practical EC networks are typically characterized by their large-scale deployments, temporal and spatial variations in workload distribution, diverse application types, and varied QoS requirements [68]. These factors pose significant challenges for traditional EC systems, including inefficient resource utilization, unacceptable service delays, and limited scalability, requiring solutions that can efficiently allocate heterogeneous resources to services that are included in a Network Slice Description (NSD).

The multi-dimensional nature of network resources and the potential inter-dependencies between different network layers (*i.e.,* cloud, fog, edge) also necessitate a sophisticated learning approach that can capture these relationships effectively and account for the uncertainties that exist at each network layer, while trying to achieve multiple provisioning objectives such as maximizing the number of accepted slice requests deployed at the edge and optimizing resource efficiency.

2.1.4 Online Algorithms for Network Slicing

A key challenge in modern mobile networks is the allocation of multiple resources in increasingly complex and dynamic environments. This challenge is exacerbated by the fact that in edge-based mobile networks, decisions on how much resources to allocate to applications, services or NSLs must be made in real-time, typically without the prior knowledge of future requirements or demands of existing or new requests. Such unpredictability makes

it difficult to optimize different networking objectives such as: maximizing revenue or profit, optimizing resource efficiency and maximizing QoS.

To address this challenge, Online Algorithms (OAs) are typically designed in order to make real-time decisions based only on the information that is currently available. By definition, an OA can be considered as any algorithm that sequentially processes an input sequence without having knowledge of the whole sequence at the beginning of the run. These algorithms have found applications in various fields including: cloud computing [69]–[71], traffic routing [72], appointment booking [73] and electric vehicle charging [74], [75]. The performance of OAs is typically evaluated through the paradigm of competitive analysis [76], where the objective of an OA is to minimize the Competitive Ratio (CR), which is the worst-case ratio of the performance obtained by the offline algorithm in hindsight and the performance of an online algorithm [77]. More formally, given an arrival instance $\mathcal{I}$, we denote by $\mathsf{OPT}(\mathcal{I})$ the optimal performance (*i.e.,* utility or value) achieved in the offline setting when the information of the arrival instance $\mathcal{I}$ is known prior. Similarly, let $\mathsf{ALG}(\mathcal{I})$ denote the performance achieved by an online algorithm ALG. The competitive ratio of the online algorithm can then be represented as $\mathrm{CR} = \max_{\mathcal{I}} \frac{\mathsf{ALG}(\mathcal{I})}{\mathsf{OPT}(\mathcal{I})}$, where $\mathrm{CR} \geq 1$ and the closer the competitive ratio is to 1, the better the performance of the OA compared to the optimal offline algorithm [78].

Network slicing in modern mobile networks presents unique resource allocation challenges that require efficient online optimization that can be addressed through algorithmic approaches such as stochastic network optimization and the domain of competitive online algorithms [79]. Effective OAs for network slicing must address multi-dimensional resource constraints spanning multiple domains including the radio access network, edge network, transport network and core network components, while minimizing computational complexity to enable real-time decision-making at the network edge. While the inherent design of online algorithms typically leads to overly conservative practical solutions that are optimized for worst-case scenarios, they provide strong theoretical results that are provable through performance guarantees, robust to adversarial inputs and interpretable or explainable through transparent decision-making logic, which are important characteristics for their potential deployment in practical network slicing scenarios. To overcome the traditionally conservative nature of OAs, recent advances have considered leveraging historical data to predict future inputs towards the development of ML-enhanced Online Algorithms which infuse predictions in their decision making process, *i.e.,* learning-augmented algorithms [80]–[84].

In this thesis, we addresses the SAC problem in mobile networks by proposing a novel online algorithm. Our approach leverages online reservation policies to facilitate dynamic decisions on slice admission and resource commitment, based on real-time network state. Building upon this foundation, we introduce a data-driven online policy selection framework to augment the algorithm's performance. This extension is specifically designed to improve its average-case performance and resilience when subjected to potentially adversarial input sequences (*i.e.,* requests that are crafted to exploit weaknesses in static reservation policies or overwhelm the system under non-stationary or bursty traffic conditions).

2.1.5 Machine Learning for Network Management

Machine Learning is a branch of AI that is dedicated to developing systems that learn directly from data. This learning process typically consists of two phases, the training phase and the testing phase. During the training phase, typical ML algorithms analyze a dataset to create a model that represents the inherent patterns in the dataset. Subsequently, in the testing phase, the model is applied to new, previously unseen, data to generate predictions based on the learned model. ML algorithms can typically be classified into four primary types: Supervised Learning, Unsupervised Learning, Semi-Supervised Learning, and Reinforcement Learning.

While traditional ML techniques such as Supervised Learning, Semi-Supervised Learning and Unsupervised Learning have found several applications to network management over the years through fields like intrusion detection [85], [86], anomaly detection [87], [88], traffic classification [89], [90], network performance prediction [91], and fault diagnosis [92], [93], there are still several challenges related to their utilization for real-time data processing, their scalability, and their adaptability to dynamic network environments. In contrast, however, RL and OL, are promising and increasingly important techniques that deal with highly dynamic data that becomes available in a sequential order [94], [95]. Specifically, unlike deep learning approaches which often require preprocessing and long training times, online learning does not require pre-processing or offline operations as it adapts at runtime to the system and environment conditions using real-time observations [96]. Similarly, reinforcement learning has gained increasing attention and adoption for network management problem due to its ability to make autonomous decisions in dynamic and uncertain network environments, provide adaptive optimization of network resource allocation, traffic engineering, and service orchestration.

Reinforcement learning explores how an agent can discover which actions to take in an environment to maximize the total reward it accumulates over time. In this approach, the agent learns to interact with the environment by trying out actions, receiving rewards or penalties as feedback, and gradually developing a policy that associates each state with the best action, aiming to achieve the highest possible long-term reward To achieve this, RL agents are faced with the dilemma of choosing between trying out new actions to discover better rewards (exploration) and sticking with known good actions that maximized rewards in the past (exploitation) [97]. Typically, RL methods fall within one of two categories:

- Model-based Methods: these algorithms typically attempt to learn the environment's dynamics (transition probabilities and reward function)
- Model-free Methods: here the algorithms learn the policy or value function directly without explicitly learning the model

Interestingly, RL has seen tremendous success in various domains, including game playing (AlphaGo), robotics, autonomous driving, and network resource management.

OL techniques, such MABs, have emerged across numerous fields where decision-makers must balance exploration (gathering information) with exploitation (using known information) and are under the umbrella of RL methods. While they were originally developed for "ethical"

medical trials [98], this mathematical framework has helped researchers collect valuable scientific data while minimizing potential harm to patients. In the digital domain, web applications represent perhaps the most widespread modern use of MAB algorithms. Website designers employ them to optimize user interfaces and experiences, content curators use them to determine which material deserves prominence, search engines leverage them to refine results, and advertisers utilize them to place ads most effectively on webpages. Similarly, recommender systems incorporate exploration strategies to continuously improve suggestions for entertainment options like movies, dining experiences at restaurants, accommodation at hotels, and many other consumer choices [99], [100]. The domain of economics also offers another ideal ground for MAB applications [101], [102], as sellers can dynamically adjust prices and product offerings, while frequent buyers, such as procurement agencies, can optimize their bidding strategies. Auctioneers can refine their auction mechanisms over time, and crowdsourcing platforms can enhance their assignment of tasks, workers, and compensation rates to maximize efficiency and satisfaction. In robotics, MAB algorithms have been used to help machines iteratively improve their performance across various tasks through systematic trial and error [103]–[105]. Finally, in the networking domain, it is increasingly recognized that experimentation and adaptive learning often outperform rigid network designs and solutions [106], which is the strength of MAB algorithms [107]. This approach enables the optimization of network and datacenter operations, as well as networking protocols, through continuous refinement and learning.

In this thesis, we primarily focus on online learning [108] and deep reinforcement learning-based [97] approaches, in which a control policy or action model can be learned by taking different actions in a *trial-and-error* manner. Specifically, we propose solutions that leverage the lightweight MAB framework, which has seen applications in multiple domains [109] and is a classical, canonical formalization of the exploration-exploitation dilemma [110]. We also propose a solution based on DRL, which we use to address the challenge of allocating network resources to heterogeneous network slices, to optimize the resource efficiency in such networks while minimizing the SLA violations that could arise as a result of dynamic network conditions.

2.2 Related Works

2.2.1 Admission Control in Mobile Networks

The Admission Control (AC) problem in mobile network settings is an essential and fundamental problem and has been well-studied in the past. However, previous works focused on devising mechanisms for the admission of mobile users to the network, as part of the RRM process. The emerging virtualization of the network infrastructure, and the softwarization of network functionality to support multi-tenancy through network slicing, increases the complexity of the AC. Specifically, the increasing virtualization of 5G mobile networks has emphasized the SAC problem. As such, there is a need to explore new admission

control mechanisms and approaches for such networks. A survey on State-of-The-Art (SoTA) strategies and solutions for SAC in 5G networks is presented in [11].

Usually, the goal of NS is to optimize the utilization of network resources to enhance overall network utility which includes metrics like throughput, latency, and revenue, while ensuring that slice-specific SLAs are met. This ultimately translates most problems in the NS domain, such as SAC, into network utility maximization problems [72], [111], and there have been a plethora of works in the literature that focus their efforts on applying advanced techniques such as ML and RL to address such problems. To determine an optimal SAC policy, Bakri et al. [112] compare the performance of both online and offline solutions. Gholamipour et al. formulate a joint online admission control and resource allocation problem based on an Integer Linear Program (ILP) [113]. Their solution considers the energy consumption of network nodes, as well as the workload uncertainties of sliced VNFs in their approach. Sciancalepore et al. enable concurrent slice requests to be deployed over an InP's physical resources while maximizing multiplexing gains [114]. Their approach focuses on addressing the SAC problem under demand uncertainty of network slices. Salvat et al. focus on the slice orchestration problem by jointly considering admission control and resource reservation [115]. They propose solutions based on an optimal Benders decomposition method and a sub-optimal heuristic, to address the considered problem. Finally, Noroozi et al. study the SAC problem, where each slice is made of RAN and CN resources, and propose a sub-optimal two-step heuristic algorithm to maximize the total revenue gained from admitted slices, in [116].

Chen et al. [117] propose an optimal admission control mechanism for delay-sensitive services in edge environments. In their work, they look to balance maximizing provider revenue and ensuring the QoS of existing services are guaranteed by balking incoming requests. The work of Sciancalepore et al. enables concurrent slice requests to be deployed over an InP's physical infrastructure while maximizing the utilization of network resources through stochastic multiplexing and maximizing the overall InP profit of accepted NSRs [114]. Luu et al. [118] jointly study the problem of Admission Control (AC) and resource reservation in mobile networks to guarantee the SLAs of slices under uncertainty in the resource demand of slice requests. J. Leguay et al. study the AC problem under the framework of a centralized SDN controller and evaluate the performance of key AC algorithms under realistic settings [119]. Vincenzi et al. explore the application of ML for SAC [120]. They train a Neural Network (NN) to learn the best AC policies based on pre-computed conditions and use these policies to make near-optimal decisions at runtime. Lindståhl et al. consider the problem of SAC and propose a MAB-based approach to jointly decide on the admission of NSRs, as well as the number of resource measurements to take before committing to the admission decision [121]. Prasad et al. propose reservation-based mechanisms that enable quick decisions on the admission of NSRs [122]. Through simulations, they show that their approach can quickly admit slice requests, while also optimizing for revenue achieved by the InP. Sulaiman et al. propose the use of multi-agent DRL to jointly solve the problems of network slicing and SAC [123], [124]. Dayot et al. propose a Deep Contextual Bandit that combines DRL with a Contextual Bandit (CB) model for the problem of resource allocation [125]. Their results show that their approach

TABLE 2.1: Related Works: Policy Selection for SAC

Reference	Admission Control	Online Learning	Drift Detection	Policy Selection
Scianalepore et al. [114]	✓	✓	✗	✗
Chen et al. [117]	✓	✗	✗	✗
Luu et al. [118]	✓	✗	✗	✗
Vincenzi et al. [120]	✓	✗	✗	✗
Lindståhl et al. [121]	✓	✓	✗	✗
Prasad et al. [122]	✓	✗	✗	✗
Sulaiman et al. [123]	✓	✗	✗	✗
Sulaiman et al. [124]	✓	✗	✗	✗
Dayot et al. [125]	✗	✓	✗	✗
Jiang et al. [126]	✓	✗	✗	✗
Haque et al. [127]	✓	✗	✗	✗
Bakhshi et al. [128]	✓	✗	✗	✗
DARIO [15]	✓	✓	✓	✓

optimizes network resource efficiency and the achieved slice throughput compared to other intelligent solutions.

In essence, various RL and DRL-based solutions for the SAC problem in 5G networks have been proposed [123], [126]–[128]. However, such solutions typically suffer from problems such as the curse of dimensionality [129] and the cold-start problem [130] in which a large number of training samples are required in order to learn a *single* policy and are therefore likely to have limitations in terms of their practical performance. A summary of the related works can be found in Table 2.1.

2.2.2 Online Learning in Mobile Networks

NGMNs are expected to provide native, embedded intelligence to support heterogeneous, killer applications and human-centric services, ultra-fast handover under mobility scenarios, reconfiguration of network components, and smart energy consumption. This will require the development of learning-based frameworks which can detect changes in network conditions and scenarios and rapidly adapt to them. Such framework will need to go beyond pre-trained AI models as they might not be able to adequately respond to ongoing changes due to their largely static nature. More specifically, such learning-based frameworks will need to depend on ultra-fast, OL or Continual Learning (CL) models to provide AI-enabled network optimization compared to traditional learning and optimization techniques [131], [132]. This is because in contrast to traditional learning and optimization approaches, OL/CL approaches offer the advantage of adaptability to non-stationary system dynamics or distributions, as well as potentially unforeseen scenarios [6], [133]. In such systems, when the data distribution shifts, the learning model needs to adapt, which is a central concern for online and continual learning. The majority of existing solutions detect these shifts via statistical tests, changes in the empirical loss, or they utilize fixed time-windows to identify and discard outdated data, and then retrain the model [134]. Hence, the development and deployment of OL/CL-based

frameworks would enable NGMNs to automatically learn efficient resource management policies in complex scenarios characterized by time-varying statistics [135]. A comprehensive review of OL algorithms and techniques can be found in [136].

Despite the need for learning-based frameworks that can operate and adapt to real network environments, there are currently limited works that apply OL in the context of resource allocation, resource management or NS in the literature. This opens up an avenue for the development of OL solutions that can address such problems in time-varying networks. Under the banner of OL algorithms, BO and MABs [109] stand-out as well-known optimization and sequential decision making frameworks, where at every time step, a decision needs to be made between different actions that have an unknown probability of occurring [137]. Contextual Multi-Armed Bandit (C-MAB), extend this framework by observing N-dimensional context-vectors before choosing an an arm (or action), with the goal of choosing actions that maximize the reward or minimize the regret (*i.e.,* they consider context-dependent reward functions). As C-MABs are able to consider the trade-off of the cost of acquiring new information (*exploration*), against the generation of rewards based on existing information (*exploitation*) [138], they are useful in dynamic environments that necessitate sequential decision making. More specifically, as part of the OL paradigm, MABs provide an efficient solution to optimizing actions by learning through sequential feedback [139].

S. Boldrini et al. propose the use of MABs for wireless network selection by a multi-Radio Access Technology device, with the goal of maximizing the quality perceived by the device user [140]. Kerkouche et al. in propose light-weight learning methods, based on MABs, to select the communication parameters (namely spreading factor and emission power) in low power wide area networks [141]. They are able to show that such an approach can manage the trade-off between energy consumption and packet loss much better than other adaptive algorithms. An online path manager for Multi-Path TCP that is based on a contextual bandit algorithm is used to choose the optimal primary path connection that maximizes throughput and minimizes delay and packet loss in heterogeneous networks in [142]. Kalntis et al. consider the problem of handover optimization through the prism of OL and propose an algorithm that provides robust and dynamic regret guarantees even in challenging environments [143]. Bistritz et al. propose an OL algorithm for adaptive admission control of networked open multi-agent systems [144]. In their proposal, a centralized admission controller uses OL to adjust the probability of admitting new players in a distributed system in order to achieve system equilibrium.

Due to the increasing virtualization of the network, OL based solutions have been gaining increasing attention in the literature for a variety of resource management problems in mobile networks. Monteil et al. develop a learning-based framework based on the theory of Online Convex Optimization in order to study how the service providers should reserve resources to maximize their services' performance while not violating a time-average budget threshold [145]. They analyze the impact of key system parameters on the learning performance, and discuss the implications for the design of Network Virtualization markets. Kalntis et al. propose an OL-based scheme that dynamically chooses the best-performing resource allocation

algorithm and operates under minimal assumptions and without requiring knowledge of the environment [146]. They show that their proposed solution achieves sub-linear regret (zero optimality gap), and characterize their dependence on the main system parameters. Lahmer et al. propose a Continual Learning strategy as part of a novel resource allocation framework that enables an agent to adapts to sudden changes in traffic dynamics [135]. They test their strategy with a network slicing use case in which the learning agent and system users compete for the same network resources and show that their approach is able to outperform static baselines. Liu et al. propose an OL-based solution for End-to-End (E2E) network slicing [147] and service configuration [148], which allows individualized learning for each NSL and maintains its SLA through a novel constraint-aware policy update method and proactive baseline switching mechanism, and a Bayesian Optimization method. Their results show that they're able to reduce utilization by up to 61%, while maintaining zero SLA violations throughout the OL phase. Zhao et al. design a novel adaptive network slicing system, based on OL, to continuously orchestrate virtual network resources to adapt to changing network dynamics [55]. The proposed system integrates an AI/ML technique based on Bayesian Optimization with an optimization method based on Alternating Direction Method of Multipliers, to improve the virtual resource utilization of slices. Their results show that they're able to outperform State-of-The-Art solutions by over 60% for their considered metrics.

In the context of slice admission control, current solutions focus on designing fixed or static (i.e., *one-size-fits-all*) models and policies. However, as mobile networks are inherently dynamic, especially at the edge, such policies are likely to face performance shortcomings at runtime, since they rely on strong assumptions, such as the expected traffic demand [129]. The development of a framework that learns the most optimal admission policy or model by dynamically selecting and learning the performance of different admission control algorithms, is missing and is therefore a relevant gap in the literature that this thesis aims to address. Such a problem can be addressed by modeling the problem as a MAB problem, a portfolio selection problem or Policy Selection (PS) problem, of which there are many potential solutions that can be applied [149]–[154].

2.2.3 Hierarchical Learning in Mobile Networks

An important challenge in modern mobile networks is the efficient allocation of resources under uncertainty, particularly in large-scale environments such as the network edge, where the complexity and interdependence of components significantly complicate management [155]. Traditional single-layer optimization methods often fail to capture the sequential and interrelated nature of decision-making across multiple network functions, leading to suboptimal performance. The dynamic and resource-constrained nature of these networks requires adaptive strategies capable of generalizing across varied scenarios.

A Hierarchical Decision Framework (HDF) offers a structured approach to this challenge by decomposing complex tasks into multiple decision layers. In such frameworks, high-level policies define strategic objectives that guide lower-level policies responsible for more granular control,

thereby improving both scalability and coordination. The most common instantiation is based on Hierarchical Learning (HL) or Hierarchical Reinforcement Learning (HRL), where policies are learned at different abstraction levels to break complex problems into simpler sub-tasks handled by specialized agents.

Compared to centralized learning, where a single agent operates over a global state representation, and fully decentralized learning, where each agent must estimate the policies and actions of all others, HL provides a balanced middle ground. It scales more effectively than centralized approaches while avoiding the high coordination complexity of fully decentralized systems [156]. In hierarchical architectures, coordination requirements are largely confined to well-defined interfaces between layers, enabling efficient information exchange without full network-wide synchronization.

This trade-off is illustrated in Figure 2.3, which presents the spectrum of learning-based architectures applicable to dynamic resource allocation under uncertainty. The figure also highlights the specific coordination assumptions required by each paradigm: from the strong global observability and synchronous updating needed in centralized systems, to the minimal but more complex agent reasoning in fully decentralized systems, to the structured and more scalable intermediate approach enabled by HL or HRL.

In the literature, HRL has been used to learn hierarchical provisioning policies for NS. Wei et al. propose an intelligent hierarchical NS framework that operates at two different time-scales [157]. Their results demonstrate that their lightweight framework outperforms benchmark algorithms in terms of system utility, throughput and transmission delay. Sun et al. [158] propose a hierarchical radio resource allocation architecture for allocating sub-channels to NSLs and NSLs resources to UEs. They show that they are able to outperform the benchmark algorithms that they compare their solution to. A hierarchical meta-RL solution is proposed by Chen et al. [159] who seek to optimize the provisioning of services and manage the allocation of radio resources in O-RANs. Through numerical analysis, they show that their propose HRL approach outperforms other SoTA benchmarks. Similarly, Qiao et al. [160] propose a HL multi-cell, multi-dimensional resource, and multi-timescale O-RAN slicing framework to meet the QoS of heterogeneous services and lower the costs for SPs. Their solution is based on a DRL approach in which, the upper layer of the framework make decisions regarding the size and number of Physical Resource Blocks (PRBs) and the amount of computing resources to allocate to each NSL, while the lower layer of the framework is tasked with making PRB selection and computing resource allocation decisions. A general summary of the related works are given in Table 2.2.

HRL in mobile networks has emerged as a crucial paradigm for addressing the challenges of distributed intelligence, resource constraints, and dynamic environments inherent in modern mobile systems. Generally, in HL, agents develop high-level policies for goal-oriented behavior while simultaneously acquiring lower-level policies for managing specific sub-tasks [161]. In the context of network slice provisioning or placement, incorporating hierarchical learning into a hierarchical decision framework can enhance the effectiveness of the learned provisioning policy.

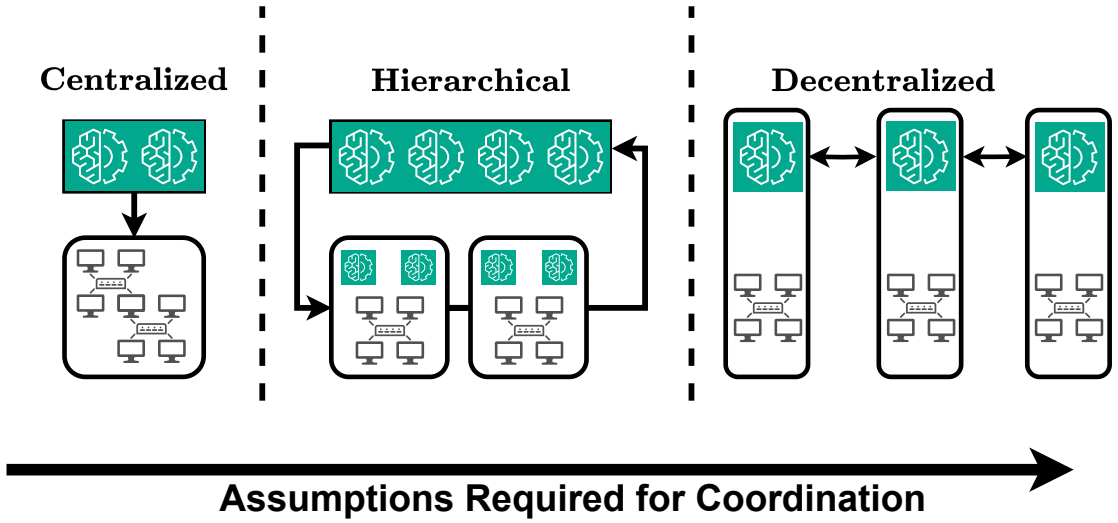


FIGURE 2.3: Architectures for Addressing Dynamic Resource Allocation under Uncertainty (Adapted from [156]).

TABLE 2.2: Related Works: Network Slice Provisioning

RNTs = Real Network Topologies

Reference	NSP	OL	MOO	HDF	RNTs
Lin et al. [32]	✓	✗	✗	✗	✗
Mao et al. [162]	✓	✗	✗	✗	✗
Wu et al. [163]	✓	✓	✗	✗	✗
Hojeij et al. [164]	✓	✗	✗	✗	✗
Yue et al. [165]	✓	✗	✗	✗	✗
Bi et al. [166]	✓	✓	✗	✗	✗
Han et al. [167]	✓	✗	✗	✗	✗
Li et al. [168]	✓	✗	✗	✗	✗
Tran et al. [169]	✓	✗	✗	✗	✗
Lim et al [170]	✗	✓	✓	✓	✗
HELIOS [171]	✓	✓	✓	✓	✓

By leveraging hierarchical structures, complex tasks, such as service function chain placement, can be decomposed into a series of smaller, more tractable sub-tasks, each addressed at a specific level of the hierarchy. This decomposition can yield significant performance gains by enabling more focused decision-making at each stage. However, while hierarchical learning effectively reduces complexity, it may inadvertently oversimplify inter-dependencies between sub-tasks. For example, in network slice provisioning, constraints or correlations across different service function placements might be overlooked and potentially lead to suboptimal global policies.

Inherently, the network slice provisioning problem involves addressing multiple conflicting objectives, such as optimizing resource utilization, minimizing service delays, and ensuring fairness across slices. Multiple-Objective Optimization (MOO) techniques allow network operators to balance trade-offs dynamically, adapting to changing service demands and network conditions. However, there are not many works that design hierarchical solutions that aim to solve multiple objectives jointly and in a fair manner. It is also noted in [172], that

there is currently no consolidated framework that can be used to learn hierarchical policies where different approaches are used for different benefits. More specifically, there's a lack of HDFs that are designed to operated in an online manner and without the need for extensive and expensive upfront training. In this thesis, we look to fill this gap by proposing a HMAB framework for network slice provisioning. Our proposed framework, Hierarchical nEtwork sLIce prOviSioning (HELIOS), integrates contextual information into the decision-making process while ensuring an effective balance between exploration and exploitation. Building on a HMAB formulation, HELIOS addresses a key gap in the NSP literature by structuring the decision process across multiple hierarchy levels. At each level, specialized algorithms are employed to tackle the distinct subproblems inherent to that layer, enabling more efficient and context-aware resource allocation.

2.2.4 Proactive Resource Optimization in Mobile Networks

Leconte et al. propose a flexible and lightweight resource allocation framework, which uses cloud-native architectural concepts, and methods from continuous optimization, to provision and auto-scale slices in real-time [173]. Their approach is based on flexible utility functions that are subject to the network bandwidth and processing power capacity of cloud instances. Chien et al. propose an algorithm that jointly considers communication and compute resources in the allocation of resources for heterogeneous, multi-tenant 5G services [174]. The proposed algorithm adjusts the capacity of services and traffic allocation to minimize the over-provisioning of network resources, while satisfying the latency constraints and other SLA of tenants. To jointly manage radio and computational resources in resource-constrained edge environments, Liu et al. propose a solution that achieves low-latency for delay-sensitive applications [175]. Their solution minimizes service latency by optimizing a range of network parameters, including: uplink (UL) transmission power and the amount of computational resources allocated to users of edge applications. Mahmoud et al present a data-driven resource allocation approach which is tested in an O-RAN environment [176].

In their approach, they compared the effectiveness of different algorithms in minimizing the number of PRBs utilized in the system, by optimizing the Throughput-to-Bandwidth ratio. An intelligent network application (or xApp) for slicing the RAN using AI and Deep Learning techniques is proposed by Yeh et al. [177]. Through their evaluations, the authors show how different services can co-exist over a common network infrastructure, while meeting the SLAs of each service. Wang et al. propose a Multiple-Agent Deep Reinforcement Learning (MADRL) framework for robust Network Slicing in dynamic environments [178]. Their solution looks to address the problem of admitting randomly arriving service requests from users, under time-varying channel and user mobility scenarios. Kak et al. develop an automatic Network Slicing framework to address the objectives of computing optimal network routes and allocating network resources, with minimal SLA violations [179]. Their proposed framework does not utilize prior information concerning the resource requirements associated with a NSL, and is therefore robust to a wide variety of use cases and scenarios. Mohanti et al. propose a resource allocation framework that uses predictions of the network quality [180]. Their proposed solution

is used to demonstrate how look-ahead forecasting of channel conditions can improve the performance of the network over traditional resource allocation methods, however, they don't specifically consider the O-RAN architecture or the challenge of slice reconfiguration. Wei et al. propose a predictor-optimizer framework that intelligently performs inter-slice reconfiguration with the aim of minimizing the energy consumption while provisioning NSL [181]. In their work, they use prediction intervals that are made of lower and upper bounds which constrain the future traffic demands with a pre-specified probability. In the work of Balasingam et al., a novel RAN slicing system that provides assurances of application-level throughput and latency, is proposed [182]. By forecasting the availability of RAN resources, their proposed slicing system is able to determine whether or not the admission of a new slice request will ensure that the SLAs of already admitted requests will be met.

While a plethora of works [183]–[187] consider RL and Value Iteration methods for data-driven optimization and close-loop control of NSLs in O-RAN, such approaches either require long training periods, fail to scale or do not meet system requirements [188], and as a result, they are challenging to apply to real-time scenarios since they must be re-trained each time the environment changes [156]. Moreover, such techniques, which are based on DL architectures, often require extensive *offline* training to be able to accurately forecast future traffic demands [189], which limits their application for real-time reconfiguration in edge or O-RAN environments. Our approach differs from previous works as we build our online predictive optimization framework for NSL reconfiguration based on 2-components. The first component carries out traffic demand forecasting to determine the amount of traffic that would be expected in each slice in an upcoming period (e.g., 2 minutes in advance). The second component, based on the predicted traffic demand obtained in the first step, dynamically allocates resources to the deployed NSLs (i.e., adapts the slices) in order to maximize their time-varying SLAs, collectively.

2.3 Chapter Conclusions

This chapter lays out the foundational groundwork, including background information and related concepts that are essential to the different topics explored within this thesis. In Section 2.1, we give a general overview of the background on modern mobile networks, discussing concepts such as network softwarization, edge and fog computing, network slicing, online algorithms for admission control and ML for network management. The covered concepts are required for understanding the contributions which are subsequently presented in Chapters 3, 4, 5 and 6.

In Section 2.2, we discuss various related works that cover the state-of-the-art techniques and solutions that address the research questions of this thesis, which we described in Section 1.3. We highlight the major drawbacks of the different related works that necessitate new solutions to handle the research questions in this thesis.

In Section 2.2.1, we evaluate the most relevant works related to SAC in mobile networks, considering both the online and offline settings. We find that most works study the SAC

problem in the offline setting and propose solutions that require full knowledge of all NSRs before deciding on the requests to admit, which severely limits their practicality in real networks. Of the works that consider the online setting, most utilize RL or DRL solutions that require a long training period to find learn an admission policy.

In Section 2.2.2, we discuss the relevant literature on the wide and diverse use of OL approaches in mobile networks. We highlight the ability of CL and OL models to provide AI-enabled network optimization compared to traditional learning and optimization techniques as they are better at adapting to changes in network conditions. We further explain how such models detect network changes through statistical tests, changes in the empirical loss, or fixed time-windows in order to adapt. Then, we explain how current solutions to resource management problems, such as SAC, are typically designed as fixed or static models and policies which could have poor performance in real-time, dynamic networks and highlight how a solution for selecting between multiple admission policies could lead to improved network performance.

The problem of scalable network slice provisioning is considered in Section 2.2.3, where we discuss HL or HRL as a typical way to address the issue of dynamic resource allocation in large-scale mobile networks by breaking up the problem into sub-problems that can be addressed by different agents. We present the related works and how they apply such techniques in the context of resource allocation and provisioning NSLs and explain their advantages over centralized and decentralized approaches with respect to their complexity and in terms of their assumptions required for coordination.

Finally, in Section 2.2.4, we give details on how AI/ML techniques such as DL can be used to forecast traffic demands and resource availability in order to proactively allocate resources and improve network management tasks across a range of architectures.

To solve the various issues discussed in the related works and address the previously formulated research questions, we design and evaluate four approaches in this thesis, which improve the provisioning of NSLs, as explained in Chapters 3, 4, 5 and 6. In particular, Chapter 3 addresses the challenge of online SAC, as described in Section 2.2.1. Chapter 4 addresses the issues of learning an admission control policy in an online setting by selecting between different policies, as explained in Section 2.2.2. In Chapter 5 addresses the problems of scalable NSL provisioning in edge-enabled networks, as described in Section 2.2.3. Finally, in Chapter 6, we address the challenge of proactive or predictive resource optimization for heterogeneous traffic in 5G networks, as explained in Section 2.2.4.

Chapter 3

Multi-dimensional Slice Admission Control

3.1 Introduction

A major challenge in the management of slice-enabled 5G networks is the allocation of resources required to support dynamic and online (or real-time) provisioning of network slices. Addressing this problem requires efficient and timely assignment of virtualized network resources to optimize network objectives such as improved resource efficiency, improving Quality of Service (QoS) metrics, or increasing infrastructure provider revenue by maximizing the number of admitted slices in the network [118], [190]. Hence, an important consideration in the dynamic provisioning of network slices is the decision that an Infrastructure Provider (InP) needs to make when a slice request (i.e., Network Slice Request (NSR)) from a tenant is received for the deployment of a new network slice onto the physical network infrastructure [191]. This is commonly described as the Slice Admission Control (SAC) problem, and requires finding an optimal strategy that achieves the InPs's primary objectives, while also considering the spatio-temporal traffic dynamics of network slices [112]. While the SAC problem is challenging to address as it typically involves a level of uncertainty in terms of the unknown arrival rate, multi-dimensional resource requirements, expected revenue, and the lifetime of NSRs, devising an appropriate SAC mechanism is important for achieving improved resource efficiency and fairness in a shared network infrastructure [126].

In this chapter, we investigate the problem of online slice admission control in next-generation mobile networks, where an InP has to determine whether or not to accept arriving NSRs in an online manner without the benefit of knowing the demands or offers of future requests from Slice Tenants (STs).

The contributions of this chapter aim to answer the following research questions described in Section 1.3.1.

RQ 1.1: How can admission control policies effectively manage the **multi-dimensional** resource demands of network slices while maintaining high resource efficiency in dynamic mobile networks?

RQ 1.2: How can online admission control policies ensure long-term **revenue optimization** in network slicing, while accommodating heterogeneous slice requirements and uncertain demand?

RQ 1.3: How does **economic disparity** among slice tenants influence admission control decisions, and how can admission control policies balance revenue optimization with fairness?

To address the above mentioned challenges, we propose the Online Slice Admission Control (OSAC) algorithm. This algorithm leverages online reservation functions to decide on the admission of sequentially arriving slice requests without explicit knowledge about potential benefits of future requests. We address the ***RQ 1.1*** by formulating the SAC problem as an Online Multidimensional Knapsack Problem (OMdKP) where the goal of the problem is to maximize the value of the requests admitted into the network. We consider that the value of each slice request is a weighted combination of the required multi-dimensional resources and the duration of the request in order to accurately model a range of request types and observe their impact on the long-term resource efficiency in the network. We address the ***RQ 1.2*** by developing an algorithm that leverages online reservation functions to reserve network resources based on linear or exponential cost functions. This ensure that scarce network resources are available for high-valued slice requests, regardless of the order the requests arrive in. To address ***RQ 1.3***, we construct different scenarios, where each scenario is based on the assumption of the distribution of the number of high-valued and low-valued requests. This Beta distribution is based on a parameter that controls the unit value of each request and helps us to evaluate the performance of our algorithm under different scenarios for the pre-determined metrics.

Our contributions are as follows.

- We model the OSAC problem as an OMdKP with unknown resource demands and lifetimes and where the objective is to maximize the long-term revenue received from NSRs, while respecting the capacity constraints of the system resources.
- To address the OSAC problem, we leverage two novel policies, Linear Reservation Policy (LinRP) and Exponential Reservation Policy (ExpRP), and propose an online algorithm that utilizes either of the two policies to perform admission control of NSRs in the considered online scenario.
- Through extensive simulations, we evaluate the performance of the online algorithms in terms of the average resource utilization, the acceptance ratio, and total revenue gained compared to an online greedy solution.

The following sections discuss the content in our paper published in the 2023 IEEE 20th Annual Consumer Communications and Networking Conference (CCNC) [15]. Section 3.1 describes the research questions addressed in this chapter and discusses the contributions of the online algorithms proposed to address the considered SAC problem. Section 3.2 describes the system model and formulates the SAC problem. Section 3.3 presents the OSAC algorithm and the online reservation functions leveraged in our solution, in greater detail. Section 3.4 describes

TABLE 3.1: Summary of Notations

Symbol	Description
m	Number of resource types
$\mathcal{H}$	Index set of NSRs
$r_{h,j}$	Requested amount of resource j by NSR h
δ_h	Lifetime of NSR h
τ_h	Timestamp of NSR h
π_h	Revenue of NSR h
p_h	Unit value of NSRs h
α_h	Coefficient vector of NSRs h
C_j	Aggregated capacity of resource j
x_h	Decision variable on NSR h
$q^{\text{lin}}_{h,j}$, $q^{\text{exp}}_{h,j}$	Normalized Resource Utilization of LinRP and ExpRP policies for resource j when NSR h enters the system
ϕ^{lin}_h, ϕ^{exp}_h	System Admission Cost of LinRP and ExpRP policies when NSR h enters the system
κ_j	Resource heterogeneity ratio for resource j
θ	Willingness-To-Pay-Ratio

the simulation setup and discusses the evaluation results. Finally, section 3.5 concludes the study in this chapter by reviewing the main contributions and highlighting the relevant results.

3.2 System Model and Problem Formulation

In this section, we formulate our problem in a typical network slicing scenario, in which an InP owns the network resources and can lease them to STs for variable periods. Thus, we describe the system model of the virtualized mobile network and formulate the SAC problem.

3.2.1 System Model

Infrastructure Model: We consider a scenario that consists of a sliced edge infrastructure made of BSs co-located with Edge Servers (ESs), which are equipped with an arbitrary amount of virtualized network and compute resources (e.g., bandwidth, CPU, RAM, storage). In this scenario, an InP owns and leases the physical network resources by dynamically allocating them to incoming NSRs. We assume that the physical resources of all edge servers in the system are pooled together to support heterogeneous NSs with different resource requirements. Therefore, in this work, the NSRs are admitted and executed on a single virtual infrastructure, physically distributed over a set of edge servers. In the considered scenario, STs seek to deploy and instantiate new slices (with their Virtual Network Functions (VNFs)) on the infrastructure to provide their services, as shown in Figure 3.1.

In line with the time-varying network conditions at the network edge in Network Function Virtualization (NFV)-enabled mobile networks, we assume that time in the system is divided into consecutive intervals called *time slots*. Each time slot is indexed with an element of the ordered index set $T = \{1, \ldots, |T|\}$. We assume that the sliced edge infrastructure offers

m different types of resources, whose amount does not change over time. We define $[m] = \{1, \ldots, m\}$ as an index set that identifies each of the m resource types. For example, in this work, we assume that $m = 3$, as the resource pool primarily consists of computing resources such as CPU, memory, and storage, which can be hosted by edge servers or a regional cloud provider. We define the variable $C_j \in \mathbb{R}_+$, with $j \in [m]$, as the aggregate capacity of the j-th resource in the whole infrastructure. For example, in a scenario with $m = 3$ types of resources, C_1, C_2, and C_3 can indicate the infrastructure-wide capacity of CPU, memory, and storage, respectively.

Slice Request Model: Each of the consecutive tenant-generated NSRs is identified by the incremental index $h \in \mathcal{H}$, where $\mathcal{H} = \{1, \ldots, |\mathcal{H}|\}$ is defined as the ordered set of all NSR indices. Furthermore, each NSR's features are summarized by a tuple $(\boldsymbol{r}_h, \delta_h, \tau_h, \pi_h)$, whose components are described hereafter. The variable $\boldsymbol{r}_h$ is an m-dimensional vector, where each component $r_{h,j}$ is the requested amount of a resource of type j by an NSR h. The variable $\delta_h \in \mathbb{N}$ represents the *lifetime* of NSR h, defined as the number of time slots the NSR needs to access the network resources. The variable $\tau_h \in \mathbb{N}$ represents the *timestamp* of NSR h, defined as the time slot index at which the request arrives to the scheduler. The variable $\pi_h \in \mathbb{R}_+$ represents the *revenue* of NSR h, defined as the economic benefit gained by the InP from accepting the tenant's slice request and granting it its requested resources. We calculate it as $\pi_h = \delta_h p_h \boldsymbol{\alpha}_h \boldsymbol{r}_h$, with $\|\boldsymbol{\alpha}_{h,j}\|_1 = 1$. The j-th component of the coefficient vector $\boldsymbol{\alpha} \in (0,1]^m$ represents the relative weight of the j-th resource to constitute a logical resource unit, and p_h represents the *unit value* of each logical resource in the NSR. The Manhattan-norm $\|\boldsymbol{\alpha}_{h,j}\|_1$ is used to represent the sum of coefficients of each resource j, for an NSR h, and is given by $\sum_{j \in [m]} \alpha_{h,j} = 1$. In this revenue model, we assume that a higher value of a coefficient represents a higher priority of the resource towards meeting the SLA of the NSR.

3.2.2 Problem Formulation

The considered dynamic SAC problem aims at maximizing the long-term revenue collected from admitting NSRs by optimizing the decisions on which NSRs should be admitted to the system while respecting the capacity requirements of the available resources.

Let us define x_h as the *decision variable* that takes a value of 1, if the NSR is admitted, and 0 otherwise. We define $\boldsymbol{x} = (x_1, \ldots, x_{|\mathcal{H}|})$ as the *decision vector*, where $\boldsymbol{x} \in \{0,1\}^{|\mathcal{H}|}$. Let us define the *indicator function* $\mathbb{1}_A : T \to \{0,1\}$ so that $\mathbb{1}_A(t) = 1 \iff t \in A$ and $\mathbb{1}_A(t) = 0 \iff t \notin A$. We can now define the SAC problem as follows.

$$\max_{\boldsymbol{x} \in \{0,1\}^{|\mathcal{H}|}} \quad \sum_{h \in \mathcal{H}} \pi_h x_h \tag{3.1a}$$

$$\text{s.t.} \quad \sum_{h \in \mathcal{H}} r_{h,j} x_h \mathbb{1}_{[\tau_h, \tau_h + \delta_h]}(t) \leq C_j, , \quad \forall j \in [m], \forall t \in T \tag{3.1b}$$

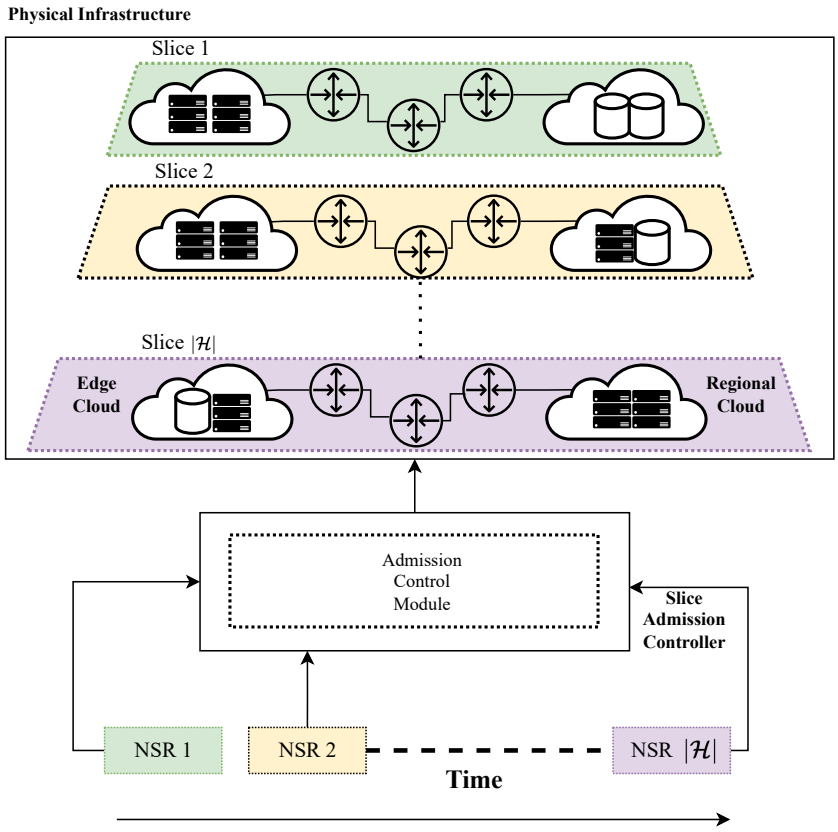


FIGURE 3.1: System Overview

In this problem, constraint 6.3b ensures that at any time slot and for any resource, the system allows only a number of NSRs whose aggregate resources are less than the aggregate system capacity. We observe that the stated SAC problem is NP-hard [192], as it can be reduced to a variant of the Multidimensional Knapsack Problem (MdKP) problem, which is also NP-hard. The literature offers several methods to approximate the solution of an NP-hard problem through algorithms that have polynomial complexity. However, the main limitation of this *offline* formulation of the SAC problem is that, to compute the optimal solution, the scheduler must know the whole sequence of arriving NSRs, including those who will arrive in the future, which is unfeasible in real-world scenarios. The lack of future information when deciding on the admission of NSRs makes the considered SAC intractable [190]. Hence, to enable real-world systems to approximate the solution of the stated offline SAC problem, we design a scheduler that uses an *online* algorithm for Slice Admission Control to optimize revenues.

3.3 Online Slice Admission Control

We consider the online version of the SAC problem, where the unknown inputs are the multi-dimensional resource requirements (or demand) of the NSRs, and their corresponding revenue. In such a problem, the goal is to design an online algorithm that determines how

sequentially arriving requests are accepted, based only on the currently available information. Specifically, we consider the scenario in which arrving NSRs cannot be buffered in the scheduler *queue*, and therefore a decision must be made on it's admission immediately and before the end of the current time-slot, as this would enable immediate resource allocation for the arriving NSR. Finding a policy that can solve the SAC problem in polynomial time while considering the unpredictable nature in which requests can arrive, is challenging.

Primer on Online Multi-dimensional Knapsack Problems

The Online Multidimensional Knapsack Problem is one in which there exists a knapsack (*i.e.,* an edge server/gNodeB) whose capacity is represented by several dimensions (*i.e.,* $2 \leq m$), and where each dimension (*sic.* resource) could have a different capacity. In this problem, items (or slice requests), with a different value in each dimension of the knapsack, arrive in an online manner. The goal in such a problem is therefore to admit or reject items upon their arrival in a way that maximizes the overall profit of the knapsack from the admitted items (*i.e.,* slices), while also ensuring the capacity of the knapsack in *each* dimension is maximized, as well as ensuring that the SLA of new & existing slices are not violated while being served.

The OMdKP can be seen as a natural generalization and combination of the classic *single*-dimension knapsack problem but applied to multiple dimensions [193], and the *online* variant of the knapsack problem. Such a definition is an appropriate consideration to the online slice admission control problem applied to programmable and virtualized 5G networks, where each incoming slice request from a tenant could include multi-dimensional resources (*i.e.,* CPUs, Storage, Memory, GPUs, and FPGAs) of varying values, that span multiple domains (*i.e.,* Edge, RAN, and Cloud Core). Unlike online multiple knapsack problems where actions are required to determine admission and allocation, problems in the OMdKP domain are primarily concerned with admission decisions.

Proposed Solution

We address the challenge of approximating a solution for the considered online SAC problem based on it's reduction to the OMdKP. This enables us to leverage two policies[193] designed for the OMdKP, namely the LinRP and ExpRP, in our solution. The policies are based on online reservation functions that associate an implicit admission cost based on the utilization of each network resource. The intuition behind this is that by multiplying the current scarcity of each resource by the corresponding resource requirement of an NSR, the cost of admitting a request is effectively evaluated in an online manner. Furthermore, it is important to discourage high utilization of any resource in order to increase the chances of admitting higher valued future requests, as well as avoid performance degradation from resource sharing between co-located slices.

Let us define the *WTPR* θ, as the ratio between the maximum and minimum price per resource unit per time slot the tenants are willing to pay to reserve resources on the provider's infrastructure. The main purpose of θ is to quantify the spread (i.e., the variation) in the NSRs' utilities, and, therefore, scenarios with a higher value of θ lead to higher revenues.

However, such scenarios could likely have an impact on the admission rate due to the need for guaranteed QoS from high paying tenants. In this work, we assume that the value of θ is fixed and known in advance (Section 3.4). However, in real-world scenarios, the slice admission controller can learn the value of θ online by exploiting real-time information on sequentially arriving NSRs and by using data-driven algorithms. Let us define $u_{h,j}$ as the *utilization* of the network resource j after a NSR h has been accepted onto the infrastructure, and let us recall that C_j is the total capacity of the j-th resource in the network. Finally, let us define the *resource heterogeneity ratio* $\kappa_j = \frac{1}{C_j}\sum_{z\in[m]} C_z$, which captures the heterogeneity of the capacities of individual resources in the network (an important factor in the considered multi-dimensional setting). The details of the two policies are as follows.

- **Linear Reservation Policy (LinRP)**: In this policy, the scarcity of each network resource increases linearly based on the normalized utilization of the resource. Hence, the higher the utilization of a given resource is, the higher is the admission cost of admitting a new NSR that requests that resource. As a result, in periods where the overall network infrastructure has a higher load leading to higher resource utilization, the admission cost increases too to reserve the resources for higher valued requests. Therefore, we define the *normalized resource utilization* $q^{\text{lin}}_{h,j}$ and the *system admission cost* ϕ^{lin}_h for LinRP in Equations 3.2 and 3.3, respectively.

$$q^{\text{lin}}_{h,j} = \left\lfloor \frac{u_{h,j}}{C_j}\sqrt{\theta m} \right\rfloor \tag{3.2}$$

$$\phi^{\text{lin}}_h = \max_{j\in[m]} q_{h-1,j}\sqrt{\frac{2\kappa_j}{m}} r_{h,j} \tag{3.3}$$

- **Exponential Reservation Policy (ExpRP)**: The second online reservation function considered is the Exponential Reservation Policy (ExpRP). Intuitively, the scarcity of each resource in the system increases following an exponential reservation function and is also based on normalized resource utilization. Therefore, we define the *normalized resource utilization* $q^{\text{exp}}_{h,j}$ and the *system admission cost* ϕ^{exp}_h for ExpRP in Equations 3.4 and 3.5, respectively.

$$q^{\text{exp}}_{h,j} = \left\lfloor \frac{u_{h,j}}{C_j}\log\left(\theta\kappa_j\right) \right\rfloor \tag{3.4}$$

$$\phi^{\text{exp}}_h = \sum_{j=1}^{m} (2^{q_{h-1,j}} - 1) r_{h,j} \tag{3.5}$$

  Due to the exponential increase in the scarcity of resources, the ExpRP approach can be considered as a more conservative approach, compared to LinRP, which could be suitable for situations with very high resource utilization to ensure only the highest valued requests are accepted.

Algorithm 1 Online Slice Admission Control

Input: θ, κ_j, C_j, $\forall j \in [m]$
Output: x_h

1: $(h, u_0, q_0) \leftarrow (0, 0, 0)$
2: **loop**
3: $h \leftarrow h + 1$
4: $(\boldsymbol{r}_h, \delta_h, \pi_h) \leftarrow$ WaitNSR()
5: `// Update Admission Threshold (if LinRP)`
6: $\phi_h \leftarrow \phi_h^{\text{lin}} = \max_{j \in [m]} q_{h-1,j} \sqrt{\frac{2\kappa_j}{m}} r_{h,j}$
7: `// Update Admission Threshold (if ExpRP)`
8: $\phi_h \leftarrow \phi_h^{\text{exp}} = \sum_{j=1}^{m} (2^{q_{h-1,j}} - 1) r_{h,j}$
9: $x_h \leftarrow 0$
10: **if** $\pi_h \geq \phi_h$ **and** $r_{h,j} \leq C_j - u_{h-1,j}, \forall j \in [m]$ **then**
11: $x_h \leftarrow 1$
12: `// Update Resource Utilization`
13: $u_{h,j} \leftarrow u_{h-1,j} + x_h r_{h,j}, \forall j \in [m]$
14: **while** CheckFinishedNS() **do**
15: $\boldsymbol{r}_h \leftarrow$ GetFinishedNS()
16: `// Release Resources`
17: $u_{h,j} \leftarrow u_{h,j} - r_{h,j}, \forall j \in [m]$
18: `// Update Normalized Utilization (if LinRP)`
19: $q_{h,j} \leftarrow q_{h,j}^{\text{lin}} = \left\lfloor \frac{u_{h,j}}{C_j} \sqrt{\theta m} \right\rfloor, \forall j \in [m]$
20: `// Update Normalized Utilization (if ExpRP)`
21: $q_{h,j} \leftarrow q_{h,j}^{\text{exp}} = \left\lfloor \frac{u_{h,j}}{C_j} \log(\theta \kappa_j) \right\rfloor, \forall j \in [m]$
return x_h

Online Slice Admission Control Algorithm

Our algorithm works in a time-slotted fashion in which NSRs that arrive in a given time slot are handled in immediately, upon which they are either accepted and admitted onto the infrastructure or rejected immediately. We now describe the steps of Algorithm 1. When a new NSR arrives, we check its resource requirements (line 4). Then, the admission threshold is updated based on the current scarcity of each resource with the requested resource. When the LinRP policy is applied, the scarcity is based on the resource with the higher scarcity or normalized utilization (line 6), while with the ExpRP approach, the sum of scarcities in each dimension is summed up (line 8). A NSR meets the criteria for admission when the value of the request is higher than the admission threshold for either policy and there is enough space in the system to accommodate the resource requirements of the request, otherwise the request is rejected (lines 9-11). Based on the decision to accept or reject the NSR, we update the resource utilization in the system (line 13). Since we consider that each admitted slice is in the system for a certain period of time, at the end of each time slot we update the current resource utilization by releasing the allocated resources (line 14-17), and update the normalized utilization of the system too (lines 18-21).

3.4 Performance Evaluation

3.4.1 Simulation Setup

We compare the performance of the implemented ExpRP and LinRP policies with those of FCFS, an online greedy policy that admits arriving NSRs onto the network infrastructure if there is sufficient capacity. We implemented the three policies and the simulated scenarios in Python and publicly released the source code on GitHub[1]. As the literature does not offer public datasets on NSRs, we evaluate the performance of our proposed approach on a synthetic dataset $\mathcal{H}$ containing a sequence of $5 \cdot 10^7$ NSR slots for each simulation. The size of this dataset is sufficient to estimate the average, long-term performance of the proposed approach with high confidence. We assume that the system contains $m = 3$ resource types, namely CPU, RAM, and storage. Each NSR in the dataset h contains information about its duration δ_h, requested resources $\boldsymbol{r}_h$, revenue π_h, and arrival timestamp τ_h. We assume a time-slotted environment, where, during each time slot t, a random number X_t of NSRs enters the system. We model X_t as a Poisson process in which all $X_t, \forall t \in \{1, \dots, |T|\}$ follow a Poisson distribution Pois(λ) with identical arrival rate $\lambda = 2$, where λ is the average number of arrivals per slot.

To evaluate how the compared policies perform under variable amounts of requested resources, we assume that each component of the resource vector $\boldsymbol{r}_h$ requested by a NSR comes from a normalized uniform distribution $\mathcal{U}([0,1])$. Furthermore, for the sake of simplicity, we assume that resource capacities are also normalized, so $C_j = 1, \forall j \in [m]$. To simulate the impact of different durations on the system, we assume that the slice lifetimes δ_h are uniformly distributed as $\delta_h \sim \mathcal{U}(\{1, \zeta\})$, where ζ is the upper duration bound. To represent different economic conditions of tenants that issue NSRs, we model the slice *unit value* p_h as $p_h = 1 + (\sigma - 1)Y$, where Y is a random variable that follows a symmetric Beta(ω, ω) distribution, and $\sigma \geq 1$ is an *economic scale* factor such that $p_h \in [1, \sigma]$. The parameter $\omega \in (0, +\infty)$ represents the *economic inequality* of the tenants: when $\omega \to 0$ an increasing share of tenants will request slices with unit values close to 1 or to σ, while when $\omega \to +\infty$, tenants will offer increasingly similar unit values for their NSRs. We evaluated our approach in a range of scenarios, where $\omega \in \{0.05, 0.1, \dots, 1\}$, $\sigma \in \{10, 20 \dots, 100\}$, and $\zeta \in \{10, 30, 100\}$. In these scenarios, the values of $\max_{i \in \mathcal{H}} \delta_i p_i = \sigma\zeta$ and $\min_{i \in \mathcal{H}} \delta_i p_i = 1$. Therefore, the WTPR θ that characterizes each scenario is $\theta = \sigma\zeta / 1 = \sigma\zeta$. Furthermore, considering that $C_j = 1, \forall j \in [m]$, the resource heterogeneity ratio is $\forall j \in [m] : \kappa_j = m = 3$.

3.4.2 Performance Metrics

The following three metrics are used to evaluate our approach:

- **Average Revenue Relative Gain**. We define the *average revenue* as the ratio $\mu = \frac{1}{|\mathcal{H}|} \sum_{h \in \mathcal{H}} \pi_h$ between the total revenue and the number $|\mathcal{H}|$ of all received slice requests. We define the *average revenue relative gain* for the two policies LinRP and ExpRP as

[1] https://github.com/CDS-Bern/Online-Slice-Admission-Control

$\frac{\mu_L-\mu_F}{\mu_F}$ and $\frac{\mu_E-\mu_F}{\mu_F}$, respectively, where μ_L, μ_E, and μ_F indicate the average revenues for LinRP, ExpRP, and FCFS, respectively.

- **Acceptance Ratio Relative Gain**. We define the *acceptance ratio* $\eta = n/|\mathcal{H}|$ as the ratio between the number n of accepted slices and the number $|\mathcal{H}|$ of all received slice requests. The *acceptance ratio relative gain* for the two policies LinRP and ExpRP are defined as $\frac{\eta_L-\eta_F}{\eta_F}$ and $\frac{\eta_E-\eta_F}{\eta_F}$ respectively, where η_L, η_E, and η_F indicate the acceptance ratios for the LinRP, ExpRP, and FCFS policies, respectively.
- **Average Resource Utilization Relative Gain**. We define the *average resource utilization* $\rho = \frac{1}{|\mathcal{H}|}\sum_{h\in\mathcal{H}}\sum_{j\in[m]} u_{h,j}/C_j$ as the sum of the normalized utilization for all resources for all received slice requests divided by the number $|\mathcal{H}|$ of all received slice requests. The *average resource utilization relative gain* for the two policies LinRP and ExpRP are defined as $\frac{\rho_L-\rho_F}{\rho_F}$ and $\frac{\rho_E-\rho_F}{\rho_F}$ respectively, where ρ_L, ρ_E, and ρ_F indicate the average resource utilization for the LinRP, ExpRP, and FCFS policies, respectively.

3.4.3 Simulation Results

Average Revenue Relative Gain

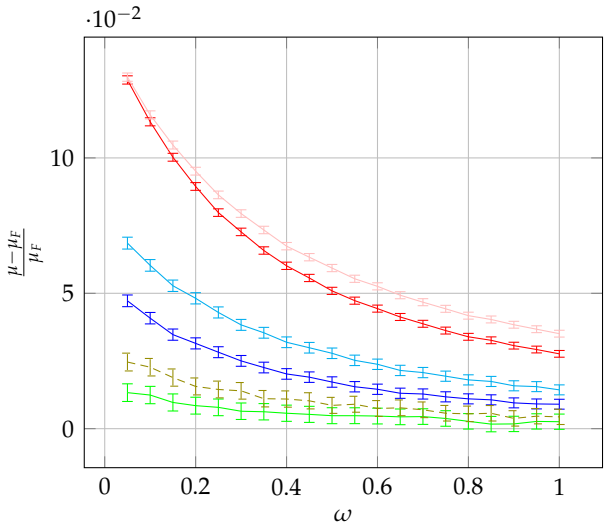
Figure 3.2a shows the relative gains of the LinRP (red, blue and green lines) and ExpRP (light red, light blue, and light green) policies, compared to the greedy (FCFS) policy for the mean revenue. The results are based on the slice holding time δ_h of each incoming NSRs, which is bounded by a specific duration, i.e., $\zeta \in \{10, 30, 100\}$. From this, we can see that applying both policies in our algorithm leads to higher average revenue than the greedy policy across various upper bounds on the NSR durations and for the range of unit values considered. Specifically, it can be observed that in situations with higher economic inequality between tenants ($\omega \to 0$), the revenue gain is the highest (over 10%) when using both policies. This is because, in such scenarios, the value of the accepted NSRs is higher due to the online reservation function that places a higher threshold on the values that can be accepted. The results shown are for confidence intervals at level 99.9%.

Acceptance Ratio Relative Gain

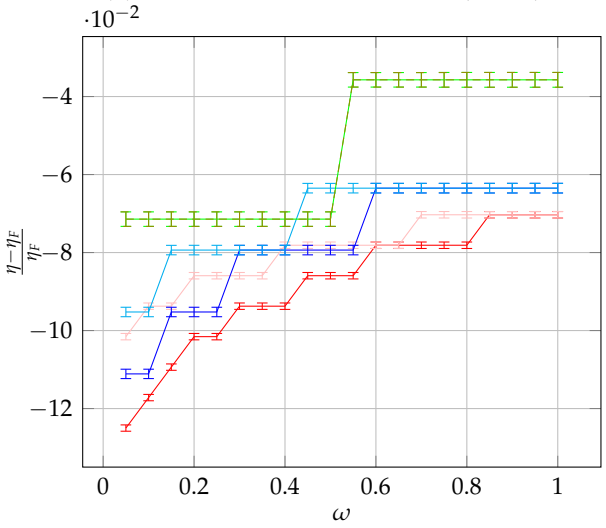
The relative gains in the Acceptance Ratio (AR) by using the online reservation-based policies, are shown in Figure 3.2b. We see that by utilizing the introduced policies in our approach, fewer NSRs are admitted compared to the greedy FCFS approach for the considered range of upper-bounded NSR durations times. The performance of our solution improves when the average duration of requests increases and when the value of ω approaches 1. This is primarily due to the admission criteria of the reservation-based policies compared to that of the greedy policy, where the proposed solution rejects lower-valued requests at a higher rate compared to higher-valued slice requests with longer durations. This leads to a lower overall acceptance ratio for the considered parameters, but a higher revenue, as seen in the previous results for *Average Revenue Relative Gain*. Here again, the results shown represent confidence intervals at level 99.9%.

Average Resource Relative Utilization

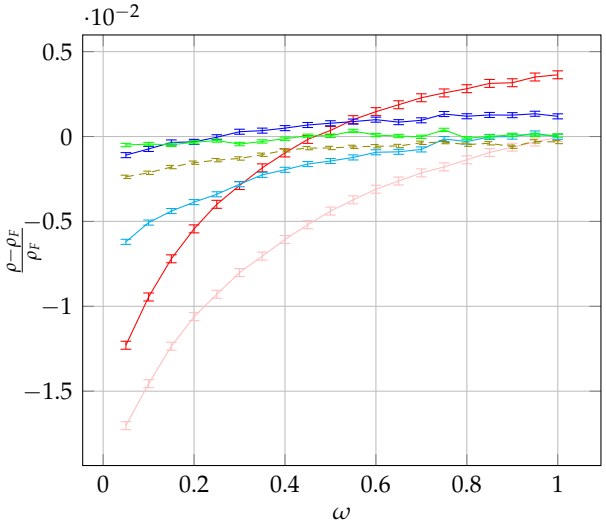
It can be seen in Figure 3.2c, that across most of the evaluated upper-bounded duration and unit price parameters, the resource utilization is lower on average using our approach. This is in line with the results from the acceptance ratio gain, which shows that based on the reservation-based policies our approach leads to a lower overall acceptance ratio, and hence, there are periods during the experiments when the resources are not fully utilized but new NSRs are rejected due to their revenue and the current utilization level of any given resource in the system. Achieving a lower average utilization is key to achieving the goal of maximizing the revenue for InPs as it ensures that the scarce resources are reserved for the NSRs that offer the most revenue. The displayed results show confidence intervals at level 99.9%.



(A) Average Revenue Relative Gain ($\lambda = 10$)



(B) Acceptance Ratio Relative Gain ($\lambda = 10$)



(C) Average Resource Utilization Gain ($\lambda = 10$)

FIGURE 3.2: Relative Gain for LinRP and ExpRP against FCFS for different values of ω and $\sigma = 10$.

3.5 Chapter Conclusions

This Chapter primarily investigated the problem of online admission control of NSLs in mobile networks. In order to address the problem, we modeled the problem as an Online Multidimensional Knapsack Problem in which decisions on sequentially arriving NSRs which offer the InP some revenue must be made, with the goal of maximizing the long-term revenue gained from accepting requests in the network. Specifically, we focus on the dynamic nature of mobile networks by considering an online and multidimensional scenario in which NSRs contain multiple resources which must be considered during the admission process in order to ensure that the SLAs of a NSL are met during it's lifetime. This problem is exacerbated in this scenario as decisions must be based on the currently available information only, and in the absence of future information. To address the problem, we proposed a novel online solution that leverages reservation-based policies to determine wether or not NSRs are admitted onto the InPs's virtualized mobile network infrastructure. To evaluate our proposed solution, we consider scenarios that represent different economic conditions of tenants that issue NSRs and show that our proposed solution is able to outperform the greedy FCFS baseline solution in terms of revenue gained from accepted requests, while accepting fewer NSRs and saving on the consumption of system resources.

However, our proposed solution, OSAC, assumes that the reservation policies utilized are optimal across all the evaluated scenarios and that the distribution of network slice request parameters are fixed and stationary over time, which isn't necessarily the case in real-networks. In Chapter 4, we present a method for detecting changes, or drifts, in the underlying parameters of network slice requests that define a scenario (*i.e.,* a given WTPR), and for dynamically selecting the optimal admission policy by learning how different algorithms perform across various scenarios, in an online manner.

Chapter 4

Data-Driven Online Policy Selection

4.1 Introduction

For 5G Infrastructure Providers (InPs), the efficient management of network resources is becoming an increasingly critical challenge in multi-tenant, multi-service and multi-domain environments. As a result, Slice Admission Control (SAC) emerges as a fundamental component of resource management, tasked with determining which Network Slice Requests (NSRs) to admit based on available resources and the objectives of the NSRs. Traditional approaches to admission control often employ static policies that make reservation of network resources for a specified duration of time based on instantaneous network states without considering the temporal dynamics of the network environment [194]. Such approaches, while computationally efficient, fail to capture the inherent uncertainties and variations that characterize real-world mobile networks [195].

5G network systems are inherently dynamic and time-varying, exhibiting fluctuations in resource availability, user mobility patterns, and traffic demands [61], [196]. These time-varying dynamics introduce what can be referred to as *drift* in the network state *i.e.,* a systematic shift in network characteristics or conditions that evolve over time [41]. Conventional admission control policies that neglect these drifts may lead to suboptimal resource utilization, increased Service Level Agreements (SLAs) violations, and reduced revenue for InPs. The effectiveness of an admission control policy is therefore contingent on its ability to adapt to changing network conditions[195]. In the context of slice admission control in mobile networks, the challenge of adaptive online policy selection has not received sufficient attention in the literature. However, due to the time-varying network dynamics and admission cost functions involved, it is an important consideration since admission decisions must be made sequentially along a single continuous trajectory, rather than restarting from an identical initial network state to assess alternative admission control policies. [197].

This chapter introduces a novel framework for drift-aware policy selection in 5G networks, to address the problem of SAC. We propose a meta-decision system that dynamically selects the most appropriate admission control policy from a portfolio of candidate policies based

on detected network drift patterns. By incorporating drift awareness, our approach enables proactive adaptation to evolving network conditions, thereby enhancing long-term system performance. We formulate the Slice Admission Control Policy Selection (SACPS) problem as a Multi-Armed Bandit (MAB), and develop a framework for addressing this problem by characterizing drifts in the patterns of slice requests. This enables us to establish a mechanism for mapping optimal admission control policies to detected drifts conditions. Through extensive simulations using realistic network scenarios, we demonstrate that our drift-aware approach significantly outperforms static policy selection methods across multiple performance metrics, including resource utilization efficiency, revenue generation and the acceptance of NSRs.

The contributions of this chapter aim to answer the following research questions described in Section 1.3.2.

RQ 2.1: How can admission control policies be **dynamically adapted** to cope with evolving network slice request patterns in 5G networks?

RQ 2.2: What is the effect of **temporal shifts** in the distribution of network slice request characteristics on the performance and robustness of admission control policies?

RQ 2.3: How can online learning techniques be integrated with concept drift detection to enable **continuous adaptation** of slice admission control policies in dynamic environments?

We address the above mentioned challenges, by proposing Drift-AwaRe upper confIdence bOund (DARIO), a data-driven solution to adaptive policy selection for slice admission control. We address ***RQ 2.1*** by formulating the policy selection problem as an MAB problem to address the exploration-exploitation trade-off in the problem. We address ***RQ 2.2*** by creating a synthetic dataset of NSRs that contains distribution shifts to evaluate the performance of the baseline admission policies across time-varying distributions. Based on these observations, we design a change detection mechanism based on ADaptive WINdowing (ADWIN) to detect the changes in the statistical features of NSRs which could trigger admission policy adjustments through adaptive thresholds. The ***RQ 2.3*** is addressed by integrating the Sliding-Window Upper Confidence Bound (SW-UCB) algorithm with the change detection technique to create a data-driven, drift-aware online learning framework that learns the performance of parametric admission policies, by updating their parameters based on the estimated changes in the historical request patterns.

Our contributions can be summarized as follows.

- We study the problem of online SAC and focus on the challenge of adaptively selecting between a set of AC policies for this problem. We formulate the policy selection problem as a MAB problem.
- To address the MAB problem, we design a framework, DARIO, that aims to learn the performance of the SAC policies and to identify efficient admission policies across time-varying scenarios.

- Through extensive evaluations, we show that our proposed framework is able to outperform the Single Best (SB) Policy (i.e., a static policy), as well as a vanilla UCB selection policy.

The following sections discuss the content in our paper published in the 2024 IEEE Network Operations and Management Symposium (NOMS) [195]. Section 4.1 describes the research questions addressed in this chapter and discusses the contributions of the proposed learning-based framework. Section 4.2 describes the system model and formulates the PS problem in the considered SAC context. Section 4.3 presents the DARIO framework in more detail. Section 4.4 describes the simulation setup and discusses the evaluation results. Finally, section 4.5 concludes the study in this chapter by reviewing the main contributions and highlighting the important results.

4.2 System Model and Problem Formulation

In this section, we describe the considered NSL scenario, while also describing the infrastructure and NSR models. Finally, we formulate the considered SACPS problem.

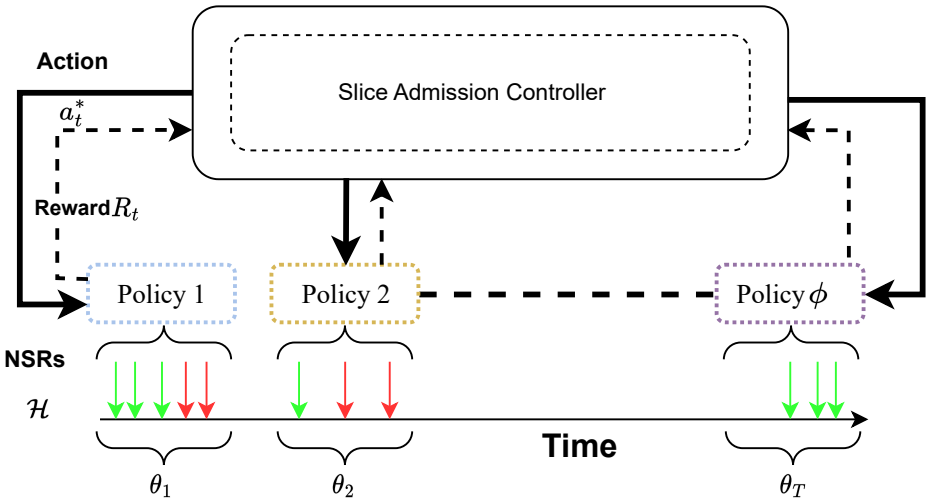


FIGURE 4.1: Policy Selection Problem for SAC

To model the problem of adaptive selection of AC policies, we first consider a sequence $\mathcal{B} = \{b_1, b_2, ..., b_T\}$ of problem instances, where each problem instance $b_t \in \mathcal{B}$ represents an instance of the online SAC problem [15] in which a decision is required on the admission of arriving NSRs in order to maximize the long-term revenue received by the InP. Furthermore, we consider a set $\mathcal{A} = \{A_1, A_2, ..., A_N\}$ of policies such that each problem instance b_t can be solved by at least one policy $A_n \in \mathcal{A}$, where $n \in \mathcal{N}$ and the elements of $\mathcal{N} = \{1, 2, ..., N\}$ represent the index of an admission policy. In such a setting, the goal is to select the policy that performs best according to the optimization objective of the problem instance [198]. This goal could be achieved either by identifying the Single Best policy for the whole set of problem instances or, the best policy for each instance of the problem. However, in this work, we consider a more general setting in which a selection policy Φ is to be sequentially learned for the adaptive selection of different admission control policies, where each policy is expected to perform well on different subsets

TABLE 4.1: Summary of Notations

Symbol	Description
m	Number of resource types
$\mathcal{H}$	Index set of NSRs
$\mathcal{B}$	Set of SAC problem instances
$\boldsymbol{r}_{h,j}$	Requested amount of resource j by NSR h
δ_h	Lifetime of NSR h
ι_h	Timestamp of NSR h
ν_h	Revenue of NSR h
p_h	Unit value of NSRs h
α_h	Coefficient vector of NSRs h
C_j	Aggregated capacity of resource j
x_h	Decision variable on NSR h
κ_j	Resource heterogeneity ratio for resource j
$\theta/\hat{\theta}$	Willingness-To-Pay-Ratio or Estimate of WTPR
$\mathcal{A}$	Set of AC policies
A_t	Arm (or policy) selected at time t
R_t	Reward at the time t
$q_*(a)$	Value of arm a^* at time t
$\mu_{a_t^*}/\mu_{a_t}$	Mean of arm a^* or a at time t

of the problem instances $\mathcal{B}$. In both scenarios, there is a clear trade-off between *exploring* the performance of different admission control policies over the set of problem instances $\mathcal{B}$ and *exploiting* the estimated optimal admission control policy based on the performance on a subset of problem instances. Formally, such a problem can be modeled in the MAB setting, which is a well-known paradigm for dealing with this trade-off.

We consider that the set of applicable admission control policies are the Linear Reservation Policy (LinRP) and Exponential Reservation Policy (ExpRP) [193], which use linear and exponential reservation functions, respectively, to determine the admission thresholds for incoming network slice requests. Such threshold-based policies could be said to characterize the *attitude* of an InP toward the uncertain utilities of future NSRs. That is, they determine how much of the currently available network resources should be reserved in advance for the possible arrival of future NSRs that could offer a higher utility. This is done by evaluating the marginal utility of each network resource at its current utilization level.

By modeling the SACPS problem as an MAB problem, we consider the scenario in which the range of offered utilities in the sequentially arriving NSRs are unknown in advance but must be estimated by the considered AC policies to adaptively determine their admission thresholds. We quantify the variation in the NSRs utilities by defining the Willingness-To-Pay-Ratio (WTPR) $\theta = \frac{p_{\max}}{p_{\min}}$ as the ratio between the maximum and minimum price per resource unit per time slot the tenants are willing to pay for resource reservations on the InP's infrastructure, i.e., $p_{\max}$ and $p_{\min}$ are the upper and lower bound on the price per resource unit per time slot i.e., $p_h \in [p_{\min}, p_{\max}]$. As $p_{\min}$ and $p_{\max}$ may vary between time slots, we assume that θ can also fluctuate over time.

Let us define $\mathcal{H}_t$ as the set of the slice requests received by the slice admission controller during time slot t, where each time slot effectively represents a period (i.e. a few minutes) in which a certain admission control A_t policy is applied. The decision variable $x_h \in \{0,1\}$ is 1 if request h is accepted and 0 otherwise. We assume that the slice admission controller selects an admission policy at the time slot's beginning and keeps it the same for the whole duration of the time slot. This assumption means that all slice requests received during a time slot are subject to the same acceptance policy (i.e., the bandit's arm).

A key challenge in designing an appropriate solution for the formulated SACPS problem is determining a reasonable performance metric that can be used to compare policies that earn different utilities (i.e., amounts of revenue), and utilize different amounts of resources [199]. A direct comparison of the revenues earned as a result of applying the decisions of admission control policies, leaves out the fact that different policies can have distinct admission thresholds and therefore can admit requests at different rates. This leads to the selection and application of each policy utilizing varying amounts of resources which potentially leaves fewer resources available in the future and impacts the decisions of subsequent policies. We address by defining the reward R_t (Equation 4.1) for time slot t as the ratio between the actual revenue gained by applying policy a and the total revenue that would be gained if all slice requests were accepted, towards the maximization objective of the InP's revenue:

$$R_t = \frac{\sum\limits_{h \in \mathcal{H}_t} \nu_h x_h}{\sum\limits_{h \in \mathcal{H}_t} \nu_h}. \tag{4.1}$$

We assume that the average reward of each arm R_t is an unknown mean μ_a, where $a \in \mathcal{A}$. The advantage of such a reward formulation is that $R_t \in [0,1]$, which allows the application of bandit-based algorithms [200]. We define the action $A_t \in \mathcal{A} = \{\texttt{LinRP}, \texttt{ExpRP}\}$ at time slot t as the admission policy selected by the slice controller at that time slot. We define the value of an arm a (i.e., the choice of a policy) as $q_*(a) = \mathbb{E}[R_t | A_t = a]$.

Let us define $\mu_{a_t^*}$ and μ_{a_t} to denote the expectation of rewards $R_t(a_t^*)$ and $R_t(a_t)$, respectively, at time slot t, i.e. $\mu_{a_t^*} = \mathbb{E}[R_t(a_t^*)]$ and $\mu_{a_t} = \mathbb{E}[R_t(a_t)]$, and T is the time horizon. Our MAB model aims to minimize the expected cumulative regret $\mathbb{E}[R(T)]$ for T arm plays (Equation 4.2), which is the difference between the rewards obtained through the chosen arm and the optimal arm (i.e. the SB policy or benchmark) $a_t^* = \mathsf{argmax}_a \ \mathbb{E}[R_t(a)]$ at time t:

$$\mathbb{E}[R(T)] = \mathbb{E}\left[\sum_{t \in T}(R_t(a_t^*) - R_t(a_t))\right] = \sum_{t \in T}(\mu_{a_t^*} - \mu_{a_t}) \tag{4.2}$$

4.3 Drift-Aware Policy Selection

To address the formulated policy selection problem, DARIO aims to select the optimal policy that maximizes the long-term reward by learning the performance of the policies for SAC across different scenarios, each characterized by a time-varying, unknown WTPR θ_t at time t,

estimated by an estimator:

$$\hat{\theta}_t = \frac{\max_{h \in \mathcal{H}_t} p_h}{\min_{h \in \mathcal{H}_t} p_h} \tag{4.3}$$

which is computed on the arriving NSR's features. We note that under the *full* feedback setting, such a problem could be addressed by keeping a running average of the reward of each arm (or policy) and playing the arm with the best estimate, as is done by the Follow-The-Leader (FTL) meta-algorithm. However, this gives rise to a purely exploitation-based approach, which could lead to a sub-optimal arm being constantly chosen despite the potentially better performance of other arms across different scenarios. As a result, the regret achieved by such an approach grows linearly over time [201]. Furthermore, this approach is likely to impose high overhead as the performance of the adjacent policies in $\mathcal{A}$ would need to be considered during the selection of the AC policy. Hence, in the design of DARIO, we consider a more flexible setting with bandit feedback. In particular, DARIO builds upon the SW-UCB algorithm [202] by incorporating a change-detection mechanism to monitor Concept Drift (CD). The change detection mechanism enables the framework to learn the performance of the policies by adapting the selection policy Φ across different scenarios, based on the WTPR estimate $\hat{\theta}_t$. This is achieved by updating the thresholds of the admission policies and resetting the selection policy when a drift in $\hat{\theta}_t$ is detected in order to compensate for the changing features of the arriving NSRs (Figure 4.2).

4.3.1 Change Detection

Our framework aims to learn the performance of the admission control algorithms in various scenarios by incorporating a change-detection technique to monitor the presence of CD. CD is the phenomenon in which the statistical features of a target domain change over time and in an arbitrary manner [203]. The CD mechanism used to detect the changes in the underlying distribution of unit values is presented in Algorithm 2 and is based on an ADWIN technique [204]. We chose this technique as its able to work with any kind of real-valued input and does not require any knowledge regarding the input distribution [205]. Detecting the distribution shifts of unit values is crucial in ensuring the robustness of our learning framework to CD scenarios based on the historical characteristics or patterns of slice requests.

To estimate the WTPR $\{\hat{\theta}_t\}_{t=1}^{T}$ in an online manner, an estimate of the distribution of NSR unit values is required. The algorithm detects whether there has been a significant change (drift) in the underlying data distribution of arriving NSRs characteristics by statistically comparing two halves of a sliding window containing historical WTPR estimates. The algorithm begins by splitting the input window W into two equal sub-windows (lines 3-4): W_0 containing the older half of the data (elements from index 0 to $|W|/2$-1) and W_1 containing the newer half (elements from index $|W|/2$ to $|W|$). It then computes the harmonic mean m (line 6) of the two sub-window lengths, which equals |W|/2 since both halves are equal in size, and uses this to calculate a drift detection threshold (line 8) $\varepsilon = \sqrt{\frac{1}{2m} \cdot \ln \frac{4|W|}{\alpha}}$ based on Hoeffding's concentration inequality [204]. This threshold accounts for both the window size (larger windows require larger differences to indicate drift) and the desired confidence level α (smaller α values make drift detection more sensitive). The algorithm then computes the empirical averages $\hat{\mu}_0$ and $\hat{\mu}_1$ of the WTPR estimates in each sub-window respectively, representing the

Algorithm 2 Concept Drift

Require: Sliding Window Size W, Error Threshold α

1: **function** ConceptDrift(W, α)
2: ▷ Split window in two subwindows
3: $W_0 \leftarrow \text{Subwindow}(0, |W|/2 - 1)$
4: $W_1 \leftarrow \text{Subwindow}(|W|/2, |W|)$
5: ▷ Harmonic mean of subwindows length
6: $m \leftarrow \frac{1}{\frac{1}{|W_0|} + \frac{1}{|W_1|}}$
7: ▷ Compute drift detection threshold
8: $\varepsilon \leftarrow \sqrt{\frac{1}{2m} \cdot \ln \frac{4|W|}{\alpha}}$
9: ▷ Compute empirical averages of the estimator in half-windows
10: $\hat{\mu}_0 \leftarrow \frac{2}{|W|} \sum_{t=0}^{|W|/2-1} \hat{\theta}_t$
11: $\hat{\mu}_1 \leftarrow \frac{2}{|W|} \sum_{t=|W|/2}^{|W|} \hat{\theta}_t$
12: **if** $|\hat{\mu}_0 - \hat{\mu}_1| > \varepsilon$ **then**
13: ▷ Discard old window W_0
14: $W \leftarrow W_1$
15: **return** W

mean behavior in the older and newer portions of the data (lines 10-11). Finally, it performs the drift test by comparing the absolute difference $|\hat{\mu}_0 - \hat{\mu}_1|$ against the threshold ε (line 12): if the difference exceeds the threshold, it indicates that the underlying distribution has changed significantly, prompting the algorithm to discard the older sub-window W_0 and return only the newer sub-window W_1 to maintain relevance (line 14); otherwise, it returns the original window unchanged (line 15). This approach provides a statistically principled method for the DARIO algorithm to adapt to changing network conditions by ensuring that only relevant, recent data influences the bandit learning process, with the trade-off that smaller α values lead to more responsive but potentially noisier adaptation, while larger α values provide more stability but slower response to genuine changes.

4.3.2 Learning Admission Policies under Concept Drift

DARIO estimates $\hat{\theta}_t$ upon drift detection and seeks to learn the performance of the admission policies (Section 4.2) in the following Sliding Window (SW), based on this estimate. This is done by considering the number $N_t(\tau, a)$ of times each arm a, is selected by the SW-UCB algorithm during the last τ time slots, which is represented by:

$$N_t(\tau, a) = \sum_{s=\max\{0,t-\tau+1\}}^{t} \mathbb{1}_{\mathcal{A}_s=a}, \tag{4.4}$$

where $\mathbb{1}_{\mathcal{A}_s=a}$ is an indicator function that evaluates to 1 when the arm selected within time-slot s is a. Furthermore, we define the empirical average reward of an arm up to the current decision interval t and define the $\overline{R}_t(\tau, a)$ for this, which can be calculated using [202]:

$$\overline{R}_t(\tau, a) = \frac{1}{N_t(\tau, a)} \sum_{s=\max\{0,t-\tau+1\}}^{t} R_s \mathbb{1}_{\mathcal{A}_s=a}, \tag{4.5}$$

where R_s represents the sum of rewards of policy a during time-slot s. Based on the SW-UCB approach, the framework learns the average performance of each algorithm's decisions in a given SW. Therefore, we define the bandit's sequential arm selection policy at a given time slot Φ_t as:

$$\Phi_t = \underset{a \in \mathcal{A}}{\arg\max} \quad \overline{R}_t(\tau, a) + c\sqrt{\frac{\log(t \wedge \tau)}{N_t(\tau, a)}}, \tag{4.6}$$

where the second addend is used to encourage the exploration phase during the learning process and $(t \wedge \tau)$ denotes the minimum of t and τ [202]. Based on $N_t(\tau, a)$, (Equation 4.5) and Equation (4.6), the framework learns the performance of each arm within a given SW and, subsequently, across different scenarios by adjusting the learned policy through the feedback information and the concept drift detection. Hence, it learns the performance of the admission control policies in dynamic network conditions which are captured by the dynamic value of $\hat{\theta}$.

4.3.3 Proposed Solution

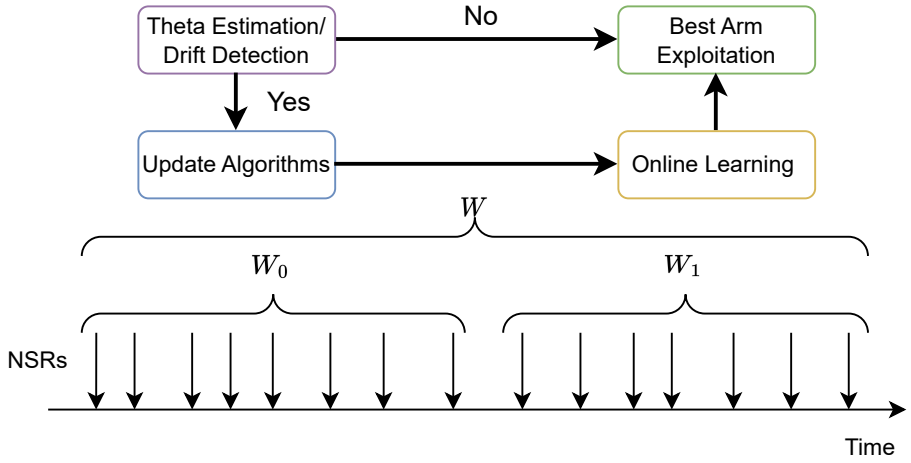


FIGURE 4.2: DARIO Workflow

The DARIO algorithm is a MAB-based approach that dynamically selects between different network slice admission policies, while adapting to concept drift in network slice request patterns. Algorithm 3 shows DARIO's operation. The algorithm begins by initializing key parameters including time slot counter t, initial resource utilization u_0 and initial normalized resource utilization q_0, an empty sliding window W for storing WTPR estimates, and the resource heterogeneity ratio for resource j, κ_j, which is set to 5 for each resource type $j \in [m]$ (line 1). At each time slot, DARIO first collects incoming NSRs during the current period (line 4), then estimates the WTPR $\hat{\theta}_t$ by computing the ratio between the maximum and minimum request values among all arriving requests (line 6), which captures the request heterogeneity and resource demands. This WTPR estimate is appended to the sliding window W (line 8), which is then processed through the ConceptDrift algorithm that splits the window into older and newer halves, computes a statistical threshold based on Hoeffding's inequality, compares the empirical averages of both halves, and discards the older half if significant drift is detected, ensuring that only relevant recent data influences future decisions (line 10). Using the updated

Algorithm 3 **D**rift-**A**ware uppe**R** conf**I**dence b**O**und (DARIO)

Require: Sliding Window Size w, Drift sensitivity coefficient α

1: $t, h, u_0, q_0 \leftarrow 0$, $W \leftarrow \emptyset$, $\kappa_j \leftarrow 5$
2: **loop**
3: ▷ Collect NSRs arriving during timeslot t
4: $\mathcal{H}_t \leftarrow \text{WaitNSRs}(t)$
5: ▷ Estimate the WTPR for current time slot
6: $\hat{\theta}_t \leftarrow \max_{h \in \mathcal{H}_t} p_h / \min_{h \in \mathcal{H}_t} p_h$
7: ▷ Append the WTPR estimation to the window
8: $W \leftarrow W \cup \{\hat{\theta}_t\}$
9: ▷ Update window if CD detected
10: $W \leftarrow \text{ConceptDrift}(W, \alpha)$
11: ▷ τ is the window size in timeslots
12: $\tau \leftarrow |W|$
13: ▷ Prepare bandit over τ window
14: **for** $\phi \in \mathcal{A}$ **do**
15: $N_t(\tau, \phi) \leftarrow \sum_{s=\max\{0,t-\tau+1\}}^{t} \mathbb{1}_{\mathcal{A}_s=\phi}$
16: $\overline{R}_t(\tau, \phi) \leftarrow \frac{1}{N_t(\tau,\phi)} \sum_{s=\max\{0,t-\tau+1\}}^{t} R_s(\phi) \mathbb{1}_{\mathcal{A}_s=\phi}$
17: ▷ Apply the SW-UCB policy over τ window
18: $\Phi_t \leftarrow \arg\max_{\phi \in \mathcal{A}} \left[\overline{R}_t(\tau, \phi) + c \sqrt{\frac{\log(t \wedge \tau)}{N_t(\tau,\phi)}} \right]$
19: **if** $\Phi_t =$ `LinRP` **then**
20: ▷ Apply Linear Reservation Policy and collect reward
21: $R_t \leftarrow \text{RewardLinRP}(\mathcal{H}_t, \hat{\theta}_t)$
22: **else if** $\Phi_t =$ `ExpRP` **then**
23: ▷ Apply Exponential Reservation Policy and collect reward
24: $R_t \leftarrow \text{RewardExpRP}(\mathcal{H}_t, \hat{\theta}_t)$
25: ▷ Release resources of finished NSRs
26: $\boldsymbol{r}_h \leftarrow \text{GetFinishedNS}(t)$
27: $u_{h,j} \leftarrow u_{h,j} - r_{h,j}, \forall j \in [m]$
28: ▷ Estimate Average Reward
29: $\hat{R}_t \leftarrow \frac{1}{t} \sum_{s=0}^{t} R_s$
30: $t \leftarrow t + 1$

window of size τ, DARIO applies a SW-UCB policy by computing for each policy ϕ in the action set $\{\texttt{LinRP}, \texttt{ExpRP}\}$ the number of times it was selected $N_t(\tau, \phi)$ and its average reward $\overline{R}_t(\tau, \phi)$ within the current window, then selects the policy that maximizes the upper confidence bound $\overline{R}_t(\tau, \phi) + c\sqrt{\frac{\log(t \wedge \tau)}{N_t(\tau,\phi)}}$ which balances exploitation of historically good policies with exploration of potentially better alternatives (lines 12-18). When LinRP is selected (line 19), a subroutine (algorithm 4) computes an admission threshold (line 4) $\phi_h = \max_{j \in [m]} q_{h-1,j} \sqrt{\frac{2\kappa_j}{m}} r_{h,j}$ that scales linearly with resource utilization levels and request resource demands, admits a slice if its value v_h exceeds this threshold and sufficient resources are available across all dimensions (lines 5-6), or rejects otherwise (line 8), updates resource utilization $u_{h,j}$ by adding the allocated resources (line 10), and updates normalized resource utilization $q_{h,j}$ using a square-root scaling formula $\left\lfloor \frac{u_{h,j}}{C_j} \sqrt{\hat{\theta}_t m} \right\rfloor$ that grows moderately with utilization (line 11). Alternatively, when ExpRP is selected, another subroutine (algorithm 5) computes an exponential admission threshold (line 4)

Algorithm 4 Linear Reservation Policy (LinRP)

1: **function** RewardLinRP($\mathcal{H}_t$, $\hat{\theta}_t$)
2: **for** $h \in \mathcal{H}_t$ **do**
3: **Update Admission Threshold**
4: $\phi_h \leftarrow \max_{j\in[m]} q_{h-1,j}\sqrt{\frac{2\kappa_j}{m}} r_{h,j}$
5: **if** $\nu_h \geq \phi_h$ **and** $r_{h,j} \leq C_j - u_{h-1,j}, \forall j \in [m]$ **then**
6: $x_h \leftarrow 1$ ▷ Admit slice
7: **else**
8: $x_h \leftarrow 0$ ▷ Reject slice
9: **Update Resource Utilization**
10: $u_{h,j} \leftarrow u_{h-1,j} + x_h r_{h,j}, \forall j \in [m]$
11: $q_{h,j} \leftarrow \left\lfloor \frac{u_{h,j}}{C_j}\sqrt{\hat{\theta}_t m} \right\rfloor, \forall j \in [m]$
12: **return** $R_t = \frac{\sum_{h\in\mathcal{H}_t} \nu_h x_h}{\sum_{h\in\mathcal{H}_t} \nu_h}$

Algorithm 5 Exponential Reservation Policy (ExpRP)

1: **function** RewardExpRP($\mathcal{H}_t$, $\hat{\theta}_t$)
2: **for** $h \in \mathcal{H}_t$ **do**
3: **Update Admission Threshold**
4: $\phi_h \leftarrow \sum_{j=1}^{m} (2^{q_{h-1,j}} - 1) r_{h,j}$
5: **if** $\nu_h \geq \phi_h$ **and** $r_{h,j} \leq C_j - u_{h-1,j}, \forall j \in [m]$ **then**
6: $x_h \leftarrow 1$ ▷ Admit slice
7: **else**
8: $x_h \leftarrow 0$ ▷ Reject slice
9: **Update Resource Utilization**
10: $u_{h,j} \leftarrow u_{h-1,j} + x_h r_{h,j}, \forall j \in [m]$
11: $q_{h,j} \leftarrow \left\lfloor \frac{u_{h,j}}{C_j} \log\left(\hat{\theta}_t \kappa_j\right) \right\rfloor, \forall j \in [m]$
12: **return** $R_t = \frac{\sum_{h\in\mathcal{H}_t} \nu_h x_h}{\sum_{h\in\mathcal{H}_t} \nu_h}$

$\phi_h = \sum_{j=1}^{m} (2^{q_{h-1,j}} - 1) r_{h,j}$ that grows exponentially with normalized resource utilization levels, creating increasingly restrictive admission conditions as resources become scarce, admits or rejects NSRs using the same value and resource availability criteria (lines 5-8), and updates normalized resource utilization levels using a logarithmic formula $\left\lfloor \frac{u_{h,j}}{C_j} \log(\hat{\theta}_t \kappa_j) \right\rfloor$ that grows more aggressively than LinRP's square-root approach (lines 10-11). After applying the selected policy, DARIO computes the reward R_t as the fraction of total request value successfully admitted (lines 21-24), releases resources from NSRs that have completed their service duration (lines 26-27), updates the cumulative average reward estimate (line 29), and increments the time counter before repeating the entire process (line 30).

This comprehensive approach enables DARIO to automatically adapt between conservative linear resource reservation LinRP, which suitable for stable, predictable NSR patterns and aggressive exponential resource reservation ExpRP, which is more appropriate for highly

variable, competitive scenarios. The drift detection mechanism ensures rapid adaptation to changing network conditions by maintaining only statistically relevant historical data, and the UCB-based policy provides guarantees on regret minimization while balancing the exploration-exploitation trade-off inherent in learning optimal reservation strategies under uncertainty [206].

4.4 Performance Evaluation

4.4.1 Simulation Setup

We evaluated DARIO's performance against state-of-the-art baselines in a range of simulated scenarios. The simulations, proposed solution, and benchmark algorithms were implemented in Python to simulate the admission process of diverse NSRs and different network scenarios. As the literature does not offer publicly available datasets on the time-varying characteristics of NSRs, we evaluate the performance of our proposed approach on a synthetic dataset $\mathcal{H}$ containing a sequence of 10^7 NSRs for each simulation. The size of this dataset is sufficient to estimate the long-term average performance of DARIO with high confidence. The system contains $m = 5$ resource types (Section 4.2). In the constructed dataset, each NSR h contains information about its duration δ_h, requested resources $\boldsymbol{r}_h$, revenue ν_h, and arrival timestamp ι_h. In the time-slotted environment, during a time slot t, a random number X_t of NSRs enters the system. We model X_t as a Poisson process in which all $X_t, \forall t \in \{1, \ldots, |T|\}$ follow a Poisson distribution Pois(λ) with identical arrival rate λ, where λ is the average number of arrivals per slot. We evaluate DARIO's performance under various load conditions by varying λ, specifically when $\lambda = 4$ and when $\lambda = 10$. Finally, we set the error threshold α defined in Algorithm 2, to 0.002 as a result of empirical observations on the generated dataset.

Dataset Generation

Generating a dataset with distribution shifts enables the evaluation of our framework across potentially different distributions. To generate the synthetic dataset used this in work, we assume that each component of the resource vector $\boldsymbol{r}_h$ requested by an NSR comes from a normalized uniform distribution $\mathcal{U}([0,1])$, and, for simplicity, we assume that the resource capacities are also normalized, so $C_j = 1, \forall j \in [m]$. Given that $C_j = 1$, the resource heterogeneity ratio is $\forall j \in [m] : \kappa_j = m = 5$. To simulate the impact of different slice durations on the system, we assume that the slice lifetimes δ_h are uniformly distributed as $\delta_h \sim \mathcal{U}(\{1, \zeta\})$, where ζ is the upper duration bound. To represent the different conditions of tenants that issue NSRs, we model the slice *unit value* p_h as $p_h = 1 + (\sigma - 1)Y$, where Y is a random variable that follows a symmetric Beta(ω, ω) distribution, and $\sigma \geq 1$ is a scaling factor such that $p_h \in [1, \sigma]$. The parameter $\omega \in (0, +\infty)$ represents the *inequality* in the conditions of the tenants: when $\omega \to 0$ an increasing share of tenants will request slices with unit values close to 1 or to σ, while when $\omega \to +\infty$, tenants will offer increasingly similar unit values for their NSRs. In these scenarios, the values of $\max_{i \in \mathcal{H}} \delta_i p_i = \sigma\zeta$ and $\min_{i \in \mathcal{H}} \delta_i p_i = 1$. Therefore, the WTPR θ characterizing each scenario is $\theta = \sigma\zeta / 1 = \sigma\zeta$.

We evaluated our approach in a range of possible scenarios, where we upper-bound the NSR lifetime by $\zeta = 10$ slots and we vary the $\omega \in \{0.05, 0.1, \ldots, 1\}$. For each scenario, we assess the capacity of our proposed frameworks to adapt its behavior to values of WTPR θ that vary over time. In particular, we divide the total number of simulated time slots into 10 equally-sized periods, each indexed by κ, and assign a constant θ_κ as WTPR for the duration of the period. This is done to ensure that the drift (or change) detection mechanism implemented in DARIO is able to detect abrupt changes in the distribution of θ at different points of the simulations, based on the recent observations of NSR features within a time slot. The values of θ_κ are sampled from a uniform distribution so that $\theta_\kappa \sim \mathcal{U}(\{10, 20, \ldots, 100\}), \quad \forall \kappa \in \{1, \ldots, 10\}$. Therefore, for each period κ, the related σ_κ is constant for the whole period and is $\sigma_\kappa = \theta_k/\zeta \in \{1, 2 \ldots, 10\}$. We generate the synthetic dataset in this way in order to limit the complexity of the considered dynamics and to ensure that DARIO can accurately detect distributional changes. However, based on the error threshold α selected, our framework would also be able to detect more gradual changes that could occur within a given time slot.

Benchmark Algorithms

We compare DARIO's performance to both static and adaptive selection policies. Namely, we compare our solution to the ExpRP, LinRP, and FCFS benchmark policies, as well as a UCB selection policy.

- **Static (or Expert) Policies**: LinRP and ExpRP assume full knowledge of θ. The FCFS policy is a greedy policy that does not consider the value of NSRs and admits them providing they pass a feasibility check.
- **Upper Confidence Bound**: The vanilla UCB is an adaptive selection policy that aims to maximize the long-term reward by choosing the action at each round with the highest upper bound.

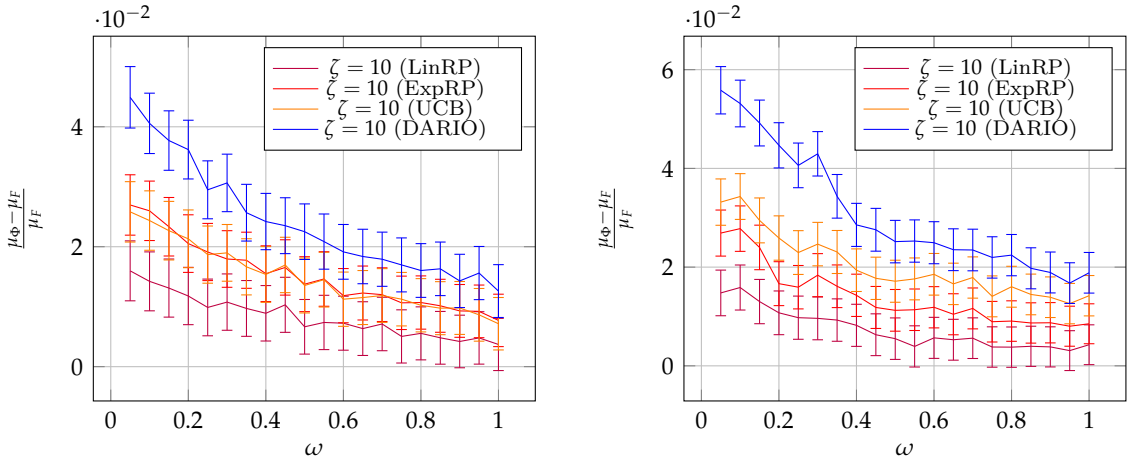
Performance Metrics

We evaluate our approach using:

- **Average Revenue Relative Gain**. The *average revenue* is the ratio $\mu = \frac{1}{|\mathcal{H}|} \sum_{h \in \mathcal{H}} \nu_h$ between the total revenue and the number $|\mathcal{H}|$ of all received slice requests. The *average revenue relative gain* for a policy is defined as the difference in the average revenue of the selection policy Φ and the FCFS policy, divided by the average revenue of the FCFS policy and is represented as $\frac{\mu_\Phi - \mu_F}{\mu_F}$, where μ_Φ and μ_F indicate the average revenues for a selection policy Φ and the FCFS policy, respectively.
- **Acceptance Ratio Relative Gain**. The *acceptance ratio* $\eta = n/|\mathcal{H}|$ is the ratio between the number n of accepted slices and the number $|\mathcal{H}|$ of all received slice requests. The *acceptance ratio relative gain* for a policy is defined as the difference in the average AR of the selection policy Φ and the FCFS policy, divided by the average AR of the FCFS policy and is represented as $\frac{\eta_\Phi - \eta_F}{\eta_F}$, where η_Φ and η_F indicate the acceptance ratios for selection policy Φ, and the FCFS policy, respectively.

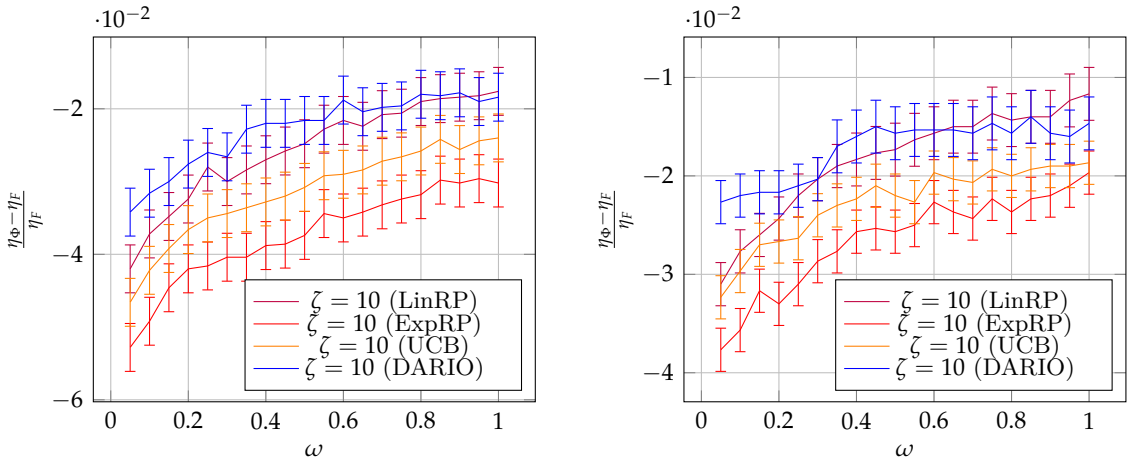
- **Average Resource Utilization Relative Gain**. The *average resource utilization* $\rho = \frac{1}{|\mathcal{H}|}\sum_{h\in\mathcal{H}}\sum_{j\in[m]} u_{h,j}/C_j$ as the sum of the normalized utilization for all resources for all received slice requests divided by the number $|\mathcal{H}|$ of all received slice requests. The *average resource utilization relative gain* for a policy is defined as the difference in the normalized utilization for all resources as a result of the selection policy Φ and the FCFS policy, divided by the normalized utilization for all resources with the FCFS policy and is represented as $\frac{\rho_\Phi - \rho_F}{\rho_F}$, where ρ_Φ and ρ_F indicate the average resource utilization for a policy Φ and the FCFS policy, respectively.

4.4.2 Simulation Results



(A) Average Revenue Relative Gain ($\lambda = 4$) (B) Average Revenue Relative Gain ($\lambda = 10$)

FIGURE 4.3: Average Revenue Relative Gain for DARIO against the Static and Adaptive baselines.

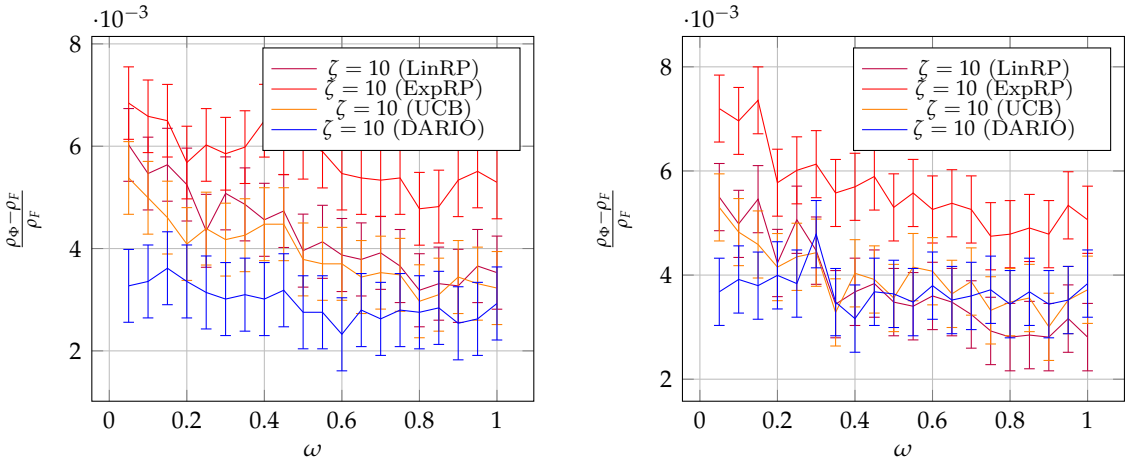


(A) Acceptance Ratio Relative Gain ($\lambda = 4$) (B) Acceptance Ratio Relative Gain ($\lambda = 10$)

FIGURE 4.4: Acceptance Ration Relative Gain for DARIO against the Static and Adaptive baselines.

Average Revenue Relative Gain

Figure 4.3a and Figure 4.3b show the relative gains of the LinRP (red, blue and green lines) and ExpRP (light red, light blue, and light green) policies, compared to the greedy (FCFS)



(A) Average Resource Utilization Relative Gain ($\lambda = 4$) (B) Average Resource Utilization Relative Gain ($\lambda = 10$)

FIGURE 4.5: Average Resource Utilization Relative Gain for DARIO against the Static and Adaptive baselines.

policy for the mean revenue. The results are based on the slice holding time δ_h of each incoming NSRs, which is bounded by a specific duration, i.e., $\zeta \in \{10, 30, 100\}$. From this, we can see that applying both policies in our algorithm leads to higher average revenue than the greedy policy across various upper bounds on the NSR durations and for the range of unit values considered. Specifically, it can be observed that in situations with higher economic inequality between tenants ($\omega \to 0$), the revenue gain is the highest (over 10%) when using both policies. This is because, in such scenarios, the value of the accepted NSRs is higher due to the online reservation function that places a higher threshold on the values that can be accepted.

Acceptance Ratio Relative Gain

The relative gains in the AR by using the online reservation-based policies, are shown in Figure 4.4a and Figure 4.4b. We see that by utilizing the introduced policies in our approach, fewer NSRs are admitted compared to the greedy FCFS approach for the considered range of upper-bounded NSR durations times. The performance of our solution improves when the average duration of requests increases and when the value of ω approaches 1. This is primarily due to the admission criteria of the reservation-based policies compared to that of the greedy policy, where the proposed solution rejects lower-valued requests at a higher rate compared to higher-valued slice requests with longer durations. This leads to a lower overall acceptance ratio for the considered parameters, but a higher revenue, as seen in the previous results for *Average Revenue Relative Gain*.

Average Resource Utilization Relative Gain

In Figure 4.5a and Figure 4.5b, we see that across most of the evaluated upper-bounded duration and unit price parameters, the resource utilization is lower on average using our approach. This is in line with the results from the acceptance ratio gain, which shows that based on the reservation-based policies our approach leads to a lower overall acceptance ratio, and hence, there are periods during the experiments when the resources are not fully utilized

but new NSRs are rejected due to their revenue and the current utilization level of any given resource in the system. Achieving a lower average utilization is key to achieving the goal of maximizing the revenue for InPs as it ensures that the scarce resources are reserved for the NSRs that offer the most revenue.

4.5 Chapter Conclusions

In this chapter, we focused on the problem of learning to select the optimal admission control policy among a set of policies, in slice-enabled mobile networks. We address the problem by modeling it as an Multi-Armed Bandit problem in which the selected policy is used to decide whether on not sequentially arriving NSRs will be admitted onto the network infrastructure. This approach overcomes the limitations of designing or learning a single one-size-fits-all policy for different network conditions and enables a dynamic and adaptive solution for learning a *meta-policy* for slice admission control. This is particularly important in dynamic 5G environments where the traffic patterns can change significantly over time.

To address the formulated MAB problem, we proposed DARIO, a Drift-Aware OL-based framework for the adaptive selection of admission control policies for network slicing. We consider two online policies as the possible AC policies and learn their performance in dynamic network scenarios. To detect when to switch between policies, we incorporate a CD mechanism that detects changes (or drifts) in the distribution of the underlying network request patterns, which triggers an update in the thresholds of the admission policies. We show that by detecting changes in the underlying features in the sequence of arriving NSRs, DARIO outperforms static policies and an adaptive selection policy that doesn't consider updating thresholds based on the changing distribution of the features of NSRs. Through simulations, we evaluate DARIO in a range of scenarios and show the benefit of extending the SW-UCB algorithm with a Concept Drift detection mechanism, as we achieve a higher average revenue gain (approximately 4.5%) compared to the static policies and the UCB policy, while our solution also leads to accepting marginally fewer NSRs compared to a naive FCFS policy.

In order to provision NSLs, the process of deciding whether to admit NSRs onto the mobile network infrastructure is only the first step, especially given the distributed infrastructure of modern edge networks that form part of the mobile network architecture. During the commissioning phase of the NSL lifecycle (Section 2.1.3), the decision of where to place the request on the underlying network infrastructure to allocate and configure it's resources is a crucial step. In Chapter 5, we design a scalable hierarchical framework to learn a placement policy in distributed edge networks in an online manner to minimize the provisioning time for new NSRs.

Chapter 5

Hierarchical Network Slice Provisioning

5.1 Introduction

In next-generation mobile networks, network operators will be expected to meet complex, and increasingly heterogeneous service requirements [207]. As a result, technology enablers such as SDN, NFV and edge computing are playing an increasingly crucial role in the Management & Orchestration (MANO) of resource in 5G mobile networks [208]. These techniques are essentially redefining how applications and services, such as VR and Internet-of-Things (IoT) [209], can be effectively provisioned to meet different performance requirements. The combination of such techniques aids the goal of concurrently provisioning and multiplexing a diverse set of services over a shared communication infrastructure through NS, which is a novel virtualized infrastructure model [210] in 5G networks. The increasing adoption of NS in NGMNs enables flexible network deployment strategies without requiring significant changes to the underlying components of the network infrastructure [126]. The NS model enables the provisioning of NSLs as part of a Slice-as-a-Service (SlaaS) solution from InPs, in which STs can request and ensure the reservation of network resources for a specified period of time [194]. In this context, a NSL can be described as a virtualized logical network that runs on top of a shared physical network that spans across multiple network domains, such as the RAN, the Transport Network (TN), the CN, and Edge Networks (ENs), to provide customized services [211]. To support the diverse service requirements expected in NGMNs and to deal with the time-varying nature of such networks presents a significant challenge for the efficient management and orchestration of network resources. A key challenge in provisioning NSLs is coordinating how to distribute and use the shared network resources while ensuring that each NSL adheres to its own set of rules and resource limits [212]. This challenge is further complicated by rapidly increasing network sizes and continually changing network conditions and environments. This motivates the development of novel solutions that can be used to address the NSP problem in NGMNs, where NSP refers to the process of allocating physical network resources to NSRs [213].

To address the problem above, we specifically focus on the placement of chained VNFs (i.e., SFCs) in virtualization-enabled edge networks. To do this, we assume that a NSR is deployed on

a per-service basis, i.e., one service, represented by an SFC, is mapped to one slice instance [214]. While finding an optimal solution to the NSP problem is already an NP-hard problem for static network conditions, its complexity is exacerbated in the online setting due to the presence of heterogeneous edge computing capabilities and multi-dimensional service requirements [215], [216]. As a result, traditional optimization and heuristic approaches are unlikely to provide satisfactory solutions within an adequate time to meet the low-latency requirements of network requests and services. To address this problem, we formulate the NSP problem as a two-step problem. In the first step, we aim to solve a Contextual Multiple-Objective Multi-Armed Bandit (MO-MAB) problem in order to select where in the network the arriving NSRs will be placed, based primarily on the requirements of the request and the capabilities of different parts of the network. In the second step, based on the solution provided in the first step, we seek to address a Combinatorial MO-MAB to jointly deploy the VNFs of a request across the nodes in the network. In the Contextual MO-MAB problem, we seek to learn a high-level action selection policy, which maps the current context information to the performance of the arms in meeting the different objectives. In the Combinatorial MO-MAB problem, the goal is to learn a policy for the joint deployment of VNFs in an SFC to nodes in the network. Based on this formulation, we propose a hierarchical framework to learn a provisioning policy that optimizes multiple provisioning objectives, simultaneously. Typically, the NSP problem has different optimization objectives, including maximizing resource efficiency, network latency minimization, and throughput maximization (i.e., QoS optimization) [162], [216]. In this context, the goal of addressing multiple objectives simultaneously and in an online manner can be a complex task, as a potential solution that maximizes one objective could adversely affect another. Therefore, finding the optimal trade-off that balances potentially conflicting objectives can be a challenging task [217].

The contributions of this chapter aim to answer the following research questions described in Section 1.3.3.

RQ 3.1: How can we increase the rate of admitted NSLs while reducing the required allocated resources, especially in **large-scale** networks with highly **dynamic** resource and user requirements?

RQ 3.2: How can we design a **hierarchical architecture** so that agents in different network subdomains make their own placement decisions?

RQ 3.3: How can we design a NS-provisioning performance metric that considers **multiple objectives** and enables a network policymaker to specify the objectives' relative **importance** and mutual **fairness**?

To address the challenges mentioned above, we propose HELIOS, which is a hierarchical multi-armed bandit solution to learn a NSL placement policy in distributed edge networks. We address ***RQ 3.1*** by demonstrating how dividing the network into clusters using community detection (Louvain algorithm) and implementing a two-tier agent structure provides an effective NSL provisioning solution. Our experimental results show that the hierarchical approach outperforms centralized baselines across multiple network topologies with heterogeneous

resources. The ***RQ 3.2*** is addressed by implementing the Generalized Gini Function (GGF) as a fairness-preserving aggregation function and integrating it within the HDF framework. Our results show that this approach successfully balances the dual objectives of maximizing acceptance ratio while minimizing node resource utilization within the network. We address ***RQ 3.3*** by showing that the utilization and combination of UCB-based algorithms at both the upper and lower levels of the hierarchy consistently outperform centralized baseline approaches in the considered network scenarios. This is validated by our time-slotted simulations, which demonstrate the ability of our HDF to make sequential provisioning decisions that improve over time as the system learns from previous slice deployments.

Our contributions are summarized as follows.

- We formulate the NSP problem as an general offline constrained optimization problem, considering computing and network resources as constraints. Then, we reformulate NSP problem as a HMAB problem to allow an online solution.
- We propose HELIOS, a novel two-level hierarchical learning system to solve the online NSP problem by jointly placing the VNFs of a SFC in the network. At HELIOS' high level, a contextual bandit agent directs each slice request to a specific region in the network, depending on the measured resource state and slice features. At HELIOS' low level, a combinatorial bandit agent determines the nodes on which the VNFs will be placed. HELIOS is designed to learn a placement policy that scales with network size.
- We leverage the GGI aggregation function which scalarizes and balances multiple provisioning objectives, together. By maximizing this function, we aim to find a point on the Pareto front of the multi-objective optimization problem. This allows for a direct optimization of the different provisioning objectives, enabling trade-offs based on the chosen weights.

The following sections discuss the content in our paper published in the IEEE 50th Conference on Local Computer Networks (LCN) [171]. Section 5.1 describes the research questions addressed in this chapter and discusses the contributions of the proposed learning-based framework. Section 5.2 describes the system model and formulates the Online Network Slice Provisioning (ONSP) problem. Section 5.3 presents the HELIOS framework in more detail. Section 5.4 describes the simulation setup and discusses the evaluation results. Finally, section 5.5 concludes the study in this chapter by reviewing the main contributions and highlighting the important results.

5.2 System Model and Problem Formulation

5.2.1 System Model

Network Model

We consider a distributed edge-computing network primarily comprising of edge servers and physical paths that are required to support the provisioning of diverse NSRs [215], see Fig 6.1.

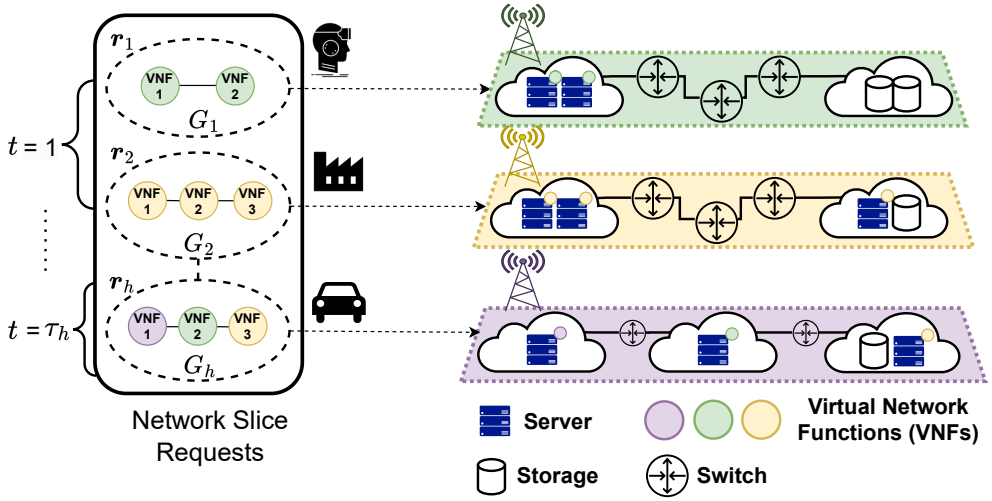


FIGURE 5.1: System Model

Each edge server has an arbitrary but limited amount of compute and network resources. Such a network can be modeled as a connected, undirected graph $G = (N, E)$, where N represents the set of edge servers available in the network and E represents the set of physical links in the network. In the considered network scenario, we assume the availability of node-specific resources such as CPU (MIPS), GPU (GFLOPS), RAM (MB), and Storage (MB), as well as link-specific resources like Bandwidth (Mbit/s) for communication between servers. We define C_j^n as the maximum capacity of resource type j on a network node $n \in N$, and C_j^q as the maximum capacity of resource type j on a network link $q \in E$. To avoid ambiguity when aggregating these values, we define the total capacity for each category separately. The aggregated capacity for the j-th type of node resource across all nodes is given by $C_j^{\text{node}} = \sum_{n \in N} C_j^n$. Similarly, the aggregated capacity for the j-th type of link resource across all links is given by $C_j^{\text{link}} = \sum_{q \in E} C_j^q$. Based on this infrastructure model, we assume that the InP owns and leases the physical network resources in an elastic, pay-as-you-go model by dynamically allocating and de-allocating them to incoming and existing NSR. In line with the time-varying conditions at the network edge, we model the time in the system as divided into discrete consecutive intervals, i.e., *time slots* $t \in T$.

Request Model

In our request model, an NSR is identified by the incremental index $h \in \mathcal{H}$, where $\mathcal{H} = \{1, \ldots, |\mathcal{H}|\}$ is defined as the ordered set of all request indices. We define $\boldsymbol{r}_h = (r_h^1, \ldots, r_h^m, \delta_h)$ as the vector containing the required amount of each resource request, as well as the *lifetime* of the request, $\delta_h \in \mathbb{N}$, which is defined as the number of time slots the request needs to access the network resources. We assume that slice tenants or service providers submit requests for NSLs, where each request defines the resources required by the NSL and the duration of the request. Based on the resource requirements of request h defined by $\boldsymbol{r}_h$, the tenant defines a suitable set of VNFs to provide the network slice request with a sufficient performance level to support the associated application. This set of VNFs, connected together through virtual links,

form a Service Function Chain (SFC) which would need to be deployed over the edge network to provision a NSL for services with specific performance requirements, as seen in Figure 6.1.

We indicate the single SFC associated with NSR h as $\mathcal{G}_h = (N_h, E_h, \delta_h, \tau_h)$, where N_h represents the set of all VNFs in SFC G_h, the quantity E_h represents the set of links concatenating the VNFs in the request, and $\tau_h \in \mathbb{N}$ represents the *timestamp* of the request, defined as the time slot index at which the request arrives in the network.

We denote the multi-dimensional resource requirements of a VNF v in a given SFC as $\boldsymbol{\varphi}_v = (\varphi_v^1, \ldots, \varphi_v^{m_{\text{node}}})$, with $r_h^j = \sum_{v \in N_h} \varphi_v^j$, where m_{node} is the number of node resource types. We denote the multi-dimensional resource requirements of a virtual link $e = (v, v') \in E_h \subseteq N_h^2$ between a pair of VNFs v and v' in a given SFC h as $\boldsymbol{\varphi}_e = (\varphi_e^1, \ldots, \varphi_e^{m_{\text{link}}})$, with $r_h^j = \sum_{e \in E_h} \varphi_e^j$, where m_{link} is the number of link resource types.

5.2.2 Problem Formulation

Given an arriving SFC, the objective in the NSP problem is to determine the optimal placement for each VNF in the SFC, while achieving the objectives of minimizing resource utilization and maximizing the number of accepted requests. More specifically, given an SFC G_h, our goal is to find an optimal mapping between each VNF v in N_h to an edge server $n \in N$ in the network topology.

Constraints

To formulate this problem, we introduce two binary variables x_v^n and y_e^q to (6.3a) indicate a mapping of a VNF v to an edge node n

$$x_v^n = \begin{cases} 1, & \text{if VNF } v \text{ is placed on edge node } n \\ 0, & \text{otherwise} \end{cases} \tag{5.1}$$

and (5.2), to indicate whether or not a virtual link between two VNFs $e \in E_h$, is mapped to a corresponding physical link in the edge network $q \in E$

$$y_e^q = \begin{cases} 1, & \text{if virtual link } e \text{ is mapped to physical link } q \\ 0, & \text{otherwise.} \end{cases} \tag{5.2}$$

Let us recall the indicator function $\mathbf{1}_{[p]}$, which is equal to 1 if predicate p is true and 0 otherwise. We now introduce two families of constraints (5.3) and (5.4) to ensure that the nodes' and links' capacities, respectively, are sufficient to support the NSR resources. It is worth noting that (5.3) and (5.4) jointly introduce a total of $2mT|N|$ constraints.

$$\sum_{h \in \mathcal{H}} r_h^j x_v^n \mathbf{1}_{[\tau_h \leq t < \tau_h + \delta_h]} \leq C_j^n, \qquad \forall n \in N, \forall v \in N_h, \\ \forall j \in \{1, \ldots, m_{\text{node}}\}, \forall t \in T \tag{5.3}$$

$$\sum_{h \in \mathcal{H}} r_h^j y_e^q \mathbf{1}_{[\tau_h \leq t < \tau_h + \delta_h]} \leq C_j^q, \qquad \forall q \in E, \forall e \in E_h, \\ \forall j \in \{m_{\text{node}} + 1, \ldots, m_{\text{node}} + m_{\text{link}} = m\}, \forall t \in T \tag{5.4}$$

where, m_{node} refers to the number of resources on a node, while m_{link} refers the number of physical link resources in the network. To ensure that each VNF v_h of an NSR is mapped to at most one node in the physical network graph G, we define the following constraint (5.5).

$$\sum_{v \in N_h} x_v^n = 1, \qquad \forall n \in N \tag{5.5}$$

Objectives

The NSP optimization problem aims to determine, at each time step, the optimal placement of an NSR that satisfies the resource constraints and minimizes a cost function U to achieve a tradeoff between the two following objectives.

- *Maximize Accepted NSRs*: This objective (5.6) aims to maximize the number of accepted requests in the network, which would maximize the revenue gained by the InP. Formally, we represent this objective as:

$$f_{\text{a}} = -\sum_{n \in N} \sum_{v \in N_h} x_v^n \tag{5.6}$$

- *Minimize Resource Utilization*: In the second objective, we aim to minimize the average resource utilization of edge nodes. We define the optimization objective for the j-th node resource as in (5.7).

$$f_j = \sum_{n \in N} \frac{1}{C_j^n} \cdot \sum_{h \in \mathcal{H}} \sum_{v \in N_h} r_h^j \cdot x_v^n, \tag{5.7}$$

  and the optimization objective for the j-th link resource as in (5.8).

$$f_j = \sum_{q \in E} \frac{1}{C_j^q} \cdot \sum_{h \in \mathcal{H}} \sum_{e \in E_h} r_h^j \cdot y_e^q \tag{5.8}$$

Multi-Objective Optimization: Let us define $\boldsymbol{w} = (w_{\text{a}}, w_1, \ldots, w_m) \in [0,1]^{m+1}$ as the weighting parameters for the different objectives, where $\|\boldsymbol{w}\| = 1$. We define the multi-objective utility vector as $\boldsymbol{f} = (f_{\text{a}}, f_1, \ldots, f_m)^\top$. We can combine the optimization objectives, while considering the constraints, to formulate a scalarized multi-objective problem as (9):

$$\min \quad U(\boldsymbol{f}) = \mathbf{w}^\top \boldsymbol{f} = w_a f_a + \sum_{j=1}^{m} w_j f_j \tag{5.9a}$$

$$\text{s.t.} \quad (1) - (5) \tag{5.9b}$$

Finding the optimal solution to such a combinatorial optimization problem is NP-hard [218], [219], as it would require knowing prior information about the resource requirements of NSRs and their performance objectives, which are not always available at runtime, and it could require exhaustive search to find an optimal solution. Furthermore, the overall space of placement options grows exponentially with the number of edge devices and the length of SFCs. Commercial solvers have previously been used to address similar optimization problems [220], however, they assume perfect system knowledge and could take considerable time to arrive to an optimal solution for the considered problem. Moreover, the difficultly of solving the considered optimization problem grows with the size of the physical network [221] and the number of optimization objectives [222]. Instead, we look to re-formulate the considered problem in the MAB setting, by converting the problem into a sequential decision problem, specifically the HMAB problem, that can be solved at each time slot following an online solution.

5.3 Hierarchical Placement Learning

This section describes HELIOS, our proposed data-driven NSP method that learns a multi-objective online provisioning policy. HELIOS is based on a two-level, hierarchical architecture (Figure 5.2). At the higher level, we have a centralized HLA that dynamically assigns arriving NSRs to one of the LLAs. Each LLA operates over a fixed-size subset of nodes and links (i.e., a cluster) in the network and effectively determines onto which nodes within the cluster the request's VNFs will be deployed. This design choice is driven by the observation that learning a provisioning policy in large-scale networks using a fully centralized approach may be impractical for learning algorithms due to the prohibitively large state and action spaces [223].

5.3.1 Background

To address the challenges outlined in Section 5.2.2, we decompose the network slice provisioning problem into K sub-problems, where each sub-problem $k \in \{1, 2, ..., |K|\}$ corresponds to a connected partition of the network graph $\mathcal{G}$ (See Figure 5.2). This decomposition leverages the natural community structure of physical networks [156], [221], reducing the original problem's complexity through lower-dimensional state and action spaces, which is particularly important for large-scale graphs.

We formulate this as a hierarchical optimization problem with two learning layers: (1) Specialized LLAs or *experts* that solve the placement problem in cluster-specific subdomains [224], [225], and (2) A coordinating HLA that selects among these experts to balance exploration-exploitation. The graph partition is generated using the Louvain method [226], which maximizes modularity to identify communities. Each resulting community, or cluster, forms a discrete spatial region for resource allocation [227].

In our hierarchical framework, the HLA directs slice requests to these regions, optimizing the trade-off between exploration (testing underutilized clusters) and exploitation (leveraging high-reward clusters) to maximize the multi-objective performance. Essentially, the HLA

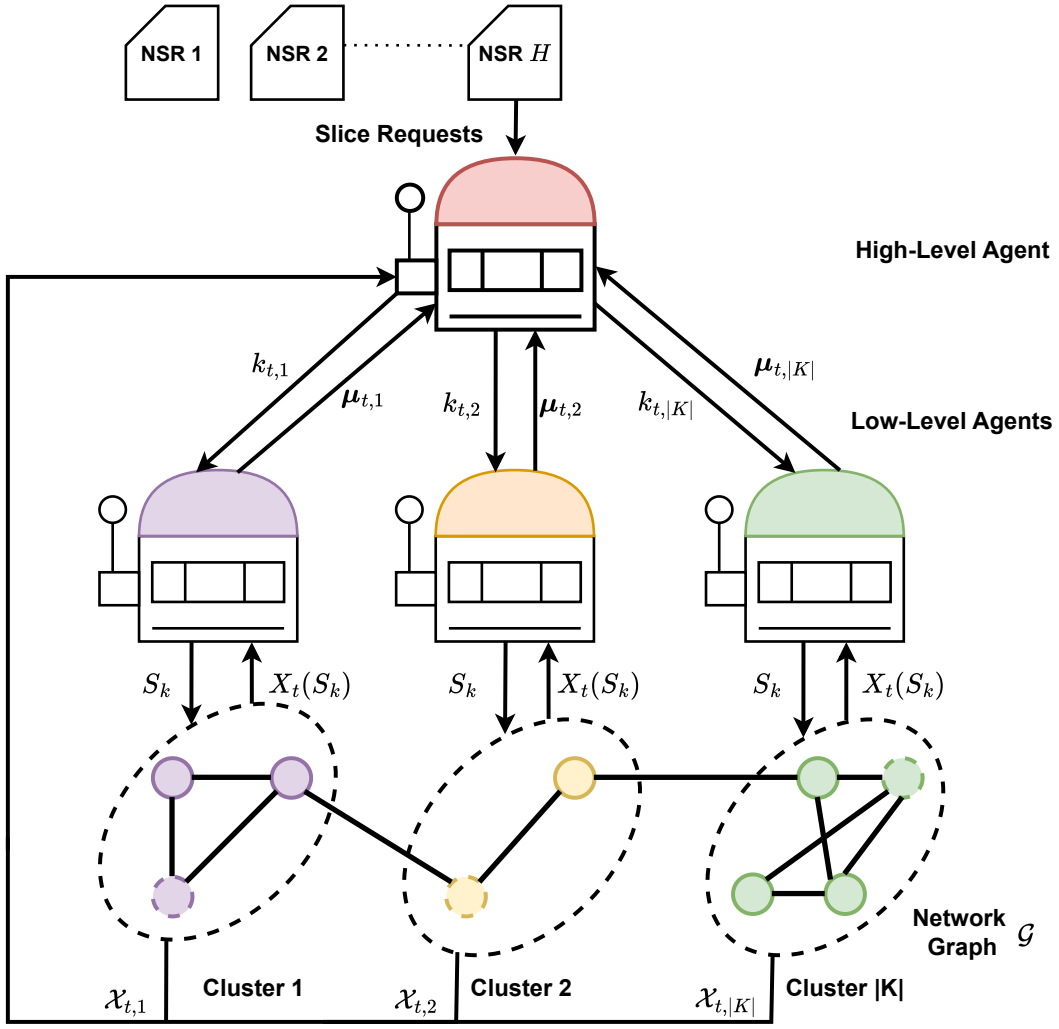


FIGURE 5.2: HELIOS Architecture.

treats each LLA as an *expert* whose performance in a given region or sub-domain is learned and evaluated over time. This selection is performed using the Multi-Objective Contextual Multi-Armed Bandit (MO-CMAB) framework [228], which balances the trade-off between exploration and exploitation. In the second layer of the framework, the LLA associated with the selected cluster is responsible for placing the components of each NSR on nodes within its assigned sub-domain. The LLA's goal is to jointly select node combinations that optimize the provisioning objectives defined in Equation 5.9a. This combinatorial selection process is approximated using the Combinatorial Multi-Objective Multi-Armed Bandit (COMO-MAB) framework [229], which efficiently handles the exploration-exploitation trade-off across node subsets and enables each LLA to learn optimal node combinations for VNF placement within its cluster, while adapting to dynamic resource constraints.

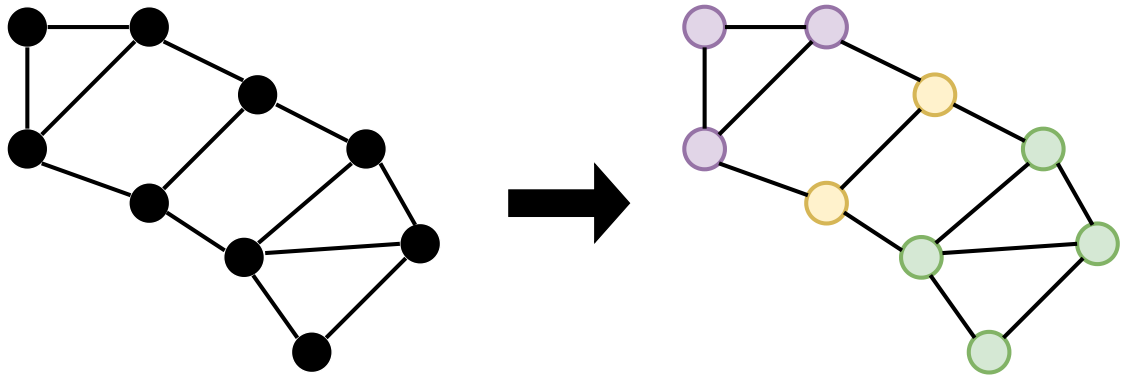


FIGURE 5.3: Example Clusters in an Abilene Topology.

5.3.2 Cluster Formation via Community Detection

Given a physical network $\mathcal{G} = (N, E)$ of unclustered nodes, our goal is to find groups, or clusters, of nodes by identifying the *communities* within the network, where a community is formed by optimizing the local modularity of each group of clustered nodes. Hence, we define a community as a group of nodes that are densely connected internally, while being sparsely connected to the nodes in other groups [230]. This enables the formation of different node clusters with heterogeneous resource capacities, where each cluster can either belong to the one or multiple InPs. To effectively optimize the modularity of the network graph G, in this work, we adopt the Louvain algorithm [226], which efficiently identifies the communities (i.e., clusters) in the network, and we use the nodes in the communities as a basis for our hierarchical bandit approach. The Louvain algorithm is a hierarchical community detection method that optimizes modularity through an iterative two-phase approach to identify cohesive subgroups within network structures. The algorithm commences by assigning each node to its own distinct community and progressively optimizes the modularity score Q, which quantifies the density of connections within communities relative to connections between communities. In each iteration, the algorithm evaluates potential node movements between communities by calculating the modularity gain ΔQ that would result from relocating a node from its current community to a neighbor's community. The modularity gain calculation considers both the density of internal connections within potential new communities and the reduction in connections to the node's original community. Nodes are reassigned to neighboring communities only when such movements yield positive modularity gains, ensuring monotonic improvement in the network's community structure. This process continues until no further improvements in modularity can be achieved, indicating a locally optimal partition of the network into communities. The algorithm's effectiveness stems from its ability to naturally determine the number of communities without requiring this as a predetermined parameter, while its modularity optimization approach inherently balances the competing objectives of maximizing intra-community connections while minimizing inter-community connections. While we leverage the Louvain algorithm for detecting clusters in our approach, other clustering techniques (i.e., Leiden, Birch, DBSCAN etc.) can be used to detect the clusters in a network graph. As an examples, we show the possible clusters formed on a simple

Abilene topology, in Figure 5.3.

5.3.3 Hierarchical Multi-Armed Bandit Framework

To formulate the problem, we consider that the set of nodes N in $\mathcal{G}$ are partitioned into $|K|$ disjoint clusters, where $k \in \{1, 2, ..., |K|\}$. As depicted in Figure 5.2, the HMAB framework consists of two parts: (1) High-level policy π^h learning at the HLA, based on the principle of contextual bandits, to orchestrate a set of high-level actions $\{a_k^h\}_{k=1}^K$, where each high-level action corresponds to selecting one of the available clusters k; (2) low-level policy π_k^l learning at the LLAs, based on the principle of combinatorial bandits, where each LLA assigned to a cluster seeks to learn a joint node selection policy that optimizes the provisioning objectives.

We consider a sequential decision-making problem over a time horizon T, where at each time step $t \in \{1, 2, \cdots, T\}$, the HLA observes a context $\boldsymbol{\mathcal{X}}_t \in \boldsymbol{\mathcal{X}} \subseteq [0,1]^{M|K|}$, with $||\boldsymbol{\mathcal{X}}_t||_2 \leq 1$ for all t. The context vector represents the current state of the network, including the available resources in each cluster and the aggregated resource requirements of the arriving requests. Specifically, we define the context $\mathcal{X}_{t,k} \in \mathbb{R}^M$ of a cluster k at time slot t as a vector containing M relevant system information elements, such as available multi-dimensional resources, requested resources by the NSR, and the number of nodes in cluster k. The overall context vector at time t is therefore represented by $\boldsymbol{\mathcal{X}}_t = (\mathcal{X}_{t,1}, \cdots, \mathcal{X}_{t,|K|}) \in [0,1]^{M|K|}$ with $\mathcal{X}_{t,k}$ associated with each cluster $k \in K$.

Based on the observed context at time step t, the HLA takes an action $a_k^h \in \boldsymbol{\mathcal{A}}^h$, where $\boldsymbol{\mathcal{A}}^h = \{a_k^h | k \in \{1, \cdots, |K|\}\}$, and receives a corresponding multi-objective reward vector $\boldsymbol{\mu}_{t,k}$, where $\boldsymbol{\mu}_{t,k} \in [0,1]^{m+1}$. The selected high-level action a_k^h directs arriving slice requests to a specific cluster, k. Then, the LLA for that cluster selects an action $a_S^l \in \boldsymbol{\mathcal{A}}_k^l$, which specifies the subset of nodes (S_k) on which to place the requests' VNFs, where $\boldsymbol{\mathcal{A}}_k^l = \{S_k \subseteq K_k | |S_k| \leq D\}$, of which K_k is the set of network nodes belonging to cluster k and D represents the number of VNFs to be placed.

Given the contextual bandit approach used by the HLA, the expected reward of selecting an action $a_{t,k}^h$ given the context $\mathcal{X}_{t,k}$ follows the linear realizability assumption and is given by: $\mathbb{E}[\boldsymbol{\mu}_{t,k} | \mathcal{X}_{t,k}, a_{t,k}^h] = \mathcal{X}_{t,k}^\top \theta_k^*$, where, we assume that there exists K unknown parameter vectors $\theta_1^*, \theta_2^*, \cdots, \theta_K^*$ for each high-level action, and where $\forall k : \theta_k^* \in \mathbb{R}^{(m+1)\times M}, ||\theta_k^*||_2 \leq 1$. For simplicity, and without loss of generality, we adopt a shared-parameter formulation in which all arms (*i.e.,* sub-domains) share the same underlying unknown parameter vector, *i.e.,*, $\theta_1^* = \theta_2^* = \cdots = \theta_K^* = \theta^*$, which avoids learning disjoint arm-specific models while allowing for generalization across the different sub-domains. Based on this assumption, we define the regret of the high-level policy π^h as

$$R_\pi(T) = \mathbb{E}\left[\sum_{t \in T} G(\boldsymbol{\mu}_{t,k^*}) - \sum_{t \in T} G(\boldsymbol{\mu}_{t,k})\right] \tag{5.10}$$

where $G(\cdot)$ is an aggregation function that maps the multi-dimensional reward vector $\boldsymbol{\mu}$ to a scalar value based on the weight vector $\mathbf{w}$ (Eq. (9)), k^* is the index of the best high-level action

at time step t and k indicates the index of the high-level action selected by the HLA. In the formulated HMAB problem, the goal is to learn a cluster selection policy at the HLA and a combinatorial node selection policy at the LLA.

High-Level Agent (HLA)

The HLA's goal is to leverage current context information to learn a multi-objective cluster selection policy that optimizes the different objectives in Equation 5.9a, and effectively acts as a *slice broker* [114], [231] that decides, at a high-level, the allocation of resources for NSRs in the network.

Let us define the known mean reward vectors of each cluster $k \in \{1, \ldots, |K|\}$ as $\boldsymbol{\mu}_k$. We consider that for a given aggregation function $G(\boldsymbol{\mu}_k)$, the optimal cluster selection policy seeks to find a strategy such that the index of $G(\bar{\boldsymbol{\mu}}_k)$ is as large as possible, where $\bar{\boldsymbol{\mu}}_k = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{\mu}_{t,k}$. This ensures that clusters are selected with the aim of maximizing the aggregation function, towards the optimization of the provisioning objectives. In this work, we consider that the aggregation function that scalarizes inputs from different objectives, is the GGF [232]. The GGF is a non-linear, concave function, and is a special case of the Ordered Weighted Averaging (OWA) aggregation operators [233] that seek to preserve impartiality with respect to individual objectives. Rather than using a deterministic strategy that always selects the single cluster maximizing the aggregated reward G, we adopt a mixed strategy that optimizes a probability distribution $\boldsymbol{\alpha}_t \in \mathbb{A}$ over all clusters, from which a cluster is sampled. Specifically, the probability distribution is the simplex $\mathbb{A} = \{\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{|K|}) \in [0,1]^{|K|} : \|\boldsymbol{\alpha}\|_1 = 1\}$, according to which a cluster α_k is selected. As explained in [234], the optimal mixed strategy cluster selection policy is given by solving the following optimization problem, $\boldsymbol{\alpha}^* \in \underset{\boldsymbol{\alpha}\in\mathbb{A}}{\arg\max}\ G\left(\sum_{k=1}^{|K|} \alpha_k \boldsymbol{\mu}_k\right)$, which determines a probability distribution over clusters that maximizes the GGI of the expected aggregated rewards. However, based on the probability distribution used by the mixed strategy, clusters with smaller estimated rewards are periodically selected to balance the trade-off between exploration of potentially useful clusters that could lead to good multi-objective rewards and exploitation of cluster that are known to have high estimated rewards. By leveraging the GGF as the aggregation function in the optimal cluster selection strategy, the regret of the high-level policy can be estimated as:

$$\hat{R}_\pi(T) = G\left(\frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{K} \alpha^*_{t,k}\boldsymbol{\mu}_{t,k^*}\right) - G\left(\frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{K} \alpha_{t,k}\boldsymbol{\mu}_{t,k}\right) \quad (5.11)$$

Given this definition, the goal of the high-level policy is to minimize the regret by maximizing the GGF of the aggregated multi-objective reward (second term in Eq. (11)), based on the current parameter estimates of the cluster selection strategy. Specifically, we utilize ridge regression [235] to get the expected reward $\boldsymbol{\mu}_{t,k}$ for a particular action $a^h_{t,k}$ at time t, given the context $\mathcal{X}_{t,k}$. The parameter estimate for objective o is given by $\hat{\boldsymbol{\theta}}^o_t = (A_t + \gamma \mathbf{I}_M)^{-1}\mathbf{b}^o_t$. Here,

$A_t = \sum_{\tau=1}^{t-1}(\mathcal{X}_{\tau,k_\tau}\mathcal{X}_{\tau,k_\tau}^\top)$ accumulates the outer products of contexts from previously selected clusters, and $\mathbf{b}_t^o = \sum_{\tau=1}^{t-1}\mathcal{X}_{\tau,k_\tau}r_\tau^o$ accumulates the products of contexts with the observed rewards, where $r_\tau^o \in \mathbf{r}_\tau$ is the reward of objective o observed from the combinatorial action of the LLA in cluster k_τ at time τ and $\mathbf{r}_\tau = \boldsymbol{X}_\tau(S_{k_\tau})$.

The cluster selection policy α_t seeks to maximize the GGF of the expected aggregated multi-objective rewards through Online Gradient Ascent (OGA) by optimizing the function $G\left(\sum_{k=1}^{|K|}\alpha_{t,k}\hat{\boldsymbol{\mu}}_{t,k}\right)$. Starting from a uniform distribution $\alpha_t = (1/|K|,...,1/|K|)$, we perform Z gradient ascent steps, where at each step z, we perform the following:

- Compute the gradient: $\mathbf{g}^{(z)} = \nabla_{\boldsymbol{\mu}} G\left(\sum_{k=1}^{K}\alpha_k^{(z)}\hat{\boldsymbol{\mu}}_{t,k}\right)\cdot\hat{\mathbf{M}}_t$, where $\hat{\mathbf{M}}_t = [\hat{\boldsymbol{\mu}}_{t,1},\ldots,\hat{\boldsymbol{\mu}}_{t,|K|}]$ is the matrix of estimated mean reward vectors of each cluster k.
- Next, we take a gradient step: $\tilde{\boldsymbol{\alpha}}^{(z+1)} = \boldsymbol{\alpha}^{(z)} + \eta_z\mathbf{g}^{(z)}$, with step-size $\eta_z = \frac{1}{\sqrt{z+1}}$
- Finally, this is projected onto the simplex: $\boldsymbol{\alpha}^{(z+1)} = \Pi_{\mathbb{A}}\left(\tilde{\boldsymbol{\alpha}}^{(z+1)}\right)$.

After Z iterations, we use $\boldsymbol{\alpha}_t = \boldsymbol{\alpha}^{(Z)}$ as our mixed strategy and sample cluster k_t according to this distribution. This approach ensures convergence to a local optimum of the GGI while maintaining the constraint that $\boldsymbol{\alpha}_t$ remains a valid probability distribution [228].

Low-Level Agent (LLA)

Upon receiving an NSR $\mathcal{H}_t$ directed by the HLA to a cluster k_t, the designated LLA for that cluster is tasked with the goal step of selecting an optimal set of physical nodes to host the D VNFs comprising the SFC. This selection is formulated as a COMO-MAB problem [229], where the LLA must learn to make choices that align with the overall system objectives defined in Eq (9). To formulate the problem, we define K_k as the set of network nodes belonging to cluster k. Each network node in set K_k can be selected by the LLA for deploying a VNF, therefore each network node in K_k is a *base arm* of the agent. We define the *power set* $\mathcal{P}(K_k)$ of set K_k as the set of all possible combinations of base arms $\in K_K$, i.e., $\mathcal{P}(K_k) = \{S_k | S_k \subseteq K_k\}$. We now define a *super arm* S_k for cluster k as a subset of the set K_k of base arms, which represents the network nodes on which the SFC's VNFs will be simultaneously deployed, i.e., $S_k \in \mathcal{P}(K_k)$. It is worth noting that $|\mathcal{P}(K_k)| = 2^{|K_k|}$, which makes the problem NP-hard as it scales exponentially with the k-th cluster size $|K_k|$.

At each time step t, the LLA pulls the super arm S_t and receives a reward $\boldsymbol{X}_t(S_k) = (X_{t,1},\ldots,X_{t,|S_k|}) \in [0,1]^{|S_k|\times(m+1)}$, and the outcomes of the base arms in $\boldsymbol{X}_t$ are assumed to be independent. The rewards represent the acceptance of the VNFs in the selected, as well as the resource utilization on the selected nodes as a result of deploying the request. The final goal of the LLAs is to eventually learn the optimal super arm S_k^* (i.e., set of nodes in K_k) that optimizes the objectives defined in Eq (8). The goal of the LLA is to learn the optimal super arm $S_{kt}^* \in \mathcal{P}(K_k)$, over time. However, selecting the optimal super arm, which is given by:

$$S_{kt}^* = \underset{S\in\mathcal{P}(K_k)}{\arg\max}\|X_t(S)\| \tag{5.12}$$

is an NP-hard problem [236], [237]. Hence, the LLAs in the Hierarchical Bandit (HB) framework uses the UCB to learn the rewards of the different arms in a given super arm and picks the super arm with the higher upper confidence bound. More specifically, a simplified version of the Combinatorial Multi-Objective Upper-Confidence Bound (COMO-UCB) [229] algorithm is developed to address the exploration and exploitation dilemma of the considered problem.

5.3.4 Proposed Algorithm

Our proposed algorithm works as follows (Algorithm 6). First, in lines 1 and 2, we initialize the contextual and combinatorial bandit parameters A_t, $\mathbf{b}_t$, and UCB_i^o which represent the identity matrix, scaled by γ for ridge regression, the reward accumulation vector for each objective o, and the upper confidence bound of the reward for each base super arm, respectively. Then (line 4), we receive the $|K|$ clusters based on the communities detected on the network graph $\mathcal{G}$. In line 5, we initialize a (uniform) probability distribution over the clusters $\boldsymbol{\alpha_t}$ which determines the initial probability of selecting each cluster.. In lines 6 and 7, we begin by collecting the NSRs arriving at the current time slot. Based on the collected NSRs and the current resources in the clusters, the HLA estimates the reward of placing the requests in each cluster (lines 18-22). In lines 24-29, we update $\boldsymbol{\alpha_t}$ using projected gradient ascent to maximize the GGI-aggregated expected reward, and then sample a cluster according to the resulting distribution.

Based on the selected cluster in the HLA procedure, the LLA function determines the combination of nodes within the cluster to deploy the VNFs of a request. For each request in the current time slot $\mathcal{H}_t$, the LLA looks to improve its VNF placement strategy through exploration and exploitation. In lines 34-37, for each node i in the selected cluster K_k and each objective o, the LLA evaluates the potential rewards from selecting that node based on the empirical mean rewards from previous times a node was selected. From lines 38-40, the LLA, based on the COMO-UCB algorithm [229], selects the super arm S_t (i.e., set of nodes) that maximizes the sum of UCB values across all the selected nodes and objectives. If the VNFs of the request are deployed on the set of nodes, then the LLA receives a multi-dimensional reward $\boldsymbol{X_t}(S_k)$ that captures the provisioning objectives (i.e., the acceptance of the request and the utilization of resources in the cluster) (lines 41-45). Based on the selected nodes by the LLA, the shortest path between the nodes is computed using Dijkstra's algorithm. In lines 13-15, we update the parameters of the contextual bandit algorithm based on the multi-dimensional rewards of the LLA's selected nodes, in order to improve the overall learning procedure.

Algorithm 6 Hierarchical nEtwork sLIce prOviSioning (HELIOS)

Require: Regularization $\gamma > 0$, learning rate $\eta_z > 0$, number of gradient steps Z

Initialize: $A_t \leftarrow \gamma I_M$, $\mathbf{b}_t^{(o)} \leftarrow \mathbf{0}_M \ \forall o \in [m+1]$
$\text{UCB}_i^o(t) \leftarrow +\infty, \forall o \in [m+1], \forall i \in [K_k], \forall t \in [\mathcal{H}_t]$
▷ Create Set of Clusters based on $\mathcal{G}$
$|K| \leftarrow \text{Louvain}(\mathcal{G})$
$\boldsymbol{\alpha}_t \leftarrow (1/|K|, ..., 1/|K|)$
for $t \in \{1, \ldots, T\}$ **do**
▷ Collect NSRs arriving during timeslot t
  $\mathcal{H}_t \leftarrow \text{WaitNSRs}(t)$
  $k_t \leftarrow \text{HighLevelAgent}(\mathcal{H}_t)$
  $\mathbf{X}_t(S_{k_t}) \leftarrow \text{LowLevelAgent}(\mathcal{H}_t, k_t)$
  $\mathbf{r}_t = \mathbf{X}_t(S_{k_t}) \in \mathbb{R}^{m+1}$
▷ Regression Update
  $A_t \leftarrow A_t + \mathcal{X}_{t,k_t} \mathcal{X}_{t,k_t}^T$
  for $o \in [m+1]$ **do**
    $b_{t+1}^o \leftarrow b_t^o + \mathcal{X}_{t,k} r_t^o$

function HighLevelAgent($\mathcal{H}_t$)
▷ Observe Cluster Contexts at timeslot t
  $\boldsymbol{\mathcal{X}}_t \leftarrow \text{GetClusterContexts}(t)$
  for $o \in [m+1]$ **do**
    $\hat{\boldsymbol{\theta}}_t^o \leftarrow A_t^{-1} \mathbf{b}_t^o$ ▷ Estimate parameter
    for $k \in K$ **do**
      $\hat{\mu}_{t,k}^{(o)} \leftarrow \mathcal{X}_{t,k}^\top \hat{\boldsymbol{\theta}}_t^o$ ▷ Predict reward
▷ Perform Online Gradient Ascent to optimize mixed strategy
  $\boldsymbol{\alpha}^{(0)} \leftarrow \boldsymbol{\alpha}_t$ ▷ Initialize
  for $z \in [Z]$ **do**
    $\nabla_{\boldsymbol{\mu}} G \left(\sum_{k=1}^K \alpha_k^{(z-1)} \hat{\boldsymbol{\mu}}_{t,k} \right)$ ▷ Compute gradient
    $\boldsymbol{\alpha}^{(z)} \leftarrow \Pi_{\mathbb{A}} \left(\boldsymbol{\alpha}^{(z-1)} + \eta_z \cdot \nabla \right)$ ▷ Update
  $\boldsymbol{\alpha}_t \leftarrow \boldsymbol{\alpha}^{(Z)}$ ▷ Set final strategy
  $k_t \sim \text{Categorical}(\boldsymbol{\alpha}_t)$ ▷ Sample cluster
▷ Send request to LLA of cluster k_t
  return k_t

function LowLevelAgent($\mathcal{H}_t, k_t$)
  for $t \in \{1, \ldots, |\mathcal{H}_t|\}$ **do**
    for $i \in [K_k], o \in [m+1]$ **do**
▷ Update Upper Confidence Bound
      if $P_i > 0$ **then**
        $\text{UCB}_i^o(t) \leftarrow \bar{X}_i^o(t-1) + \sqrt{\frac{3 \log t}{2 P_i}}$
    $S_t \leftarrow \arg\max_{S \in \mathcal{S}} \sum_{i \in S} \sum_{o=1}^{m+1} \text{UCB}_i^o(t)$
▷ Play super arm S_t and observe rewards
    $\boldsymbol{X}_t(S_k) = (X_{t,1}, \ldots, X_{t,|S_t|}) \leftarrow \text{PullArm}(S_t)$
    for $i \in S_k$ **do**
▷ Increment pull counter for arm i
      $P_{i+1} \leftarrow P_i + 1$
      for $o \in [m+1]$ **do**
        $\bar{X}_i^o(t) \leftarrow \frac{P_i \bar{X}_i^o(t-1) + X_i^o(t)}{P_{i+1}}$
  return $\boldsymbol{X}_t(S_k)$

TABLE 5.1: Simulation Parameter Settings

Parameters	**Values**
Clusters in GEANT and DTelekom, $\|K\|$	$\{3, 5\}$
GGI Weights $G_{\mathbf{w}}$	$w_o = 2^{-o+1}$
OGA Iterations Z	10
Learning Rate η_z	$\frac{1}{\sqrt{z+1}}$
Regularization Parameter γ	0.1
VNF Resource Requirements	$[5, 50]$ units
Node Capacity φ_v^m	$[5, 1000]$ units
Virtual Link Requirements	$[50, 100]$Mbit/s
Link Capacity	$[500, 5000]$Mbit/s
SFC Length (VNFs per request) ψ	$\{2, 3, 4\}$

5.4 Performance Evaluation

5.4.1 Simulation Setup

We evaluated the performance of the HELIOS framework against five baselines on two real-world network topologies of different size, namely: GEANT (22 nodes, 33 links, which is a pan-European research and education data network) [238] and DTelekom - DT2 - (68 nodes, 272 links) which is a sample topology of Deutsche Telekom [239]. The simulations, proposed solution, and baseline algorithms were implemented in Python with NetworkX to simulate different network scenarios on the two network topologies. We select the Louvain community detection algorithm to identify the $|K|$ communities (or clusters) within the network topology based on the topological features of each generated network graph. One LLA agent is deployed within each cluster to handle the joint placement of VNFs on the devices within the cluster. The CPU, RAM, storage, and GPU capacities of network nodes φ_v^m, were uniformly generated within the range of $[5, 1000]$ resource units, while the bandwidth of each link is in the range $[50, 5000]$Mbit/s. For each NSR we randomly generate an SFC with $\{2, 3, 4\}$ VNFs, where each VNF represents a generic NF. The resource requirements of each VNF and virtual link in a request are uniformly generated within the range of $[5, 50]$ resource units and $[50, 100]$Mbit/s, respectively.

To conduct realistic evaluations, we evaluate the performance of our algorithm in an online setting with $T = 5000$ time slots. We assume a random number v_t of NSRs arriving in the system at the beginning of each time slot, which defines the time between successive requests. We model v_t as a Poisson process in which all $v_t, \forall t \in [T]$ follow a Poisson distribution, Pois(λ), with identical arrival rate λ, where λ is the average number of arrivals per slot. We evaluated scenarios, where $\lambda = 2$ and $\lambda = 5$. The lifetimes δ_h of the requests (in minutes) are generated from a uniform distribution $\delta_h \sim \mathcal{U}(\{10, \zeta\})$, where ζ is the upper bound of the request duration and we evaluated the scenario in which $\zeta = 50$ to investigate the impact of different request lifetimes on the system. Finally, we initialize the weight vector $\mathbf{w}$ to $w_o = 2^{-o+1}, o \in [m+1]$ [234]. Table 5.1 summarizes the simulation parameters used.

5.4.2 Benchmark Algorithms

We compare HELIOS against a set of benchmark algorithms that are adapted to address the considered slice provisioning problem:

- *Random*: The random provisioning policy uniformly and jointly assigns VNFs of an SFC to different nodes in the network without considering their available resources.
- *ϵ-greedy*: This provisioning policy uses the $\epsilon-$greedy strategy to determine where to deploy the VNFs of a request, based on exploration parameter, ϵ. We set $\epsilon = 0.5$ in our evaluations.
- *Linear Upper Confidence Bound (LinUCB)* [240]: LinUCB is a learning-based algorithm that leverages context information to determine where to place requests given the entire topology.
- *Contextual-Combinatorial Upper-Confidence Bound (C^2UCB)* [241]: The C^2UCB algorithm combines context information with combinatorial node selection to determine the nodes on which to place a request jointly.
- *Contextual Thompson Sampling (CTS)* [242]: CTS is a lightweight solutions that learns a provisioning policy based on estimating the posterior distribution of the expected rewards of different placements conditioned on the context information.

5.4.3 Performance Metrics

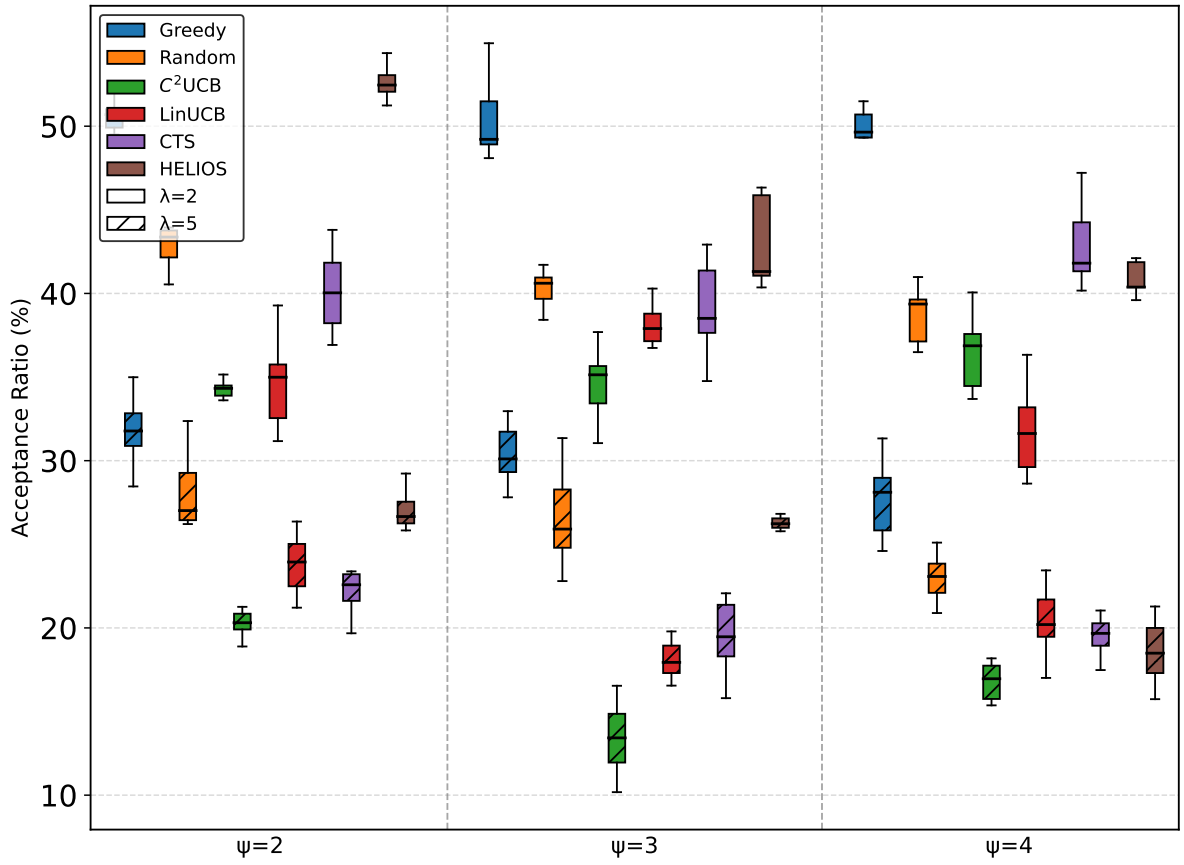
We evaluated the performance of our approach by considering the following three metrics:

- **Request Acceptance Ratio**: The request acceptance ratio is effectively defined as the ratio between the total number of NSRs submitted by the tenants and the total number of NSRs accepted onto the infrastructure.
- **Average Node Resource Utilization**: The average node resource utilization quantifies the mean percentage of computational, memory, or bandwidth resources used across all nodes in the network over a given period. It is calculated by averaging the utilization rates of individual nodes in the network.
- **Average Execution Time**: The average execution time represents the mean time required to process and complete NSL placement from the moment they are initiated until completion.

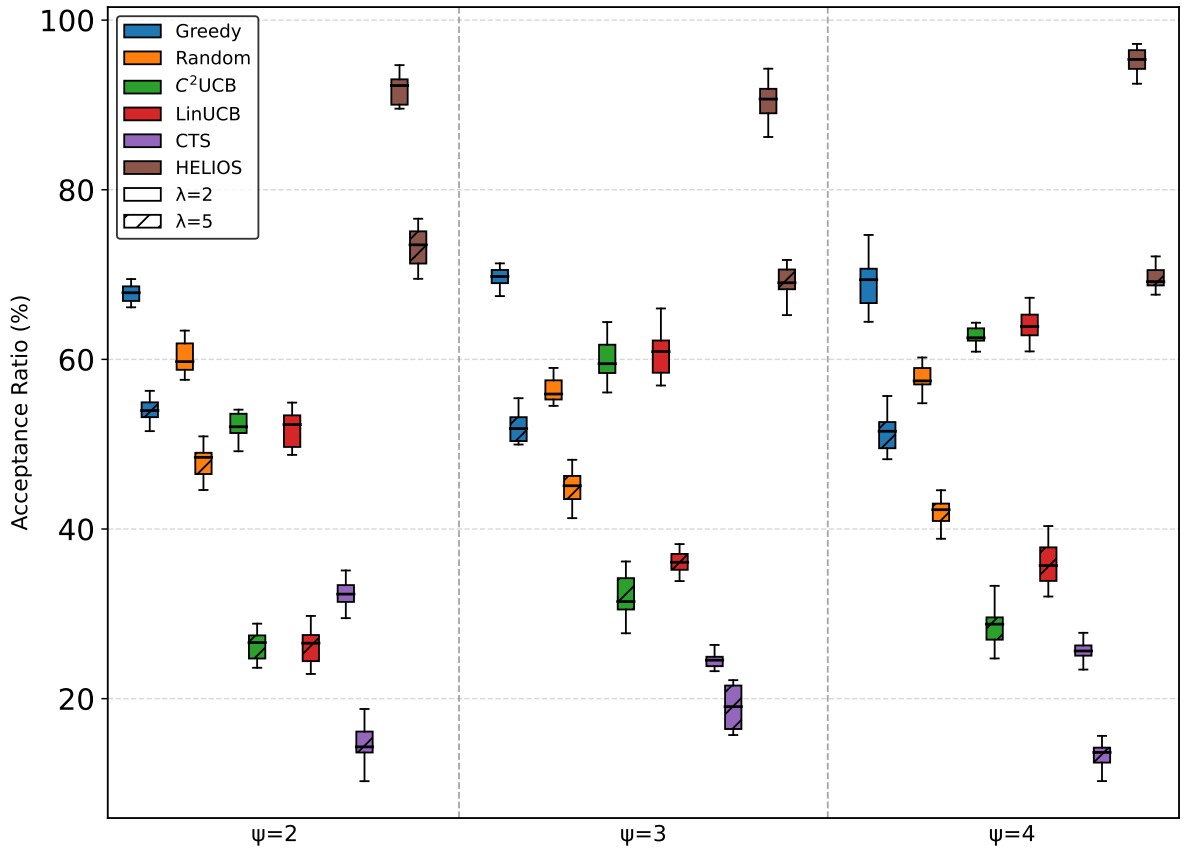
5.4.4 Simulation Results

Request Acceptance Ratio

Defined as the ratio between the total number of NSRs submitted by the tenants and the total number of NSRs accepted onto the infrastructure. Figure 5.4a and Figure 5.4b show the performance of our approach compared to the baseline algorithms on the GEANT and DT2 topologies, respectively. The results are obtained for the scenario where the parameters for

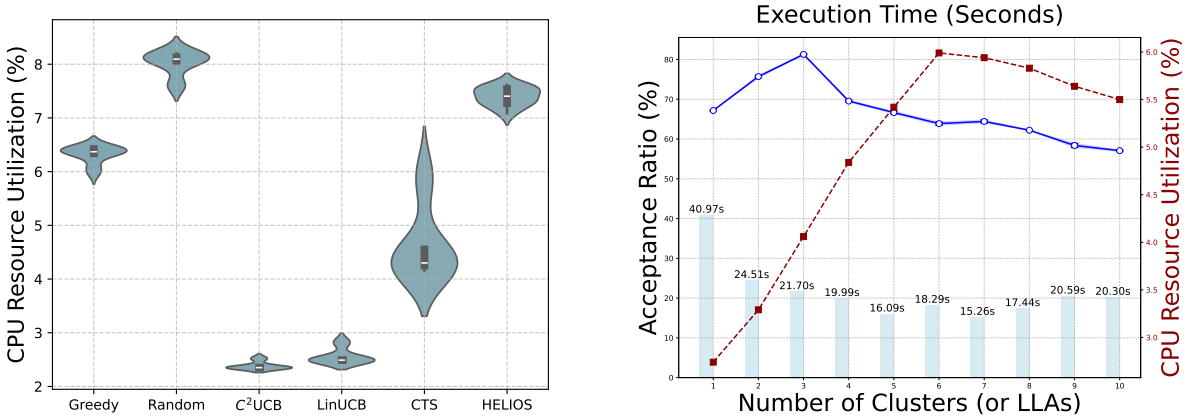


(A) Request Acceptance Ratio



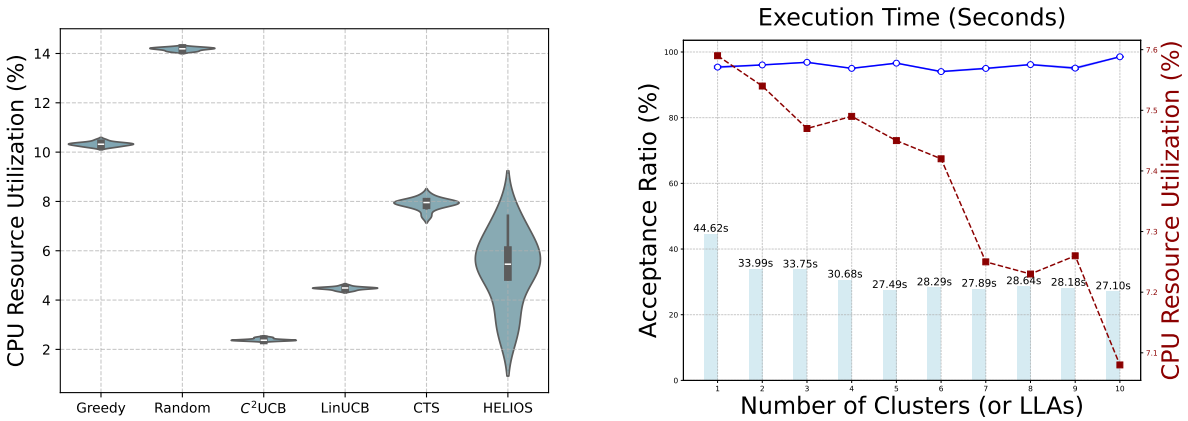
(B) Request Acceptance Ratio

FIGURE 5.4: Request Acceptance Ratio over the (A) GEANT and (B) DT2 Topologies when $\lambda = 2$, $\lambda = 5$ and $\zeta = 5$.



(A) Average Resource Utilization (B) Cluster-based Performance

FIGURE 5.5: Average Resource Utilization and Cluster-based Performance over the GEANT topology.



(A) Average Resource Utilization (B) Cluster-based Performance

FIGURE 5.6: Average Resource Utilization and Cluster-based Performance over the DT2 topology.

arrival rate and request duration are set as $\lambda = 2$ and $\zeta = 5$, respectively. We see that on the larger DT2 topology, our solution is able to admit a higher number of NSRs (up to 97%) under different arrival conditions (λ) and across the different chain lengths (ψ), compared to the baselines. However, in the smaller GEANT topology, the overall performance decreases across the different scenarios. This is attributed to the fact that there are fewer nodes per cluster in the GEANT topology, making it less likely for each cluster to contain an optimal combination of nodes that can meet the provisioning objectives. This is especially true for scenarios with longer chains and higher arrival rates, as node resources are more likely to be congested, leading to a lower overall acceptance rate. Despite this, our results suggest that HELIOS can learn a hierarchical provisioning policy that leads to better performance by fragmenting the original problem into sub-domains due to the inherent spatial structure of communication networks and leveraging context information in the placement decision.

Average Node Resource Utilization

In Figure 5.5a and Figure 5.6a, we observe the resulting average CPU resource utilization per node in the considered system, based on the provisioning policy of the proposed approach and

the baseline algorithms. The results are obtained for the scenario where the parameters for arrival rate and request duration are set as $\lambda = 5$ and $\zeta = 5$, respectively. Based on the provisioning objectives, we see that the provisioning policy learned by our approach leads to an average resource utilization per node per time slot that is relatively low, approximately 5%, especially in the larger DT2 network, which is reflected in the higher acceptance ratio for the network scenarios considered in Figure 5.4b. However, in the smaller GEANT topology, the average CPU resource utilization is considerably higher reaching up to 9% average utilization, which leads to a lower acceptance ratio for this topology, as fewer requests are deployed on the GEANT network. This highlights how our proposed solution is able to minimize average resource utilization, while seeking to maximize the number of accepted requests, by learning a cluster selection policy and a combinatorial node selection policy that achieves both objectives. Compared to the UCB-based approach employed by HELIOS and other baselines, the CTS approach typically leads to an unbalanced distribution of actions due to the random nature of the sampling process from the posterior distribution, which contrasts the more consistent policy of UCBs-based solutions [125].

TABLE 5.2: Avg. Execution Times of Baseline Algorithms and HELIOS [s]

Topology	**C^{2}UCB**	**LinUCB**	**CTS**	**HELIOS**
GEANT	26.7	51.3	29.8	38.4
DT2	71.2	182.3	46.4	96.5

Average Execution Time

We evaluate the execution times of the different learning-based baselines on deploying a set of requests ($\sim$25000) across the considered topologies. As the random and $\epsilon-$greedy approaches are heuristically simple and, therefore, have negligible execution times, we omit their results due to brevity. Our results are shown in Table 5.2. We see that on average, our proposed solution had a higher execution time compared to the majority of baseline solutions on the GEANT and DT2 topologies, taking 38 s and 96 s, respectively. This is a result of our our hierarchical approach which adds logical complexity and requires multi-level decisions on the placement decision of requests. Furthermore, the HLA takes multiple OGA steps during the LLA selection, which adds to the longer execution time.

Cluster-based Performance

Figures 5.5b and 5.6b show the scalable performance of our solution for different numbers of clusters. The results are obtained for the scenario where the parameters for arrival rate and request duration are set as $\lambda = 2$ and $\zeta = 5$, respectively. As it can be seen in the results, the performance of our solution begins to decrease after dividing the topology into more than three sub-domains, or clusters, in the GEANT topology. However, it does not show a similar trend in the larger DT2 topology. This is attributed to the average size of clusters in each topology, as more clusters in the GEANT topology reduces the number of nodes per cluster affecting the performance of the LLAs, while the number of nodes per cluster is larger in DT2. We also see

that increasing the number of clusters in each topology reduces the average execution time of our solution, highlighting the tradeoff between performance and speed on smaller topologies.

5.5 Chapter Conclusions

In this chapter, we presented a novel approach to the ONSP problem, a critical challenge in the efficient management of NGMNs. The increasing complexity and heterogeneity of services in these networks demands intelligent solutions that are capable of simultaneously meeting diverse provisioning objectives while maintaining scalability. The core contribution of this chapter lies in the reformulation of the ONSP problem as a HMAB problem and the development of a hierarchical solution, HELIOS. HELIOS aims to learn an online provisioning policy that effectively balances the often-conflicting objectives of maximizing the acceptance of NSRs and minimizing the average node resource utilization. Through extensive simulations on real network topologies, the proposed approach demonstrated its ability to outperform baseline solutions in terms of both average node resource utilization and the number of accepted requests. This highlights the potential of HMAB approaches in addressing complex dynamic resource allocation problems in dynamic network environments, such as the network edge.

In the broader context of this thesis, the work presented in this chapter contributes a valuable OL solution for network resource management. The application of a hierarchical multi-objective bandit approach offers a framework that can be extended and adapted to other resource orchestration challenges in complex networked systems. The insights gained from this chapter pave the way for future research focused on enhancing the efficiency and adaptability of online resource allocation algorithms in dynamic and distributed environments.

As part of the management of NSLs, the ability to effectively optimize the allocation of resources to deployed slices is critical for maintaining SLAs, ensuring efficient utilization of network resources, and adapting dynamically to varying traffic and QoS requirements across diverse application scenarios. Hence, in Chapter 6, we design a resource optimization framework that seeks to proactively allocate network resources to NSLs that serve a diverse range of applications, in order to meet their QoS and SLAs.

Chapter 6

SLA-Driven Proactive Slice Optimization

6.1 Introduction

The fifth generation of cellular networks (5G) promises to revolutionize communication by supporting a heterogeneous and novel range of applications and services and bringing an evolutionary leap in performance, capabilities, and user experiences [243]. To differentiate between the different applications and services that will need to be supported in 5G, three major service categories have been suggested: 1.) eMBB services that require very high data rates (>10 Gbps), 2.) mMTC services which require a communication medium for machine-type devices and 3.) uRLLC services which require very low latencies (1 ms-10 ms). Adequately supporting these services will be a difficult challenge given the current network architecture, which is based on dedicated hardware middle-boxes and a rigid infrastructure, to provide a "Best Effort" delivery model [244]. A key tool for addressing the diverse needs of the mobile network infrastructure is flexibility in resource management which is enabled by the growing NFV and SDN to enable dynamic spectrum allocation, baseband processing, scheduling, and task containerization [245].

A cornerstone technology enabling network flexibility and versatility is NS, which supports the creation of isolated virtual networks tailored to specific requirements of applications and services deployed on shared physical infrastructure [246]. However, the dynamic and heterogeneous nature of 5G resource demands presents a significant challenge for effective slice management and orchestration. Specifically, in multi-tenant mobile networks, such as 5G, where resources must be shared among numerous tenants, users and applications, off-the-shelf solvers are typically used to repeatedly solve the optimization problem of resource distribution. However, such solvers may take several hours to solve an optimization problem, which could violate SLAs, and often struggle to keep up with the increasing size of the optimization problems, especially in dynamic and fast-growing systems [247]. Other traditional optimization approaches, which often rely on reactive mechanisms, seek to adjust resource allocation only after performance degradation has been detected [248]–[250]. The inherent latency in the reactive approaches leads to suboptimal resource utilization, potential SLA violations, and a diminishing user experiences. Hence, while NS promises tailored

application performance in 5G and beyond networks, the current coarse-grained, traditional approaches raise questions about the adaptability and granularity of resource allocation approaches in an era of increasingly diverse performance requirements and a continuously evolving network environment [251]. More specifically, ensuring that the SLAs of deployed NSLs can be met requires novel solutions that constantly optimize the slice resource allocation, while deciding whether or not to admit new NSRs [252]. These solutions need to perform network resource optimization in order to guarantee that the requirements of the NSLs are met despite unpredictable potential load variations or traffic uncertainty [253], which may lead to lower resource efficiency, SLA violation or deterioration QoS [254]. To achieve such dynamic network slice management, proactive and robust AI/ML-driven mechanisms are required to enable efficient management for NSLs [187], [255] that share the same infrastructure and who's resource requirements profiles may change over time [256], [257], and calls for strategies that are aimed at adapting network slice configurations or capacities over time, in response to the changing network conditions (i.e., updating slice configurations based on the level of resource utilization or observed SLA violations) [258].

This chapter aims to address this by proposing a novel framework, Proactive Resource Optimization for Heterogeneous nETwork slices (PROPHET), that aims to optimize the capacities of NSLs that handle diverse traffic. Specifically, our proposed solution seeks to solve the dynamic resource allocation problem by decoupling the problem into two phases: **1.)** traffic demand prediction and **2.)** adaptive resource optimization. PROPHET seeks to predict the upcoming traffic demand of each deployed NSL, by using a Deep Neural Network (DNN) architecture, specifically, an hybrid architecture that combines Bidirectional LSTM layers with attention mechanisms. Based on the predicted traffic demand of each NSL, an online resource allocation problem is solved to optimize the allocation of resources in each NSL. By combining traffic demand prediction with adaptive resource optimization, our solution dynamically and preemptively reconfigures the allocated resources of NSLs in a robust and online manner, ensuring that the time-varying SLAs of the NSLs are met and maximizing the resource efficiency.

The contributions of this chapter aim to answer the following research questions described in Section 1.3.4.

RQ 4.1: How can ML-based traffic prediction be integrated with reinforcement learning to enable proactive resource allocation in heterogeneous 5G network slicing environments?

RQ 4.2: To what extent does **forecasting accuracy** and prediction horizon impact the **effectiveness** of proactive resource allocation strategies in maintaining SLA while maximizing resource efficiency?

We address the above challenges by introducing PROPHET, a proactive resource optimization framework that seeks to optimize the allocation of network resources in slice-enabled network, through DRL. We address ***RQ 4.1*** by designing a traffic forecasting model tailored for heterogeneous NSLs, enabling proactive network management. Our forecaster leverages a hybrid architecture combining LSTM networks with attention mechanisms to effectively

capture both temporal dependencies and salient traffic patterns. We test this model on a real-world UE network traffic time-series dataset, and show that our developed model demonstrates relatively good predictive performance, particularly in capturing dynamic traffic behaviors common in mobile network environments. This capability lays the foundation for anticipatory resource allocation, where forecasting accuracy directly enhances decision-making under uncertainty. To address ***RQ 4.2***, we extend the developed framework that incorporates traffic forecasts into the resource allocation policy optimization process. Our findings demonstrate that including predicted NSL traffic in the state representation of the DRL agent enables the learning of a policy that proactively allocates resources based on anticipated demand, thereby reducing SLA violations across heterogeneous NSLs. In particular, we observe that the average cumulative reward achieved with traffic prediction is influenced by the length of the prediction horizon—longer horizons tend to result in lower performance for the resource allocation task.

Concretely, our contributions can be summarized as follows:

- Design of an attention-based deep learning model for predicting network traffic demands across heterogeneous services in slice-enabled 5G environments.
- Development of a DRL framework that learns optimal resource allocation policies for heterogeneous traffic flows while adapting to dynamic network conditions.
- Performance evaluations through simulations demonstrate improved resource utilization and reduced SLA violations, captured by the agent's reward, compared to reactive resource allocation methods.

The following sections discuss the content in our paper submitted to the 2025 IEEE Global Communications Conference (GLOBECOM) [259]. Section 6.1 describes the research questions addressed in this chapter and discusses the contributions of the proposed learning-based framework. Section 6.2 describes the system model and formulates the considered Markov Decision Process (MDP) problem. Section 6.2.3 presents the PROPHET framework, including descriptions of the traffic prediction and DRL-based optimization modules. Section 6.4 describes the dataset used in the simulation, the simulation setup and the evaluation results. Finally, section 6.5 concludes the study in this chapter by reviewing the main contributions and highlighting the important results.

6.2 System Model and Problem Formulation

In this section, we present the considered system and formulate the problem.

6.2.1 System Model

We consider a Network Slicing scenario supported by a Virtual Radio Access Network (vRAN) network architecture and a single BS, where a set of NSLs $\mathcal{H} = \{h_1, h_2, \ldots, h_n\}$ must compete for a limited amount of network resources (*i.e.,* Physical Resource Blocks (PRBs)), as illustrated in Figure 6.1. Each NSL $h \in \mathcal{H}$ is characterized by a traffic demand d_h, a QoS requirement (*e.g.,*

maximum acceptable jitter $J_h^{\max}$), and a priority weight w_h that reflects its relative importance in the system. To meet the QoS requirements for each Network Slice, resources a_h within a bounded range must be allocated and are defined by a minimum allocation $a_h^{\min}$ and a maximum allocation $a_h^{\max}$, which yields the following inequality:

$$a_h^{\min} \leq a_h \leq a_h^{\max} \quad \forall h \in \mathcal{H} \tag{6.1}$$

The total available amount of PRBs are limited to K, and resource allocation decisions must ensure that the cumulative allocation across all NSLs does not exceed this capacity. This constraint is represented as:

$$\sum_{h=1}^{|\mathcal{H}|} a_h \leq K \tag{6.2}$$

Additionally, the jitter experienced by a NSL is influenced by its allocated resources. For each NSL h, we define a binary variable $v_h \in \{0,1\}$ to indicate whether an SLA violation occurs, where $v_h = 1$ represents a violation and $v_h = 0$ indicates SLA compliance. Additionally, we introduce a non-negative variable $\delta_h \geq 0$, which effectively quantifies the absolute difference between the total traffic demand of NSL h and the total amount of resources allocated to it. This variable captures the fairness or deviation in resource distribution relative to the actual demand, serving as a measure of allocation imbalance.

6.2.2 Problem Formulation

The objective of the optimization problem is to maximize the network's objective function by balancing NSL satisfaction and resource efficiency. Specifically, the goal is to maximize the weighted sum of SLA satisfied slices, represented by $\sum_{h\in\mathcal{H}} w_h(1-v_h)$, while minimizing the overall deviation between traffic demand and resource allocation, captured by $\sum_{h\in\mathcal{H}} \delta_h$. The resulting objective function ensures that high-priority NSLs are favored, SLA violations are penalized, and fairness in resource allocation is encouraged.

To detect SLA violations, we consider a simple jitter model which assumes that jitter is inversely proportional to allocated resources. This relationship is modeled as: $\frac{d_h}{a_h+\epsilon} \leq J_s^{\max}$ where ϵ is a small constant to avoid division by zero. If this condition is not satisfied for a given NSL, it is considered an SLA violation. This direct formulation allows us to clearly identify when the allocated resources are insufficient to meet the jitter requirement.

In our considered scenario, resource efficiency is evaluated by comparing the relative proportions of traffic demand and allocated resources. For each NSL h, the demand proportion p_h^d is defined as the fraction of total traffic demand attributed to it, while the allocation proportion p_h^a represents the fraction of total resources it receives. To quantify the mismatch between these proportions, we introduce the variable δ_h, which captures the absolute difference between demand and allocation $\delta_h = |p_h^d - p_h^a|$. This formulation ensures that even small mismatches between demand and allocation are accounted for, thereby promoting fairness and efficiency in resource distribution.

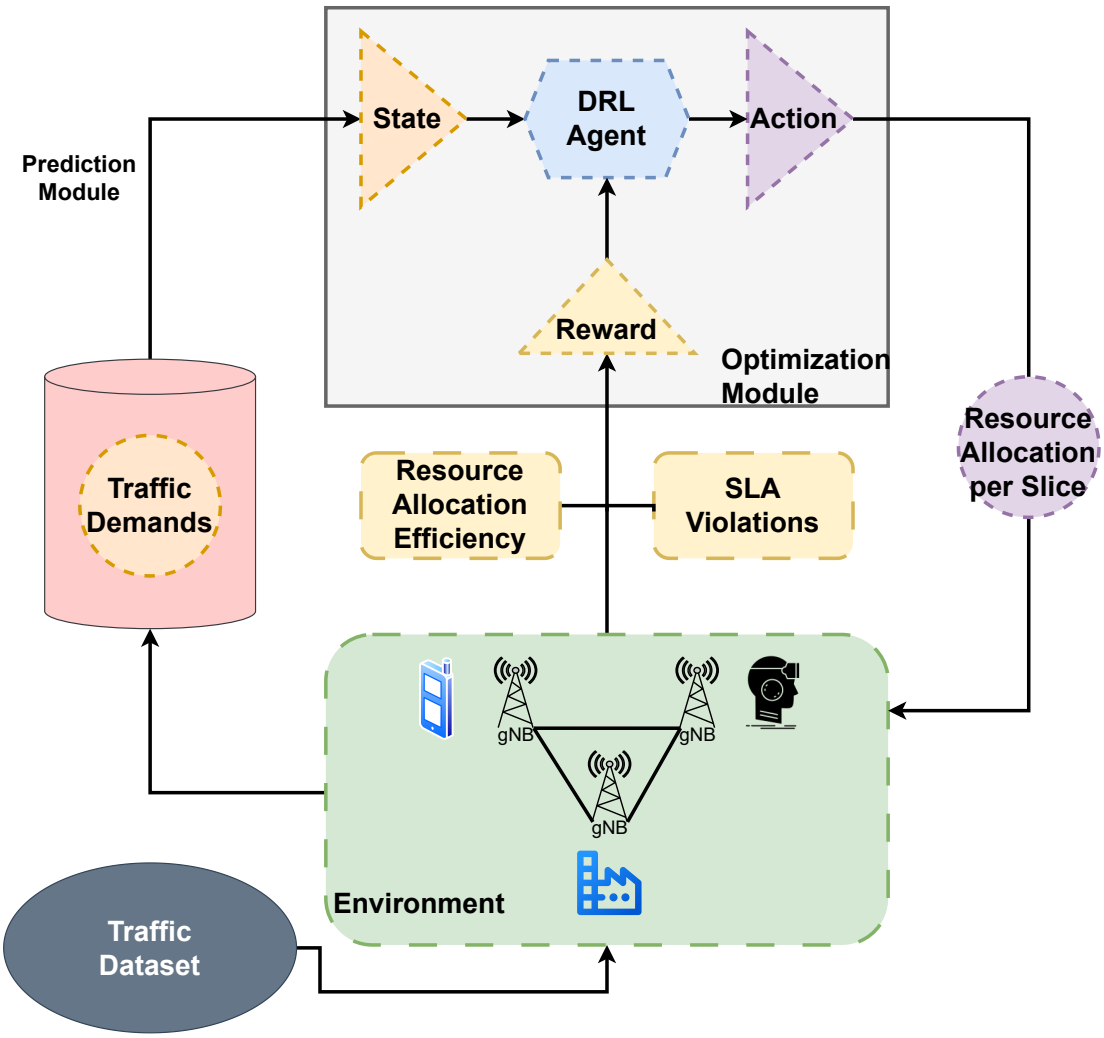


FIGURE 6.1: O-RAN System Model

The objective of the considered problem is therefore given by:

$$\max \quad \sum_{h\in\mathcal{H}} w_h(1-v_h) - \sum_{h\in\mathcal{H}} \delta_h \tag{6.3a}$$

$$\text{s.t.} \quad (1)(2) \tag{6.3b}$$

This problem can be formulated as a MDP, where an agent dynamically selects actions based on the current network state with the goal of optimizing a given reward function that considers both SLA satisfaction and resource efficiency.

6.2.3 MDP Formulation

To overcome the challenge of finding a solution to the formulated problem within a reasonable time, the problem is reformulated as a sequential decision problem within the framework of an MDP, where the goal is to learn a resource allocation policy that meets the objectives of the stated problem, while respecting the constraints. An MDP can generally be specified by a five-tuple which includes state space, action space, transition probability from the current state to the next, reward, and discount factor, i.e.,$\langle S, A, P, R, \gamma\rangle$, respectively.

- *State Space*: The state space represents the environment, which are defined as a set of states $s(t) \in S$, and represents the information available to the agent at the beginning of each time step t. We define this by: $s_t = \{d_h(t), a_h(t-1), v_h(t-1), c_h(t), f_h(t), K(t)\}_{h\in\mathcal{H}}$, where $d_h(t)$ represents the current traffic demand for NSL h, $a_h(t)$ represents the previous resources allocated to the NSL h, $v_h(t)$ represents the current SLA status of the slice, which includes the average jitter for the set of UEs $\mathcal{U}$ in the NSL, the average Channel Quality Indicator (CQI) for the set of UEs in the slice is represented by $c_h(t)$, $f_h(t)$ represents the forecasted aggregated traffic demand for the NSL over a specified future horizon (*i.e.,* the demand at $t+w$, where w is the considered horizon) and, finally, the capacity of the resources is represented by $K(t)$.

- *Action Space*: The action space typically defines the agent's behavior and is based on a policy π, which determines the agent's actions on the environment (*i.e.,* the resource allocation decision taken by the agent in a given state s_t). The agent's action is given by $a(t) = \{a_h(t)\}_{h\in\mathcal{H}}$, where $a_h(t)$ represents the proportion of total resources to be allocated to NSL h and $a_t \in A$. The environment processes this action a_t to derive the amount of resources (*i.e.,* PRBs) to effectively allocate by scaling the allocation proportions to the total available resources K (Eq. (2)) and applying per-slice minimum and maximum resource constraints (Eq. (1)).

- *Reward Function*: We define the reward as $r_t = R(s_t, a_t)$, which is received after taking action a_t in state s_t. Specifically, we compute the reward as: $r_t = \sum_{h\in\mathcal{H}} w_h(1-v_h) - \sum_{h\in\mathcal{H}} \delta_h$ where w_h is the priority weight of the NSL h, $v_h = \mathbb{1}(\frac{d_h}{a_h+\epsilon} \geq J_s^{\max})$ is an indicator function that evaluates to 1 if the SLA is violated and 0 otherwise, and $\delta_h \geq$

$|p_h^d - p_h^a|$ captures the resource efficiency of the action, with $p_h^d(t) = \frac{d_h(t)}{\sum_{h' \in \mathcal{H}} d_{h'}(t)}$ and $p_h^a(t) = \frac{a_h(t)}{\sum_{h' \in \mathcal{H}} a_{h'}(t)}$.

The goal of an agent is to find a policy $\pi : S \rightarrow A$ that maximizes the expected return:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right] \tag{6.4}$$

$\gamma \in [0, 1]$ controls the relative importance of immediate versus future rewards.

6.3 Proactive Slice Optimization

NSL traffic demands vary continuously due to dynamic users, applications, and services, requiring proactive resource optimization strategies that can predict the spatial-temporal characteristics of traffic patterns for the online optimization of NSLs [260], [261]. As a result, accurate traffic prediction is crucial for efficient resource allocation, as it enables InPs to optimize NSL resources and meet diverse SLAs, which is a critical goal of our proposed solution.

To maintain the QoS across the shared mobile network infrastructure, network operators must monitor the performance of NSLs throughout the network and dynamically reconfigure their capacity to prevent service degradation that would potentially violate SLAs.

6.3.1 Attention-based Traffic Prediction Module

To predict the traffic in the system, we develop a traffic prediction module that leverages a hybrid architecture combining Bidirectional LSTM layers with attention mechanisms to better capture the temporal dynamics of network traffic patterns [260]. The module plays a crucial role in enabling proactive decision-making for the DRL agent. Based on the MDP formulation, at each simulated time step t, after the agent has taken an action and the current environment state (including current traffic demand $\mathbf{d}(t)$) has been observed, the traffic forecasting component is invoked.

The forecaster employs a sequence-to-sequence approach with a configurable sequence length of 10 time steps, utilizing MinMaxScaler normalization to ensure stable training. It processes historical traffic data up to time t to generate predictions $f_h(t)$ for each NSL h over the next w time steps (where w represents the forecast horizon). The architecture consists of stacked Bidirectional LSTM layers (25 units each) with dropout regularization (0.3) for capturing long-range dependencies and avoiding overfitting, followed by dense layers for final prediction [262]. This design choice enables the model to learn both forward and backward temporal patterns in the traffic data, which is particularly important for capturing the often cyclical nature of network usage patterns.

6.3.2 DRL-based Optimization Module

Based on the MDP formulation, we leverage a DRL algorithm to learn a data-driven resource allocation policy. More specifically, in the considered MDP, the agent explores the environment by taking actions in several states, without having apriori knowledge about which actions are more beneficial or optimal, with the goal of eventually learning the best policy through experience [263].

To learn a proactive resource allocation policy, we employ the PPO algorithm [264], a state-of-the-art actor-critic RL method known for its stability and sample efficiency. PPO iteratively refines a policy $\pi_\theta(a_t|s_t)$ and a value function $V_\phi(s_t)$ parameterized by neural networks with parameters θ and ϕ, respectively. The policy network maps the current comprehensive state s_t which includes current network information (*i.e.,* actual traffic, observed jitter and CQI), historical data (past resource allocations, traffic, and QoS metrics), and critically, forecasted traffic demands f_h for a future horizon w to a distribution over possible resource allocation actions a_t.

The learning process involves collecting batches of experience by allowing the agent to interact with the simulated network environment using its current policy. For each state-action pair, the advantage A_t is estimated using Generalized Advantage Estimation (GAE), which quantifies how much better the chosen action was compared to the policy's average performance from that state. Then, PPO updates it's policy parameters θ by maximizing a clipped surrogate objective function $L^{CLIP}(\theta) = \mathbb{E}[\min(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}A^t, \texttt{clip}(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon)A^t)]$. This objective encourages policy improvement while constraining the update step size via the clipping mechanism (controlled by ϵ) , preventing destructive large updates. Concurrently, the value function parameters ϕ are updated by minimizing the mean squared error between $V_\phi(s_t)$ and the empirical return.

The proactive nature of the learned policy emerges primarily from the inclusion of forecasted traffic within the state s_t. By observing predicted future demands, the agent learns to anticipate resource needs and adjust allocations preemptively, rather than merely reacting to current conditions. The PPO algorithm, through its value function and advantage estimation effectively attributes responsibility for long-term outcomes, such as SLA violations or efficient PRB resource utilization resulting from responding to forecasts, enabling it to learn sophisticated, proactive resource allocation policies that approximate a solution to the constrained optimization problem in Eq. (3).

6.4 Performance Evaluation

6.4.1 Simulation Setup

In this section, we empirically evaluate the performance of our proposed PROPHET framework. We first describe the simulation setup, including the dataset and performance metrics. Subsequently, we present and analyze the results, focusing on the efficacy of the attention-based traffic prediction, the proactive resource allocation capabilities of the PPO

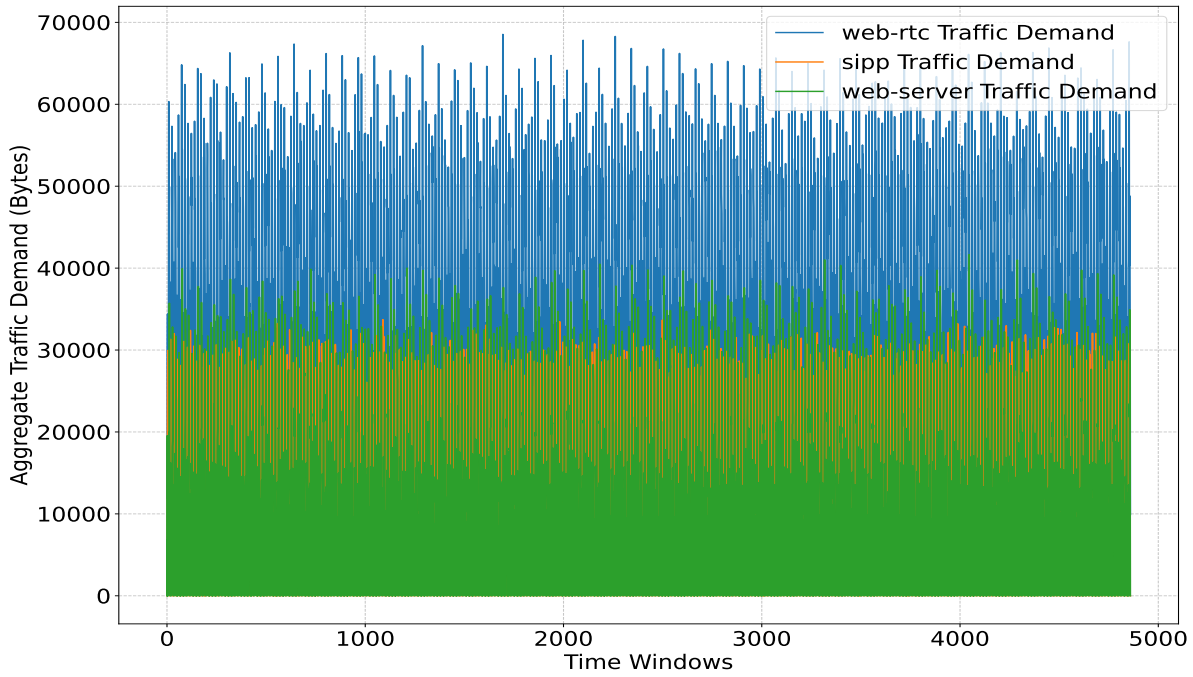


FIGURE 6.2: Per slice aggregate traffic demands

TABLE 6.1: Dataset Features and Output

Dataset Feature	Description
Current Demand	Traffic for considered applications: WebRTC, SIPp, Web-Server
CQI	Channel Quality Indicator reflecting the average quality of the radio link
Jitter	UE experienced Jitter (s)
Output	**Description**
Predicted Demand	Future traffic demand per NSL

agent in terms of SLA compliance, and overall resource utilization efficiency. We also compare PROPHET against relevant baseline approaches.

6.4.2 Dataset and Preprocessing

Our experiments leverage a publicly available dataset [265] composed of high-frequency (1 Hz) network performance data, including real-world network traffic and QoS statistics like Jitter and CQI per UE, captured from UEs operating in a commercial LTE network and from realistic network behavior in an office environment with scenarios based on network patterns observed between 10:00 AM and 11:00 AM. This dataset provides detailed time-series information for multiple UEs, quantifying traffic generated by distinct applications such as WebRTC, SIP protocol, and Web-Server, which serve as proxies for three heterogeneous network slice types in our simulation: a real-time communication slice (e.g., eMBB/URLLC hybrid), a signaling/control slice (e.g., URLLC/mMTC), and a best-effort data slice (e.g., eMBB). To align with practical DRL agent decision-making intervals and render this data suitable for modeling NSL resource allocation, we apply temporal aggregation to the raw 1 Hz data by

TABLE 6.2: Attention-LSTM Parameters

Hyperparameter	Configuration
Multi-Head Attention Layers	1
Number of Heads	4
Dimensionality of Attention Vectors	64
LSTM Layers	1
LSTM Units	25
Units per Hidden Layers	16
Dense Layers	1
Optimizer	Adam
Learning Rate	10^{-3}
Loss Function	MSE
Activation Function	ReLU

partitioning it into contiguous, non-overlapping time windows of $W_s = 10$. Within each window, traffic volumes for UEs belonging to the same service type are summed to compute the aggregate demand for each NSL, and key QoS metrics like jitter are averaged per UE to yield representative performance indicators. This preprocessing yields the temporally aggregated dataset used as input for our simulation environment, as illustrated in Figure 6.2, where each record encapsulates the state of slice demands and average UE conditions over a discrete time step.

Our simulations are conducted using OpenAI Gym [266], a widely recognized toolkit in Python. The Gym API serves as a bridge between our learning-based algorithm and our Python-based simulation environment. Specifically, Gym provides a standardized interface that allows for seamless integration of our custom algorithms with various simulated scenarios, enabling us to efficiently test and refine our learning approaches.

TABLE 6.3: PPO Hyperparameters

Hyperparameter	**Value**
Learning Rate	3×10^{-4}
Steps per Update	1024
Batch Size	64
Number of Epochs	10
Discount Factor (γ)	0.99
GAE Lambda (λ)	0.95
Clip Range (ϵ)	0.2
Entropy Coefficient	0.01
Value Function Coefficient	0.5
Max Gradient Norm	0.5

The attention-based prediction module (Section 6.3.1) is trained on the initial 70% of the aggregated dataset to predict per-slice traffic demands w steps ahead, where $w = 5$ windows. Specifically, we use the Mean Squared Error (MSE) loss with the Adam optimizer and incorporate early stopping to speed up the training process and avoid overfitting. The specific details about the hybrid architecture and the hyper-parameters used are given in Table 6.2

and are generated using grid search for hyperparameter tuning. The remaining 30% is used for evaluating the DRL agent. The PPO agent is configured with the parameters given in Table 6.3. The three NSLs (*i.e.,* WebRTC, SIPp, Web-Server) are configured with specific SLA targets for maximum jitter and minimum effective CQI, and resource allocation bounds (*i.e.,* a_h^{min} and a_h^{max}), as outlined in Section 6.2.1. We consider a 20 MHz configuration for the BS, which translates to 100 PRBs available to be allocated between the NSLs, based on which, we set the values of a_h^{min} and a_h^{max} to 10 PRBs and 30 PRBs, respectively. The resource allocation constraints reflect the heterogeneous requirements of different service types in the network. Finally, the network environment simulates QoS metrics (*i.e.,* jitter, effective CQI) based on the agent's resource allocation decision, observed traffic, and current dataset CQI, using the model described in Section 6.2.2.

6.4.3 Benchmark Algorithms

To evaluate the advantage of our proactive solution, we compare the performance of our approach against a PPO agent that is identical in architecture and hyperparameters, but without access to traffic forecasts in it's state space. Hence, the agent makes decisions based only on current and historical observations, and is considered as a reactive baseline solution.

6.4.4 Performance Metrics

We evaluated the performance of our approach by considering the following metrics:

- **Traffic Forecasting Accuracy**: This is evaluated using Normalized Root Mean Square Error (NRMSE) for each NSL/service type.
- **Average Cumulative Reward**: This is the average sum of discounted rewards obtained by the DRL agent per episode during evaluation.

6.4.5 Simulation Results

Traffic Forecasting Performance

TABLE 6.4: Forecasting Performance Metrics

Service Type	NRMSE
WebRTC	21.00%
SIPp	13.15%
Web-Server	14.50%

The training and validation performance of the traffic forecaster is shown in Figure 6.3. We see that the attention-based model is well fitted to the dataset that is constructed for the purpose of our simulations. The performance of the attention-based traffic forecasting module is presented in Table 6.4, based on the NRMSE evaluation metric. We can observe that the prediction module demonstrates relatively good performance across all three service types, achieving NRMSE values of 21%, 13.15%, and 14.5% for WebRTC, SIPp, and Web-Server traffic, respectively, indicating that the hybrid architecture captures some of the temporal

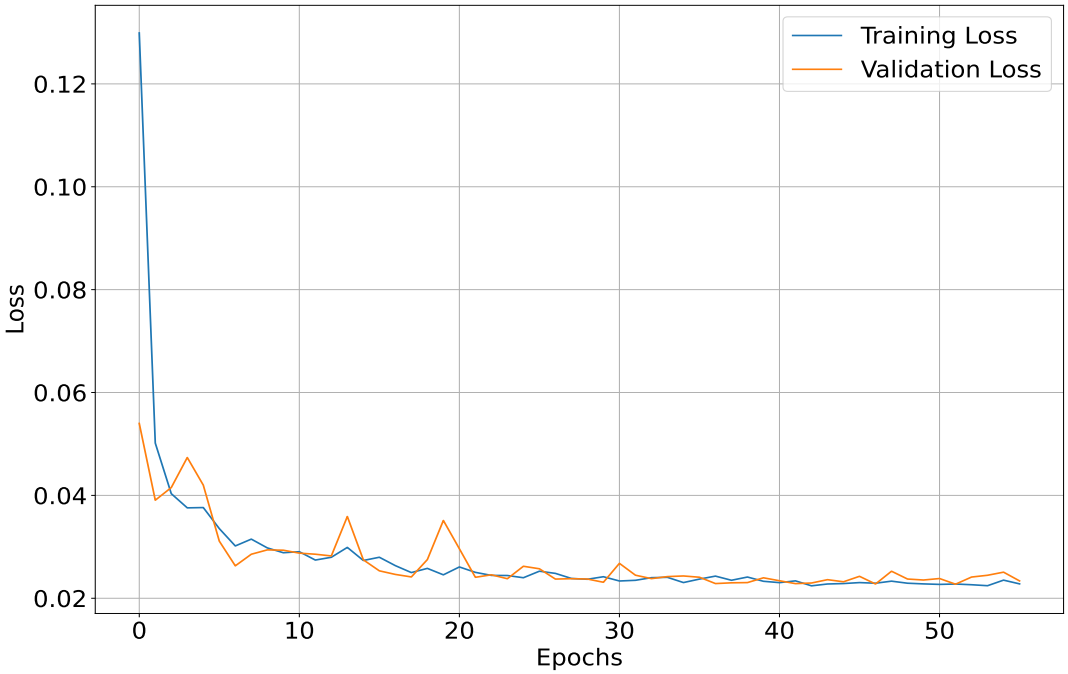


FIGURE 6.3: Training and Validation Loss

dynamics of the heterogeneous network traffic patterns within each NSL. Nevertheless, we are able to leverage the model in our simulations to predict future traffic and states to aid the DRL agent in making proactive resource allocation decisions that minimize SLA violations and optimize for resource efficiency.

Average Cumulative Reward

We investigated the impact of the forecasting horizon w on PROPHET's performance. As shown in Figure 6.4, a horizon of $w = 5$ achieved a better average cumulative average reward (approx. average: -516) compared to $w = 20$ (approx. average: -1040) and significantly outperformed the standard PPO baseline (approx. average: -752). This suggests that there is an optimal range for w, as horizons that are too short may not provide sufficient time for proactive actions, while horizons that are too long may introduce excessive forecast uncertainty or incorporate irrelevant future information.

6.5 Chapter Conclusions

In this work, we introduced the PROPHET framework for proactive resource optimization in 5G environments with network slices that support heterogeneous services. By integrating an attention-based traffic prediction module with a PPO-based DRL agent, we demonstrated a systematic approach to anticipatory resource allocation that addresses the fundamental challenge of dynamic traffic management in slice-enabled networks. Our framework leverages multi-dimensional state representations, including current network conditions, queue occupancy levels, and traffic forecasts, enabling the DRL agent to learn allocation policies that balance SLA compliance with resource efficiency.

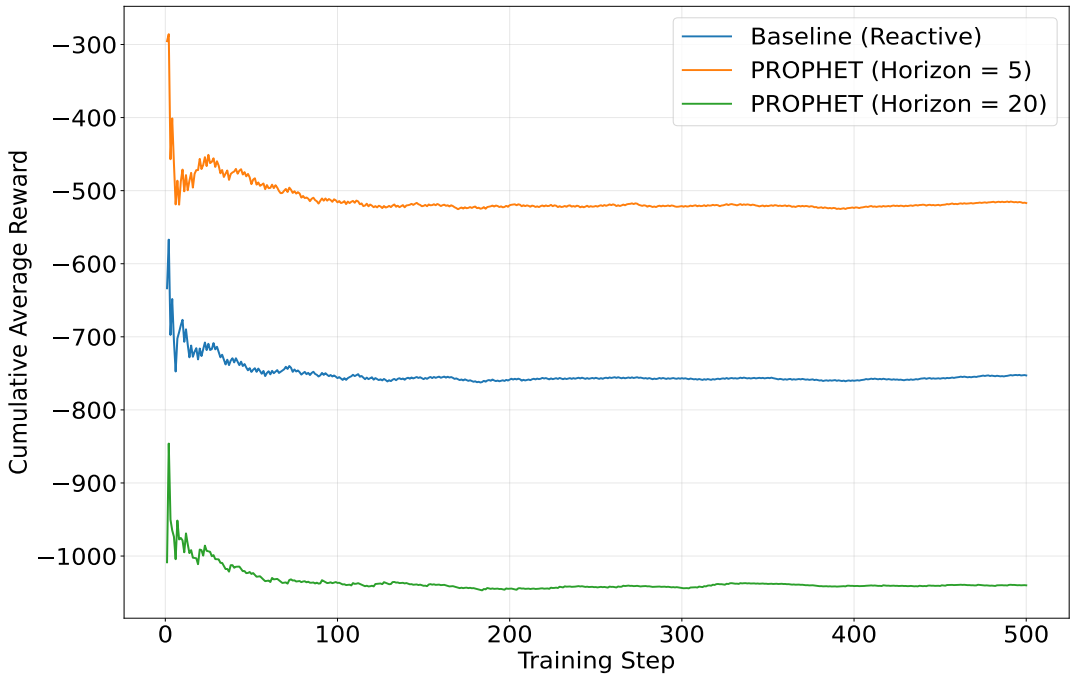


FIGURE 6.4: Average Reward: PROPHET vs Standard PPO

Through comprehensive evaluation using real-world LTE network traffic data, we observed that the prediction module achieved NRMSE values of 21%, 13.15%, and 14.50% for WebRTC, SIPp, and Web-Server NSL traffic, respectively. While these forecasting accuracies indicate room for improvement, the integrated PROPHET framework still demonstrated a better average cumulative reward which encapsulates it's resource efficiency and SLA violation policy, compared to reactive baseline methods. The framework successfully maintains service differentiation, enforcing slice-specific resource bounds while adapting to the heterogeneous QoS requirements of different slice types.

Specifically, our findings reveal both the potential and current limitations of AI-driven proactive network management. The moderate prediction accuracy highlights the inherent challenge of forecasting highly variable network traffic for heterogeneous NSLs. However, the framework's architecture proves resilient, as the DRL component learns to balance proactive allocation based on the predictions of future traffic demands with reactive adjustments based on observed conditions. This hybrid solution emerges naturally from the reward structure, which heavily penalizes SLA violations while encouraging efficient resource utilization.

Finally, the PROPHET framework represents a meaningful step toward intelligent and autonomous network slice management in 5G and beyond mobile networks. As network traffic patterns become increasingly complex and service requirements more stringent, the combination of predictive analytics and adaptive learning towards proactive and predictive optimization, as demonstrated in this work, will be essential for maintaining QoS requirements while maximizing infrastructure efficiency in next-generation mobile networks.

Chapter 7

Conclusions and Future Work

In this chapter, we first summarize the contributions of this thesis in Section 7.1, by providing a general review of the contributions in the previous chapters. Then, in Section 7.2, we discuss potential avenues for future research towards autonomous, intelligent, and sustainable network slicing.

7.1 Summary of Contributions

In this thesis, we investigate and addresses important challenges relating to the dynamic provisioning of network slices in softwarized and distributed edge computing environments. Beginning with a comprehensive literature review that established the theoretical foundation of our work, we identified significant gaps in the current resource management solutions and frameworks towards the realization of real-time network slicing, admission control, and resource optimization strategies. As a result, our work's contributions have effectively addressed these challenges through three complementary frameworks which advance the state-of-the-art in intelligent network slice management. Collectively, these frameworks tackle the complexities of resource management in modern edge-based mobile networks, offering innovative and intelligent solutions towards improving resource efficiency, service quality, and scalability in such networks.

7.1.1 Online Slice Admission Control

In Chapter 3, we introduced an innovative online slice admission control approach that leverages online reservation policies for multi-dimensional resource evaluation in heterogeneous mobile networks, to address the research questions in Section 1.3.1. This proposed method demonstrated how strategic resource reservation can maximize the revenue of infrastructure providers despite the uncertainty around the value of future slice requests. We show the efficacy of our solution in comparison to other slice admission control solutions and show that regardless of the selected reservation policy, it is able to consistently outperform the benchmark solutions across key performance metrics, such as revenue maximization and resource efficiency.

The research in this chapter addresses the following research questions:

- ***RQ 1.1***: How can admission control policies effectively manage the **multi-dimensional** resource demands of network slices while maintaining high resource efficiency in dynamic mobile networks?
- ***RQ 1.2***: How can online admission control policies ensure long-term **revenue optimization** in network slicing, while accommodating heterogeneous slice requirements and uncertain demand?
- ***RQ 1.3***: How does **economic disparity** among slice tenants influence admission control decisions, and how can admission control policies balance revenue optimization with fairness?

We address ***RQ 1.1*** and ***RQ 1.2*** by formulating the Slice Admission Control problem as an Online Multidimensional Knapsack Problem and utilizing linear and exponential reservation policies that dynamically adjust admission thresholds based on the current resource utilization level. These policies account for heterogeneous resource dimensions, preserving capacity for higher-valued future requests by calculating scarcity across each resource type and ensuring that slice requests are only admitted if there is enough capacity and their value is larger than or equal to all evaluated resource dimension costs. This approach enables our proposed online slice admission control algorithm to maximize the long-term revenue received from admitted network slice requests.

We addressed ***RQ 1.3*** in this chapter by modeling economic inequality through parameter ω in a Beta distribution, where low ω values represent high inequality with tenant values polarizing towards minimum (1) or maximum (σ) values. Our evaluation demonstrated that in high inequality scenarios, our solution selectively rejected lower-value slice requests to reserve capacity for future higher-value opportunities. This selective approach resulted in lower acceptance ratios compared to greedy algorithms but generated significantly higher revenue. Economic inequality thus creates opportunities for strategic admission control, leading to reduced average resource utilization while achieving more efficient resource allocation under network uncertainty.

By leveraging the online reservation policies in our online slice admission control algorithm, we have shown that the proposed solution to the slice admission control problem is able to reserve scarce network resources by dynamically updating the admission threshold for sequentially arriving slice requests. More specifically, the superiority of our solution is highlighted by the fact that it increases infrastructure provider revenue by over 12%, while reducing the resource utilization by up to 1.7%.

7.1.2 Drift-Aware Policy Selection for Slice Admission Control

In Chapter 4, we developed the Drift-AwaRe upper confIdence bOund framework for dynamically learning and selecting optimal slice admission control policies based on historical request patterns. By reformulating the slice admission control policy selection problem as a multi-armed bandit problem, our proposed framework learns a policy that effectively prioritizes high-value slice requests, significantly enhancing revenue generation for

infrastructure providers. The contributions of this chapter also answer the research questions posed in Section 1.3.2. The Drift-AwaRe upper confIdence bOund framework is a novel online learning framework for slice admission control to dynamically update the admission threshold of slice requests based on detecting drifts in the patterns of previous network slice requests.

The research in this chapter addresses the following research questions:

- ***RQ 2.1***: How can network slice admission control policies be **dynamically adapted** to cope with evolving network slice request patterns in 5G networks?
- ***RQ 2.2***: What is the effect of **temporal shifts** in the distribution of network slice request characteristics on the performance and robustness of slice admission control policy configurations?
- ***RQ 2.3***: How can online learning techniques be integrated with change detection mechanisms to enable **continuous selection** and adjustment of admission control policy parameters for network slicing in dynamic environments?

We address ***RQ 2.1*** by formulating the slice admission control policy selection problem as a multi-armed bandit problem in which the bandit agent determines which admission control policy to apply for a set of arriving network slice requests. Based on the formulated problem, an upper confidence bound-based algorithm is proposed for adaptively selecting between the admission control policies in order to learn their performance in an online manner.

The ***RQ 2.2*** is addressed by designing a synthetic dataset that contains distribution shifts in order to evaluate how the statistical features of slice requests affect performance of the admission control policies. Based on these observations, we design a change detection mechanism based on adaptive windowing to detect the changes in the statistical features and trigger policy adjustments through adaptive thresholds.

Finally, ***RQ 2.3*** is effectively addressed by integrating the sliding-window upper confidence bound algorithm with a change detection mechanism to create a data-driven framework that learns both from historical performance data while remaining responsive to environmental changes (i.e., fluctuations in the WTPR θ). Our framework explores the trade-off between exploring different admission control policies to gather information about their performance and exploiting known high-performing admission control policies, particularly when the underlying distribution of requests is non-stationary.

By combining online learning with change detection, our proposed framework is able adapt to changing network scenarios, which are captured by the characteristics of the arriving network slice requests. We compare the performance of our approach to the benchmark linear and exponential reservation policies, the greedy first-come-first-serve policy, as well as the vanilla version of the upper confidence bound algorithm that adaptive selects between the two policies but does not detect any changes in the characteristics or features of requests. Our results demonstrate that our propose approach is able to outperform all the baseline solutions in terms of admitting high-valued requests, achieving approximately 4.5% higher revenue gain, while maintaining the efficient utilization of resources. This is a result of our solution being

able to detect when the characteristics of historical requests deviate from a given distribution within a given time duration, and adapt to this change by either exploiting the policy that has performed well so far, or exploring other policies that could lead to better performance.

7.1.3 Hierarchical Placement Learning for Network Slice Provisioning

In Chapter 5, we focused on addressing challenge of slice provisioning in distributed edge environments through the hierarchical multi-armed bandit framework, which is a novel hierarchical learning approach to network slice provisioning. By partitioning the search space of the network into sub-domains based on the identification of connected communities in the graph, we developed a hierarchical framework of bandit agents that are distributed over the network to address the slice provisioning problem. HELIOS effectively learns a high-level mapping between the requirements of slice requests and their acceptance and utilization in different sub-domains of the network.

The research in this chapter addresses the following research questions:

- ***RQ 3.1***: How can we increase the rate of admitted NSLs while reducing the required allocated resources, especially in **large-scale** networks with highly **dynamic** resource and user requirements?
- ***RQ 3.2***: How can a **hierarchical model** be devised so that agents in different network subdomains make their own placement decisions?
- ***RQ 3.3***: How can we design a NS-provisioning performance metric that considers **multiple objectives** and enables a network policymaker to specify the objectives' relative **importance** and mutual **fairness**?

We addressed ***RQ 3.1*** by demonstrating how dividing the network into clusters using community detection (Louvain algorithm) and implementing a two-tier agent structure (HLA and LLA) provides an effective slice provisioning solution. Our experimental results demonstrated that the hierarchical approach outperforms centralized baselines across multiple network topologies with heterogeneous resources by admitting a higher number of requests and having marginally higher utilization than the baselines.

RQ 3.2 is addressed by proposing HELIOS, a novel two-level hierarchical learning system to solve the online NSP problem by jointly placing the VNFs of a SFC in the network. At HELIOS' high level, a contextual bandit agent directs each slice request to a specific region in the network, depending on the measured resource state and slice features. At HELIOS' low level, a combinatorial bandit agent determines the nodes on which the VNFs will be placed. HELIOS is designed to learn a placement policy that scales with network size.

To address ***RQ 3.3***, we leverage the GGI aggregation function which scalarizes and balances multiple provisioning objectives, together. By maximizing this function, we aim to find a point on the Pareto front of the multi-objective optimization problem. This allows for a direct optimization of the different provisioning objectives, enabling trade-offs based on the chosen weights.

By designing a novel HDF that enables distributed bandit agents to solve the ONSP at different locations in the network, we enable scalable slice provisioning in distributed and heterogeneous edge environments. We have shown that HELIOS achieves higher acceptance rates while maintaining lower resource utilization compared to both myopic and intelligent benchmark solutions. The hierarchical strategy learned by HELIOS effectively manages the complexity inherent in distributed edge networks, providing a scalable approach to network slice provisioning. Our findings in this chapter contribute to design of hierarchical learning-based frameworks that can combine the algorithmic efficiency of bandit learning with structural awareness of network topologies, enabling multi-objective optimization in resource-constrained edge environments without requiring complete prior knowledge of request patterns or infrastructure capabilities.

7.1.4 Proactive Resource Optimization for Heterogeneous Network Slices

In Chapter 6, we investigated the challenge of dynamic resource allocation in mobile networks through proactive resource optimization for heterogeneous NSLs, and in doing so, we address the research questions described in Section 1.3.4. Our approach combines traffic forecasting with DRL to enable intelligent, forward-looking allocation of network resources. By first predicting future traffic demands using an LSTM-attention-based forecaster, and then incorporating these predictions into a DRL-based resource allocation framework, we are able to reduce SLA violations and improve overall network resource efficiency. The integrated solution allows our framework to anticipate potential demand fluctuations and adapt resource distribution between heterogeneous NSLs accordingly, offering a proactive and effective strategy for managing dynamic and diverse NSL requirements in next-generation mobile networks.

The research in this chapter addresses the following research questions:

- ***RQ 4.1***: How can ML-based traffic **prediction** be integrated with **reinforcement learning** to enable **proactive resource allocation** in heterogeneous 5G network slicing environments?
- ***RQ 4.2***: To what extent does **forecasting accuracy** and prediction horizon impact the **effectiveness** of proactive resource allocation strategies in maintaining SLA while maximizing resource efficiency?

The ***RQ 4.1*** is addressed by designing an attention-based DL model for predicting mobile network traffic demands across heterogeneous services in slice-enabled 5G environments. The performance of the model is evaluated based on a real-world dataset and shows promising results in terms accurately predicting traffic demands of heterogeneous services. To address ***RQ 4.2***, we developed a DRL-based framework that learns optimal resource allocation policies for heterogeneous traffic while adapting to dynamic vRAN conditions. By evaluating the learned resource allocation policy through it's cumulative average reward and comparing it to a purely reactive solution, we show that the framework is able to achieve the goals of improving resource

efficiency and minimizing SLA violations at a better rate than the baseline solution, depending on the different prediction horizons.

By combining traffic prediction with the power of DRL, we create a unified framework that not only reacts to current network conditions but also proactively plans for future spikes in demand. This synergy allows the DRL agent to make informed resource allocation decisions based on anticipated traffic patterns, significantly improving performance in dynamic, heterogeneous network slicing environments. The predictive insights provided by the LSTM-attention model enhance the DRL agent's ability to minimize SLA violations while optimizing resource efficiency, especially under variable and dynamic traffic scenarios.

Collectively, these contributions represent comprehensive approaches to intelligent network slice management that spans admission control, policy selection, hierarchical learning and proactive/predictive optimization. Our frameworks demonstrate significant improvements in key performance metrics including revenue generation, resource utilization, service acceptance rates and QoS optimization. The adaptability of our solutions to changing network conditions addresses the dynamic nature of modern network environments, providing robust mechanisms for optimal resource allocation under network uncertainty. Furthermore, these research contributions establish a foundation for future work in intelligent, AI-driven network management, particularly as network infrastructures continue to evolve toward more distributed, heterogeneous, software-defined architectures. The methodologies and frameworks presented here offer promising directions for further investigation into autonomic network management systems that can seamlessly adapt to emerging networking paradigms, architectures, and novel service requirements under the umbrella of multi-dimensional network slicing.

7.2 Future Work

In this section, we discuss potential areas for future research that would extend the contributions made in this thesis. Primarily, the aims of the contributions presented in this thesis have focused on how NSLs can be admitted, embedded and adjusted in Next-Generation Mobile Networks, and in doing so, the contributions address the current gaps in research. Building upon these foundations, we propose several avenues for future work in this field that were elaborated based on the open issues of our contributions in this thesis. Below, we discuss the potential open questions for each chapter this thesis presents.

Digital Twin Evaluation. The evaluation of the proposed online algorithm and online learning based frameworks on a viable network digital twin test-bed constitutes an important avenue for future work. While the results from evaluating our solutions in controlled simulation environments showed promising results, their efficiency and generalizability to real network scenarios would only be determined when they have been evaluated in digital twin environments which reflect changes in the network infrastructure, in order to analyze their response to changing network scenarios and patterns, as well as overcome the performance discrepancy between simulations and real network environments, i.e., the *simulation-to-reality gap* [267].

With the recent development of large-scale emulation platforms like Colosseum [268] that can imitate dynamic wireless channel conditions in city-scale scenarios, the performance of our algorithms and frameworks can be tested under real network conditions. This is especially important for the data-driven online learning solutions, which should ideally be trained in live systems to avoid the unforeseen scenarios that are typical assumed in offline simulations. Our proposed algorithms could also be further augmented with time, resource, and action constraints in such environments to prevent the live systems or test-beds from going into unsafe states.

Multi-Domain Slice Federation. A significant challenge in provisioning network slices in next-generation mobile network is the deployment of network slices that span multiple domains. In multi-domain network slicing, the goal would be to allocate resources across multiple network domains (e.g., Edge/RAN, TN, and CN) to create virtualized network slices that are tailored to diverse services. While we typically only consider a single provider or single domain (i.e. Edge) in the majority of the works in this thesis, extending our results to multi-provider and multi-domain scenarios in which slice providers need to lease resources from other providers that own resources across different domains, is an interesting research direction. This can be addressed by extending the proposed solution in Chapter 5 to include nodes from other clusters to address the challenge of limited resources in certain clusters and increase the probability of successfully deploying NSRs across multiple clusters. This could also be considered as *leasing* resources from other clusters or domains by allowing the LLA agents to coordinate to address such a problem. By characterizing such a federated multi-domain scenario, novel online algorithms and learning-based frameworks can be developed to determine an optimal resource federation and slice composition across multiple domains, with the goal of maximizing the number of accepted NSRs and the revenue gained by each InP for hosting a part of the NSL in it's domains.

Graph Reinforcement Learning for Network Slicing. The goal of zero-touch network management in next-generation mobile networks requires intelligent solutions, driven by the ongoing developments in AI and ML techniques. One of such developments is in the fast-growing field of Graph Reinforcement Learning (GRL), which is a novel constructive decision-making method for graph-based problems [269]. More specifically, it a more appropriate ML tool for handling combinatorial optimization problems on graphs, where ML models can be used to execute known algorithms, improve existing algorithms through data-driven learning, or discover new algorithms through RL. In the context of NS, such an approach can be used to develop novel algorithms for network management problems, such as the placement or provisioning problem addressed in our third contribution. This approach could be particularly important for dynamic network slicing, where the state of the network evolves over time, as it overcomes the limitation of current ML algorithms for graph-based problems which typically assume that the network structure or topology remains static over time.

Intent-based and Explainable Slice Optimization. As a result of heterogeneous traffic and dynamic QoS requirements in mobile networks, there is an increasing need to automatically manage network resources and operations towards the goal of Zero-Touch close-loop orchestration. Intent-Based Networking (IBN) has recently emerged as an advanced approach to network management that focuses on automating and simplifying how networks are configured, operated, and maintained by enabling administrators to specify high-level intents, *i.e.,* what they want the network to achieve, using natural language or predefined policies. More specifically, an interesting future work would be to combine intent-based network management with Large Language Models (LLMs) [270] in order to optimize the performance of deployed network slices towards meeting certain performance requirements by translating the high-level intents into low-level network actions, such as slice instantiation, migration or scaling [271]. However, to realize intent-based network slice optimization would also require the development of eXplainable Artificial Intelligence (XAI) techniques that can provide clear, interpretable, and transparent explanations for their networking decisions and predictions, in order to ensure that the stability of the network through clearly defined and human-interpretable logic and actions.

Sustainable Network Slicing. With the increasingly important role of computing infrastructure in next-generation mobile networks and their critical role in delivering novel, data-hungry AI applications and services, there is a growing concern about the carbon footprint of future mobile networks. Given the role of NS in providing a customized platform for delivering such applications and services to geographically distributed users, the allocation of heterogeneous network and compute resources to NSLs should be evaluated for their impact on sustainability, with the aim of meeting strict QoS requirements while minimizing each NSLs's carbon footprint. More specifically, the creation, deployment and management of NSLs by sustainably routing traffic [272] and scheduling resources [273] in a way that significantly improves the energy efficiency in mobile networks by potentially reducing energy wastage, water consumption and operational costs while minimizing the environmental impact [274] of provisioning NSLs for increasingly demanding AI applications and services, is a promising direction for future research towards the goal of *sustainable network slicing*. Such a consideration is especially crucial in developing nations with unique environmental and resource constraints [275].

This goal can be advanced through intelligent decision-making and rigorous evaluation of learning-based algorithms on realistic sustainability benchmarks [276]. Concretely, extending the HDF from the third contribution to an RL setting for dynamic control would allow both the HLA and distributed LLAs to monitor per-cluster slice performance and resource utilization, then either adjust processor power states within a cluster or migrate workloads from heavily loaded, high-carbon clusters to lower-carbon alternatives, thereby reducing total energy consumption [277]. In parallel, ML models, such as the one designed in our fourth contribution, can be used to perform renewable-energy forecasting across clusters and time horizons to inform NSL placement and scheduling at locations and periods with higher expected renewable energy availability.

To conclude, the different contributions made in this thesis provide a solid foundation for real-time and adaptive network slicing in next-generation mobile networks through enhanced network intelligence and automated resource allocation. They effectively provide a basis for building and designing intelligent future mobile networks through the different research directions which would enable AI-driven mobile networks, which can be dynamically configured through intents, evaluated on large-scale testbeds or digital twins and be sustainable for the mobile users, service and infrastructure providers, and most importantly, the overall environment.

List of Publications

Key Publications

Paper [1] J. Ajayi, A. Di Maio, T. Braun, and D. Xenakis, "An Online Multi-dimensional Knapsack Approach for Slice Admission Control," in *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*, 2023, pp. 152–157. DOI: 10.1109/CCNC51644.2023.10060460.

Paper [2] J. Ajayi, A. Di Maio, and T. Braun, "Drift-Aware Policy Selection for Slice Admission Control," in *NOMS 2024-2024 IEEE Network Operations and Management Symposium*, 2024, pp. 1–9. DOI: 10.1109/NOMS59830.2024.10575766. 🏆

Paper [3] J. Ajayi, A. Di Maio, and T. Braun, "Hierarchical Placement Learning for Network Slice Provisioning," in *2025 IEEE 50th Conference on Local Computer Networks (LCN) [Accepted]*, IEEE, 2025.

Paper [4] J. Ajayi and T. Braun, "Proactive Resource Optimization for Heterogeneous 5G Network Slicing," in *GLOBECOM 2025 - 2025 IEEE Global Communications Conference*, 2025 [Submitted].

Other Publications

Paper [5] E. Schiller, J. Ajayi, S. Weber, T. Braun, and B. Stiller, "Toward a Live BBU Container Migration in Wireless Networks," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 301–321, 2022. DOI: 10.1109/OJCOMS.2022.3149965.

Paper [6] D. Xenakis, E. Samikwa, J. Ajayi, A. D. Maio, T. Braun, and K. Schlegel, "Towards Personality Detection and Prediction using Smartphone Sensor Data," in *2023 21st*

Mediterranean Communication and Computer Networking Conference (MedComNet), 2023, pp. 121–130. DOI: 10.1109/MedComNet58619.2023.10168869.

Bibliography

[1] K. Alam, M. A. Habibi, M. Tammen, *et al.*, *A Comprehensive Tutorial and Survey of O-RAN: Exploring Slicing-aware Architecture, Deployment Options, Use Cases, and Challenges*, 2024. arXiv: 2405.03555 [cs.NI]. [Online]. Available: https://arxiv.org/abs/2405.03555.

[2] L. Ismail and R. Buyya, "Artificial Intelligence Applications and Self-Learning 6G Networks for Smart Cities Digital Ecosystems: Taxonomy, Challenges, and Future Directions," *Sensors*, vol. 22, no. 15, 2022, ISSN: 1424-8220. DOI: 10.3390/s22155750. [Online]. Available: https://www.mdpi.com/1424-8220/22/15/5750.

[3] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A Survey on Low Latency Towards 5G: RAN, Core Network and Caching Solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3098–3130, 2018. DOI: 10.1109/COMST.2018.2841349.

[4] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017. DOI: 10.1109/COMST.2017.2705720.

[5] Q. Cui, X. You, N. Wei, *et al.*, *Overview of AI and Communication for 6G Network: Fundamentals, Challenges, and Future Research Opportunities*, 2025. arXiv: 2412.14538 [cs.NI]. [Online]. Available: https://arxiv.org/abs/2412.14538.

[6] M. Helmy, A. A. Abdellatif, N. Mhaisen, A. Mohamed, and A. Erbad, *Slicing for AI: An Online Learning Framework for Network Slicing Supporting AI Services*, 2024. arXiv: 2411.02412 [cs.NI]. [Online]. Available: https://arxiv.org/abs/2411.02412.

[7] G. Castellano, F. Esposito, and F. Risso, "A Distributed Orchestration Algorithm for Edge Computing Resources with Guarantees," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 2548–2556. DOI: 10.1109/INFOCOM.2019.8737532.

[8] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "Holistic Network Virtualization and Pervasive Network Intelligence for 6G," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, pp. 1–30, 2022. DOI: 10.1109/COMST.2021.3135829.

[9] N. Reyhanian and Z.-Q. Luo, "Data-Driven Adaptive Network Slicing for Multi-Tenant Networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 1, pp. 113–128, 2022. DOI: 10.1109/JSTSP.2021.3127796.

[10] M. Alsenwi, N. H. Tran, M. Bennis, S. R. Pandey, A. K. Bairagi, and C. S. Hong, "Intelligent Resource Slicing for eMBB and URLLC Coexistence in 5G and Beyond: A Deep Reinforcement Learning Based Approach," *IEEE Transactions on Wireless Communications*, vol. 20, no. 7, pp. 4585–4600, 2021. DOI: 10.1109/TWC.2021.3060514.

[11] M. O. Ojijo and O. E. Falowo, "A Survey on Slice Admission Control Strategies and Optimization Schemes in 5G Network," *IEEE Access*, vol. 8, pp. 14 977–14 990, 2020. DOI: 10.1109/ACCESS.2020.2967626.

[12] M. Polese, N. Mohamadi, S. D'Oro, and T. Melodia, "Beyond Connectivity: An Open Architecture for AI-RAN Convergence in 6G," *arXiv preprint arXiv:2507.06911*, 2025.

[13] L. Yang, A. Zeynali, M. H. Hajiesmaili, R. K. Sitaraman, and D. Towsley, "Competitive Algorithms for Online Multidimensional Knapsack Problems," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 5, no. 3, pp. 1–30, 2021.

[14] S. Baxi, K. Cummings, A. Jacquillat, *et al.*, *Online Rack Placement in Large-Scale Data Centers*, 2025. arXiv: 2501.12725 [math.OC]. [Online]. Available: https://arxiv.org/abs/2501.12725.

[15] J. Ajayi, A. Di Maio, T. Braun, and D. Xenakis, "An Online Multi-dimensional Knapsack Approach for Slice Admission Control," in *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*, 2023, pp. 152–157. DOI: 10.1109/CCNC51644.2023.10060460.

[16] B. Khodapanah, A. Awada, I. Viering, J. Francis, M. Simsek, and G. P. Fettweis, "Radio Resource Management in context of Network Slicing: What is Missing in Existing Mechanisms?" In *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1–7. DOI: 10.1109/WCNC.2019.8885942.

[17] Z. Mlika and S. Cherkaoui, "Network Slicing with MEC and Deep Reinforcement Learning for the Internet of Vehicles," *IEEE Network*, vol. 35, no. 3, pp. 132–138, 2021. DOI: 10.1109/MNET.011.2000591.

[18] O. Besbes, Y. Gur, and A. Zeevi, "Non-Stationary Stochastic Optimization," *Oper. Res.*, vol. 63, no. 5, pp. 1227–1244, Oct. 2015, ISSN: 0030-364X.

[19] Y. Liu, Z. Liu, and Y. Yang, "Non-Stationary Stochastic Network Optimization with Imperfect Estimations," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 431–441. DOI: 10.1109/ICDCS.2019.00050.

[20] A. Jacquillat and M. L. Li, *Learning to Cover: Online Learning and Optimization with Irreversible Decisions*, 2024. arXiv: 2406.14777 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2406.14777.

[21] N. Chen, A. Agarwal, A. Wierman, S. Barman, and L. L. Andrew, "Online Convex Optimization Using Predictions," *SIGMETRICS Perform. Eval. Rev.*, vol. 43, no. 1, pp. 191–204, Jun. 2015, ISSN: 0163-5999. DOI: 10.1145/2796314.2745854. [Online]. Available: https://doi.org/10.1145/2796314.2745854.

[22] J. Comden, S. Yao, N. Chen, H. Xing, and Z. Liu, "Online Optimization in Cloud Resource Provisioning: Predictions, Regrets, and Algorithms," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 3, no. 1, Mar. 2019. DOI: 10.1145/3322205.3311087. [Online]. Available: https://doi.org/10.1145/3322205.3311087.

[23] Y.-F. Liu, T.-H. Chang, M. Hong, *et al.*, "A Survey of Recent Advances in Optimization Methods for Wireless Communications," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 11, pp. 2992–3031, 2024. DOI: 10.1109/JSAC.2024.3443759.

[24] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian Optimization of Machine Learning Algorithms," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25, Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/05311655a15b75fab86956663e1819cd-Paper.pdf.

[25] A. Nagabandi, C. Finn, and S. Levine, *Deep Online Learning via Meta-Learning: Continual Adaptation for Model-Based RL*, 2019. arXiv: 1812.07671 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1812.07671.

[26] J. J. A. Esteves, A. Boubendir, F. Guillemin, and P. Sens, "DRL-based Slice Placement under Realistic Network Load Conditions," in *2021 17th International Conference on Network and Service Management (CNSM)*, 2021, pp. 524–526. DOI: 10.23919/CNSM52442.2021.9615593.

[27] W. Chen, X. Qiu, T. Cai, H.-N. Dai, Z. Zheng, and Y. Zhang, "Deep Reinforcement Learning for Internet of Things: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1659–1692, 2021. DOI: 10.1109/COMST.2021.3073036.

[28] G. Kahn, A. Villaflor, V. Pong, P. Abbeel, and S. Levine, *Uncertainty-Aware Reinforcement Learning for Collision Avoidance*, 2017. arXiv: 1702.01182 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1702.01182.

[29] A. Jain, K. Khetarpal, and D. Precup, "Safe Option-Critic: Learning Dafety in the Option-Critic Architecture," *The Knowledge Engineering Review*, vol. 36, 2021, ISSN: 1469-8005. DOI: 10.1017/s0269888921000035. [Online]. Available: http://dx.doi.org/10.1017/S0269888921000035.

[30] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos, "Distributional Reinforcement Learning with Quantile Regression," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, ser. AAAI'18/IAAI'18/EAAI'18, New Orleans, Louisiana, USA: AAAI Press, 2018, ISBN: 978-1-57735-800-8.

[31] Q. Zhang, F. Liu, and C. Zeng, "Online Adaptive Interference-Aware VNF Deployment and Migration for 5G Network Slice," *IEEE/ACM Transactions on Networking*, vol. 29, no. 5, pp. 2115–2128, 2021. DOI: 10.1109/TNET.2021.3080197.

[32] X. Lin, D. Guo, Y. Shen, G. Tang, B. Ren, and M. Xu, "SFT-Box: An Online Approach for Minimizing the Embedding Cost of Multiple Hybrid SFCs," *IEEE/ACM Transactions on Networking*, vol. 31, no. 4, pp. 1463–1477, 2023. DOI: 10.1109/TNET.2022.3221868.

[33] T. Liu, S. Ni, X. Li, Y. Zhu, L. Kong, and Y. Yang, "Deep Reinforcement Learning Based Approach for Online Service Placement and Computation Resource Allocation in Edge

Computing," *IEEE Transactions on Mobile Computing*, vol. 22, no. 7, pp. 3870–3881, 2023. DOI: 10.1109/TMC.2022.3148254.

[34] B. Sonkoly, J. Czentye, M. Szalay, B. Németh, and L. Toka, "Survey on Placement Methods in the Edge and Beyond," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2590–2629, 2021. DOI: 10.1109/COMST.2021.3101460.

[35] S. Wu, N. Chen, A. Xiao, P. Zhang, C. Jiang, and W. Zhang, "AI-Empowered Virtual Network Embedding:A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, pp. 1–1, 2024. DOI: 10.1109/COMST.2024.3424533.

[36] A. He, H. Pan, Y. Dai, X. Si, C. Yuen, and Y. Zhang, "ADMM for Mobile Edge Intelligence: A Survey," *IEEE Communications Surveys & Tutorials*, pp. 1–1, 2024. DOI: 10.1109/COMST.2024.3521024.

[37] J. Wang, J. Liu, J. Li, and N. Kato, "Artificial Intelligence-Assisted Network Slicing: Network Assurance and Service Provisioning in 6G," *IEEE Vehicular Technology Magazine*, vol. 18, no. 1, pp. 49–58, 2023. DOI: 10.1109/MVT.2022.3228399.

[38] E. Coronado, R. Behravesh, T. Subramanya, *et al.*, "Zero Touch Management: A Survey of Network Automation Solutions for 5G and 6G Networks," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 4, pp. 2535–2578, 2022. DOI: 10.1109/COMST.2022.3212586.

[39] N. Saha, M. Zangooei, M. Golkarifard, and R. Boutaba, "Deep Reinforcement Learning Approaches to Network Slice Scaling and Placement: A Survey," *IEEE Communications Magazine*, vol. 61, no. 2, pp. 82–87, 2023. DOI: 10.1109/MCOM.006.2200534.

[40] M. Shetty, Y. Chen, G. Somashekar, *et al.*, "Building AI Agents for Autonomous Clouds: Challenges and Design Principles," in *Proceedings of the 2024 ACM Symposium on Cloud Computing*, ser. SoCC '24, Redmond, WA, USA: Association for Computing Machinery, 2024, pp. 99–110, ISBN: 9798400712869. DOI: 10.1145/3698038.3698525. [Online]. Available: https://doi.org/10.1145/3698038.3698525.

[41] S. Liu, F. Bronzino, P. Schmitt, *et al.*, "LEAF: Navigating Concept Drift in Cellular Networks," *Proceedings of the ACM on Networking*, vol. 1, no. CoNEXT2, pp. 1–24, 2023. DOI: 10.1145/3609422.

[42] K. V. Cardoso, C. B. Both, L. R. Prade, C. J. A. Macedo, and V. H. L. Lopes, *A Softwarized Perspective of the 5G Networks*, 2020. arXiv: 2006.10409 [cs.NI]. [Online]. Available: https://arxiv.org/abs/2006.10409.

[43] Y. Azimi, S. Yousefi, H. Kalbkhani, and T. Kunz, "Applications of Machine Learning in Resource Management for RAN-Slicing in 5G and Beyond Networks: A Survey," *IEEE Access*, vol. 10, pp. 106 581–106 612, 2022. DOI: 10.1109/ACCESS.2022.3210254.

[44] E. Schiller, J. Ajayi, S. Weber, T. Braun, and B. Stiller, "Toward a Live BBU Container Migration in Wireless Networks," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 301–321, 2022. DOI: 10.1109/OJCOMS.2022.3149965.

[45] Y. Li, Q. Zhang, H. Yao, R. Gao, X. Xin, and M. Guizani, "Next-Gen Service Function Chain Deployment: Combining Multi-Objective Optimization with AI Large Language Models," *IEEE Network*, pp. 1–1, 2025. DOI: 10.1109/MNET.2025.3532212.

[46] W. Kellerer, P. Kalmbach, A. Blenk, A. Basta, M. Reisslein, and S. Schmid, "Adaptable and Data-Driven Softwarized Networks: Review, Opportunities, and Challenges," *Proceedings of the IEEE*, vol. 107, no. 4, pp. 711–731, 2019. DOI: 10.1109/JPROC.2019.2895553.

[47] J. Xie, F. R. Yu, T. Huang, *et al.*, "A Survey of Machine Learning Techniques Applied to Software Defined Networking (SDN): Research Issues and Challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 393–430, 2019. DOI: 10.1109/COMST.2018.2866942.

[48] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A Survey on Mobile Edge Networks: Convergence of Computing, Caching and Communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017. DOI: 10.1109/ACCESS.2017.2685434.

[49] M. Gramaglia, P. Serrano, A. Banchs, G. Garcia-Aviles, A. Garcia-Saavedra, and R. Perez, "The Case for Serverless Mobile Networking," in *2020 IFIP Networking Conference (Networking)*, 2020, pp. 779–784.

[50] A. Satpathy, M. N. Sahoo, C. Swain, *et al.*, "Virtual Network Embedding: Literature Assessment, Recent Advancements, Opportunities, and Challenges," *IEEE Communications Surveys & Tutorials*, pp. 1–1, 2025. DOI: 10.1109/COMST.2025.3531724.

[51] N. Nikaein, E. Schiller, R. Favraud, R. Knopp, I. Alyafawi, and T. Braun, "Towards a Cloud-Native Radio Access Network," in *Advances in Mobile Cloud Computing and Big Data in the 5G Era*, C. X. Mavromoustakis, G. Mastorakis, and C. Dobre, Eds., Cham, Switzerland: Springer International Publishing, Nov. 2017, pp. 171–202, ISBN: 978-3-319-45145-9.

[52] A. M. Nagib, H. Abou-Zeid, and H. S. Hassanein, "SafeSlice: Enabling SLA-Compliant O-RAN Slicing via Safe Deep Reinforcement Learning," *arXiv preprint arXiv:2503.12753*, 2025.

[53] S. D. A. Shah, M. Hafeez, A. Salama, and S. A. R. Zaidi, "Proactive AI-and-RAN Workload Orchestration in O-RAN Architectures for 6G Networks," *arXiv preprint arXiv:2507.09124*, 2025.

[54] J. Groen, "Optimizing and Securing Open RAN with Experimental System Validation," Ph.D. dissertation, Northeastern University, 2024.

[55] M. Zhao, Y. Zhang, Q. Liu, A. Kak, and N. Choi, *AdaSlicing: Adaptive Online Network Slicing under Continual Network Dynamics in Open Radio Access Networks*, 2025. arXiv: 2501.06943 [cs.NI]. [Online]. Available: https://arxiv.org/abs/2501.06943.

[56] A. Tuerxun and A. Nakao, "SORA: Energy-Efficient Resource Allocation in Open Radio Access Network With Online Learning," *IEEE Access*, vol. 13, pp. 113 686–113 701, 2025. DOI: 10.1109/ACCESS.2025.3584345.

[57] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network Slicing and Softwarization: A Survey on Principles, Enabling Technologies, and Solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018. DOI: 10.1109/COMST.2018.2815638.

[58] H. Kim, "Design Principles for 5G Communications and Networks," in *Design and Optimization for 5G Wireless Communications*. John Wiley & Sons, Ltd, 2020, ch. 6, pp. 195–238, ISBN: 9781119494492. DOI: https://doi.org/10.1002/9781119494492.ch6. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119494492.ch6. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119494492.ch6.

[59] M. Karbalaee Motalleb, V. Shah-Mansouri, S. Parsaeefard, and O. L. Alcaraz López, "Resource Allocation in an Open RAN System Using Network Slicing," *IEEE Transactions on Network and Service Management*, vol. 20, no. 1, pp. 471–485, 2023. DOI: 10.1109/TNSM.2022.3205415.

[60] F. E. Subhan, A. Yaqoob, C. H. Muntean, and G.-M. Muntean, "A Survey on Artificial Intelligence Techniques for Improved Rich Media Content Delivery in a 5G and Beyond Network Slicing Context," *IEEE Communications Surveys & Tutorials*, 2024.

[61] A. Banchs, G. de Veciana, V. Sciancalepore, and X. Costa-Perez, "Resource Allocation for Network Slicing in Mobile Networks," *IEEE Access*, vol. 8, pp. 214 696–214 706, 2020. DOI: 10.1109/ACCESS.2020.3040949.

[62] K. Katsalis, N. Nikaein, E. Schiller, A. Ksentini, and T. Braun, "Network Slices Toward 5G Communications: Slicing the LTE Network," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 146–154, 2017.

[63] 3. T. 28.530, *Management and Orchestration; Concepts, Use cases and Requirements (Release 18) v18.0.0*, 2024.

[64] S. Ebrahimi, F. Bouali, and O. C. L. Haas, "Resource Management From Single-Domain 5G to End-to-End 6G Network Slicing: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 26, no. 4, pp. 2836–2866, 2024. DOI: 10.1109/COMST.2024.3390613.

[65] A. Machen, S. Wang, K. Leung, B. Ko, and T. Salonidis, "Live Service Migration in Mobile Edge Clouds," *IEEE Wireless Communications*, vol. 25, Feb. 2018. DOI: 10.1109/MWC.2017.1700011.

[66] S. Nunna, A. Kousaridas, M. Ibrahim, *et al.*, "Enabling real-time context-aware collaboration through 5g and mobile edge computing," in *2015 12th International Conference on Information Technology - New Generations*, 2015, pp. 601–605. DOI: 10.1109/ITNG.2015.155.

[67] F. Debbabi, R. Jmal, L. C. Fourati, and A. Ksentini, "Algorithmics and Modeling Aspects of Network Slicing in 5G and Beyonds Network: Survey," *IEEE Access*, vol. 8, pp. 162 748–162 762, 2020. DOI: 10.1109/ACCESS.2020.3022162.

[68] A. Du, J. Jia, S. Dustdar, J. Chen, and X. Wang, "Online Service Placement, Task Scheduling, and Resource Allocation in Hierarchical Collaborative MEC Systems," *IEEE Transactions on Services Computing*, no. 01, pp. 1–14, Jan. 5555, ISSN: 1939-1374. DOI: 10.1109/TSC.2025.3536307. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/TSC.2025.3536307.

[69] X. Zhang, Z. Huang, C. Wu, Z. Li, and F. C. M. Lau, "Online Auctions in IaaS Clouds: Welfare and Profit Maximization With Server Costs," *IEEE/ACM Transactions on Networking*, vol. 25, no. 2, pp. 1034–1047, 2017. DOI: 10.1109/TNET.2016.2619743.

[70] X. Tan, B. Sun, A. Leon-Garcia, Y. Wu, and D. H. Tsang, "Mechanism Design for Online Resource Allocation: A Unified Approach," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 4, no. 2, Jun. 2020. DOI: 10.1145/3392142. [Online]. Available: https://doi.org/10.1145/3392142.

[71] Z. Zhang, Z. Li, and C. Wu, "Optimal Posted Prices for Online Cloud Resource Allocation," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 1, Jun. 2017. DOI: 10.1145/3084460. [Online]. Available: https://doi.org/10.1145/3084460.

[72] Y. Cao, B. Sun, and D. H. K. Tsang, "Online Network Utility Maximization: Algorithm, Competitive Analysis, and Applications," *IEEE Transactions on Control of Network Systems*, vol. 10, no. 1, pp. 274–284, 2023. DOI: 10.1109/TCNS.2022.3199221.

[73] C. Stein, V.-A. Truong, and X. Wang, *Advance Service Reservations with Heterogeneous Customers*, 2018. arXiv: 1805.05554 [math.OC]. [Online]. Available: https://arxiv.org/abs/1805.05554.

[74] B. Sun, A. Zeynali, T. Li, M. Hajiesmaili, A. Wierman, and D. H. Tsang, "Competitive Algorithms for the Online Multiple Knapsack Problem with Application to Electric Vehicle Charging," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 4, no. 3, Nov. 2020. DOI: 10.1145/3428336. [Online]. Available: https://doi.org/10.1145/3428336.

[75] B. Sun, X. Tan, and D. H. K. Tsang, "Eliciting Multi-Dimensional Flexibilities From Electric Vehicles: A Mechanism Design Approach," *IEEE Transactions on Power Systems*, vol. 34, no. 5, pp. 4038–4047, 2019. DOI: 10.1109/TPWRS.2018.2856283.

[76] A. Borodin and R. El-Yaniv, *Online Computation and Competitive Analysis*. cambridge university press, 2005.

[77] B. Sun, L. Yang, M. Hajiesmaili, *et al.*, "The Online Knapsack Problem with Departures," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 6, no. 3, Dec. 2022. DOI: 10.1145/3570618. [Online]. Available: https://doi.org/10.1145/3570618.

[78] A. Lechowicz, N. Christianson, B. Sun, *et al.*, "CarbonClipper: Optimal Algorithms for Carbon-Aware Spatiotemporal Workload Management," *arXiv preprint arXiv:2408.07831*, 2024. [Online]. Available: http://arxiv.org/abs/2408.07831.

[79] S. Vassilaras, L. Gkatzikis, N. Liakopoulos, *et al.*, "The Algorithmic Aspects of Network Slicing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 112–119, 2017, ISSN: 01636804. DOI: 10.1109/MCOM.2017.1600939.

[80] R. Kumar, M. Purohit, and Z. Svitkina, "Improving Online Algorithms via ML Predictions," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 9684–9693.

[81] T. Zhao, W. Li, and A. Y. Zomaya, "Learning-Augmented Scheduling," *IEEE Transactions on Computers*, 2024.

[82] Y.-C. Liang, C. Stein, and H.-T. Wei, *Learning-Augmented Online Packet Scheduling with Deadlines*, 2024. arXiv: 2305.07164 [cs.DS]. [Online]. Available: https://arxiv.org/abs/2305.07164.

[83] R. Bostandoost, A. Lechowicz, W. A. Hanafy, N. Bashir, P. Shenoy, and M. Hajiesmaili, "LACS: Learning-Augmented Algorithms for Carbon-Aware Resource Scaling with

Uncertain Demand," in *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems*, 2024, pp. 27–45.

[84] P. Li, J. Yang, A. Wierman, and S. Ren, "Learning-Augmented Decentralized Online Convex Optimization in Networks," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 8, no. 3, Dec. 2024. DOI: 10.1145/3700420. [Online]. Available: https://doi.org/10.1145/3700420.

[85] H. Attou, A. Guezzaz, S. Benkirane, M. Azrour, and Y. Farhaoui, "Cloud-Based Intrusion Detection Approach Using Machine Learning Techniques," *Big Data Mining and Analytics*, vol. 6, no. 3, pp. 311–320, 2023. DOI: 10.26599/BDMA.2022.9020038.

[86] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," in *2010 IEEE Symposium on Security and Privacy*, 2010, pp. 305–316. DOI: 10.1109/SP.2010.25.

[87] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," *ACM Comput. Surv.*, vol. 41, no. 3, Jul. 2009, ISSN: 0360-0300. DOI: 10.1145/1541880.1541882. [Online]. Available: https://doi.org/10.1145/1541880.1541882.

[88] M. Ahmed, A. Naser Mahmood, and J. Hu, "A Survey of Network Anomaly Detection Rechniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016, ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2015.11.016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804515002891.

[89] V. C. Cunha, A. Z. Zavala, D. Magoni, P. R. M. Inácio, and M. M. Freire, "A Complete Review on the Application of Statistical Methods for Evaluating Internet Traffic Usage," *IEEE Access*, vol. 10, pp. 128 433–128 455, 2022. DOI: 10.1109/ACCESS.2022.3227073.

[90] R. Ning, C. Xin, and H. Wu, "TrojanFlow: A Neural Backdoor Attack to Deep Learning-based Network Traffic Classifiers," in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, 2022, pp. 1429–1438. DOI: 10.1109/INFOCOM48880.2022.9796878.

[91] A. Manousis, R. A. Sharma, V. Sekar, and J. Sherry, "Contention-Aware Performance Prediction For Virtualized Network Functions," in *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM '20, Virtual Event, USA: Association for Computing Machinery, 2020, pp. 270–282, ISBN: 9781450379557. DOI: 10.1145/3387514.3405868. [Online]. Available: https://doi.org/10.1145/3387514.3405868.

[92] S. Qiu, X. Cui, Z. Ping, *et al.*, "Deep Learning Techniques in Intelligent Fault Diagnosis and Prognosis for Industrial Systems: A Review," *Sensors*, vol. 23, no. 3, 2023, ISSN: 1424-8220. DOI: 10.3390/s23031305. [Online]. Available: https://www.mdpi.com/1424-8220/23/3/1305.

[93] J. B. Porch, C. Heng Foh, H. Farooq, and A. Imran, "Machine Learning Approach for Automatic Fault Detection and Diagnosis in Cellular Networks," in *2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, 2020, pp. 1–5. DOI: 10.1109/BlackSeaCom48709.2020.9234962.

[94] T. P. d. Silva, A. R. Neto, T. V. Batista, F. C. Delicato, P. F. Pires, and F. Lopes, "Online Machine Learning for Auto-Scaling in the Edge Computing," *Pervasive Mob. Comput.*, vol. 87, no. C, Dec. 2022, ISSN: 1574-1192. DOI: 10.1016/j.pmcj.2022.101722. [Online]. Available: https://doi.org/10.1016/j.pmcj.2022.101722.

[95] S. Klos née Müller, C. Tekin, M. van der Schaar, and A. Klein, "Context-Aware Hierarchical Online Learning for Performance Maximization in Mobile Crowdsourcing," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1334–1347, 2018. DOI: 10.1109/TNET.2018.2828415.

[96] G. Iosifidis, N. Mhaisen, and D. J. Leith, *Optimistic Learning for Communication Networks*, 2025. arXiv: 2504.03499 [cs.NI]. [Online]. Available: https://arxiv.org/abs/2504.03499.

[97] R. S. Sutton, A. G. Barto, *et al.*, *Reinforcement learning: An introduction*, vol. 1.

[98] A. Slivkins, *Introduction to Multi-Armed Bandits*, 2019. arXiv: 1904.07272 [cs.LG].

[99] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proceedings of the 19th International Conference on World Wide Web*, ser. WWW '10, Raleigh, North Carolina, USA: Association for Computing Machinery, 2010, pp. 661–670, ISBN: 9781605587998. DOI: 10.1145/1772690.1772758. [Online]. Available: https://doi.org/10.1145/1772690.1772758.

[100] D. Bouneffouf, I. Rish, and C. Aggarwal, "Survey on Applications of Multi-Armed and Contextual Bandits," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, 2020, pp. 1–8. DOI: 10.1109/CEC48606.2020.9185782.

[101] S. Balseiro, N. Golrezaei, M. Mahdian, V. Mirrokni, and J. Schneider, "Contextual Bandits with Cross-Learning," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/6aadca7bd86c4743e6724f9607256126-Paper.pdf.

[102] Y. Li, Z. Mu, and S. Qi, "A Contextual Combinatorial Bandit Approach to Negotiation," in *Proceedings of the 41st International Conference on Machine Learning*, ser. ICML'24, Vienna, Austria: JMLR.org, 2024.

[103] S. Wakayama and N. Ahmed, "Observation-Augmented Contextual Multi-Armed Bandits for Robotic Search and Exploration," *IEEE Robotics and Automation Letters*, vol. 9, no. 10, pp. 8531–8538, 2024. DOI: 10.1109/LRA.2024.3448133.

[104] J. A. Ayala-Romero, A. Garcia-Saavedra, and X. Costa-Perez, "Risk-Aware Continuous Control with Neural Contextual Bandits," in *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, ser. AAAI'24/IAAI'24/EAAI'24, AAAI Press, 2024, ISBN: 978-1-57735-887-9. DOI: 10.1609/aaai.v38i19.30083. [Online]. Available: https://doi.org/10.1609/aaai.v38i19.30083.

[105] F. Berkenkamp, "Safe Exploration in Reinforcement Learning: Theory and Applications in Robotics," en, Doctoral Thesis, ETH Zurich, Zurich, 2019. DOI: 10.3929/ethz-b-000370833.

[106] S. Henri, C. Vlachou, and P. Thiran, "Multi-Armed Bandit in Action: Optimizing Performance in Dynamic Hybrid Networks," *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1879–1892, 2018. DOI: 10.1109/TNET.2018.2856302.

[107] A. Verma and M. K. Hanawal, "Stochastic Network Utility Maximization with Unknown Utilities: Multi-Armed Bandits Approach," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, 2020, pp. 189–198. DOI: 10.1109/INFOCOM41043.2020.9155234.

[108] F. Orabona, *A Modern Introduction to Online Learning*, 2023. arXiv: 1912.13213 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1912.13213.

[109] A. Slivkins, *Introduction to Multi-Armed Bandits*, 2024. arXiv: 1904.07272 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1904.07272.

[110] A. Krause and J. Hübotter, *Probabilistic Artificial Intelligence*, 2025. arXiv: 2502.05244 [cs.AI]. [Online]. Available: https://arxiv.org/abs/2502.05244.

[111] Q. Liu, T. Han, N. Zhang, and Y. Wang, "DeepSlicing: Deep Reinforcement Learning Assisted Resource Allocation for Network Slicing," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6. DOI: 10.1109/GLOBECOM42002.2020.9322106.

[112] S. Bakri, B. Brik, and A. Ksentini, "On Using Reinforcement Learning for Network Slice Admission Control in 5G: Offline vs. Online," *International Journal of Communication Systems*, vol. 34, May 2021. DOI: 10.1002/dac.4757.

[113] S. Gholamipour, B. Akbari, N. Mokari, M. M. Tajiki, and E. A. Jorswieck, "Online Admission Control and Resource Allocation in Network Slicing under Demand Uncertainties," *CoRR*, vol. abs/2108.03710, 2021. arXiv: 2108.03710. [Online]. Available: https://arxiv.org/abs/2108.03710.

[114] V. Sciancalepore, L. Zanzi, X. Costa-Pérez, and A. Capone, "ONETS: Online Network Slice Broker From Theory to Practice," *IEEE Transactions on Wireless Communications*, vol. 21, no. 1, pp. 121–134, 2022. DOI: 10.1109/TWC.2021.3094116.

[115] J. X. Salvat, L. Zanzi, A. Garcia-Saavedra, V. Sciancalepore, and X. Costa-Perez, "Overbooking Network Slices through Yield-Driven End-to-End Orchestration," in *Proceedings of the 14th International Conference on Emerging Networking EXperiments and Technologies*, ser. CoNEXT '18, Heraklion, Greece: Association for Computing Machinery, 2018, pp. 353–365, ISBN: 9781450360807. DOI: 10.1145/3281411.3281435. [Online]. Available: https://doi.org/10.1145/3281411.3281435.

[116] K. Noroozi, M. Karimzadeh-Farshbafan, and V. Shah-Mansouri, "Service Admission Control for 5G Mobile Networks with RAN and Core Slicing," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6. DOI: 10.1109/GLOBECOM38437.2019.9013617.

[117] S. Chen, L. Wang, and F. Liu, "Optimal Admission Control Mechanism Design for Time-Sensitive Services in Edge Computing," in *IEEE INFOCOM 2022*

- *IEEE Conference on Computer Communications*, 2022, pp. 1169–1178. DOI: 10.1109/INFOCOM48880.2022.9796847.

[118] Q.-T. Luu, S. Kerboeuf, and M. Kieffer, "Admission Control and Resource Reservation for Prioritized Slice Requests with Guaranteed SLA under Uncertainties," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022. DOI: 10.1109/TNSM.2022.3160352.

[119] J. Leguay, L. Maggi, M. Draief, S. Paris, and S. Chouvardas, "Admission Control with Online Algorithms in SDN," in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, 2016, pp. 718–721. DOI: 10.1109/NOMS.2016.7502884.

[120] M. Vincenzi, E. Lopez-Aguilera, and E. Garcia-Villegas, "Timely Admission Control for Network Slicing in 5G With Machine Learning," *IEEE Access*, vol. 9, pp. 127 595–127 610, 2021. DOI: 10.1109/ACCESS.2021.3111143.

[121] S. Lindståhl, A. Proutiere, and A. Johnsson, *Measurement-based Admission Control in Sliced Networks: A Best Arm Identification Approach*, 2022. DOI: 10.48550/ARXIV.2204.06910. [Online]. Available: https://arxiv.org/abs/2204.06910.

[122] R. Prasad and A. Sunny, "Scheduling Slice Requests in 5G Networks," *IEEE/ACM Transactions on Networking*, pp. 1–12, 2023. DOI: 10.1109/TNET.2023.3274480.

[123] M. Sulaiman, A. Moayyedi, M. A. Salahuddin, R. Boutaba, and A. Saleh, "Multi-Agent Deep Reinforcement Learning for Slicing and Admission Control in 5G C-RAN," in *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, 2022, pp. 1–9. DOI: 10.1109/NOMS54207.2022.9789903.

[124] M. Sulaiman, A. Moayyedi, M. Ahmadi, M. A. Salahuddin, R. Boutaba, and A. Saleh, "Coordinated Slicing and Admission Control using Multi-Agent Deep Reinforcement Learning," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022. DOI: 10.1109/TNSM.2022.3222589.

[125] H. Yi, Q. Lin, and M. Chen, "Balancing Cost and Dissatisfaction in Online EV Charging under Real-time Pricing," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 1801–1809. DOI: 10.1109/INFOCOM.2019.8737558.

[126] W. Jiang, Y. Zhan, G. Zeng, and J. Lu, "Probabilistic-Forecasting-Based Admission Control for Network Slicing in Software-Defined Networks," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 14 030–14 047, 2022. DOI: 10.1109/JIOT.2022.3145475.

[127] M. A. Haque and V. Kirova, "5G Network Slice Admission Control Using Optimization and Reinforcement Learning," in *2022 IEEE Wireless Communications and Networking Conference (WCNC)*, 2022, pp. 854–859. DOI: 10.1109/WCNC51071.2022.9771643.

[128] B. Bakhshi, J. Mangues-Bafalluy, and J. Baranda, "R-Learning-Based Admission Control for Service Federation in Multi-domain 5G Networks," in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 1–6. DOI: 10.1109/GLOBECOM46510.2021.9685936.

[129] M. Kalntis, G. Iosifidis, and F. A. Kuipers, *Adaptive Resource Allocation for Virtualized Base Stations in O-RAN with Online Learning*, 2023. arXiv: 2309.01730 [cs.NI]. [Online]. Available: https://arxiv.org/pdf/2309.01730.pdf.

[130] T. Dong, Q. Qi, J. Wang, *et al.*, "Standing on the Shoulders of Giants: Cross-Slice Federated Meta Learning for Resource Orchestration to Cold-Start Slice," *IEEE/ACM Transactions on Networking*, pp. 1–18, 2022. DOI: 10.1109/TNET.2022.3200853.

[131] S. Hashima, Z. M. Fadlullah, M. M. Fouda, *et al.*, "On Softwarization of Intelligence in 6G Networks for Ultra-Fast Optimal Policy Selection: Challenges and Opportunities," *IEEE Network*, vol. 37, no. 2, pp. 190–197, 2023. DOI: 10.1109/MNET.103.2100587.

[132] O. T. Ajayi, X. Cao, H. Shan, and Y. Cheng, "Self-Renewal Machine Learning Approach for Fast Wireless Network Optimization," in *2023 IEEE 20th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, 2023, pp. 134–142. DOI: 10.1109/MASS58611.2023.00024.

[133] N. Kato, B. Mao, F. Tang, Y. Kawamoto, and J. Liu, "Ten Challenges in Advancing Machine Learning Technologies toward 6G," *IEEE Wireless Communications*, vol. 27, no. 3, pp. 96–103, 2020. DOI: 10.1109/MWC.001.1900476.

[134] A. Dietmüller, R. Jacob, and L. Vanbever, "On Sample Selection for Continual Learning: a Video Streaming Case Study," *ACM SIGCOMM Computer Communication Review*, vol. 54, no. 2, pp. 10–35, 2024.

[135] S. Lahmer, F. Mason, F. Chiariotti, and A. Zanella, "Fast Context Adaptation in Cost-Aware Continual Learning," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 2, pp. 479–494, 2024. DOI: 10.1109/TMLCN.2024.3386647.

[136] S. C. Hoi, D. Sahoo, J. Lu, and P. Zhao, "Online Learning: A Comprehensive Survey," *Neurocomput.*, vol. 459, no. C, pp. 249–289, Oct. 2021, ISSN: 0925-2312. DOI: 10.1016/j.neucom.2021.04.112. [Online]. Available: https://doi.org/10.1016/j.neucom.2021.04.112.

[137] I. Juneja, S. Fatale, and S. Moharir, *Correlated Age-of-Information Bandits*, 2020. arXiv: 2011.05032 [eess.SY].

[138] O. Besbes, Y. Gur, and A. Zeevi, "Stochastic Multi-Armed-Bandit Problem with Non-stationary Rewards," in *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, Dec. 2014. DOI: 10.13140/RG.2.1.2862.3844.

[139] X. Zhang, J. Zuo, Z. Huang, Z. Zhou, X. Chen, and C. Joe-Wong, "Learning with Side Information: Elastic Multi-resource Control for the Open RAN," *IEEE Journal on Selected Areas in Communications*, 2023.

[140] S. Boldrini, L. De Nardis, G. Caso, M. T. P. Le, J. Fiorina, and M.-G. Di Benedetto, "muMAB: A Multi-Armed Bandit Model for Wireless Network Selection," *Algorithms*, vol. 11, no. 2, 2018, ISSN: 1999-4893. DOI: 10.3390/a11020013. [Online]. Available: https://www.mdpi.com/1999-4893/11/2/13.

[141] R. Kerkouche, R. Alami, R. Féraud, N. Varsier, and P. Maillé, "Node-based Optimization of LoRa Transmissions with Multi-Armed Bandit Algorithms," in *2018 25th International Conference on Telecommunications (ICT)*, 2018, pp. 521–526. DOI: 10.1109/ICT.2018.8464949.

[142] A. Alzadjali, F. Esposito, and J. Deogun, "A Contextual Bi-armed Bandit Approach for MPTCP Path Management in Heterogeneous LTE and WiFi Edge Networks," in *2020*

IEEE/ACM Symposium on Edge Computing (SEC), 2020, pp. 307–316. DOI: 10.1109/SEC50012.2020.00042.

[143] M. Kalntis, A. Lutu, J. O. Iglesias, F. A. Kuipers, and G. Iosifidis, *Smooth Handovers via Smoothed Online Learning*, 2025. arXiv: 2501.08099 [cs.NI]. [Online]. Available: https://arxiv.org/abs/2501.08099.

[144] I. Bistritz and N. Bambos, "Gamekeeper: Online Learning for Admission Control of Networked Open Multiagent Systems," *IEEE Transactions on Automatic Control*, vol. 69, no. 11, pp. 7694–7709, 2024. DOI: 10.1109/TAC.2024.3398139.

[145] J.-B. Monteil, G. Iosifidis, and L. Da Silva, "Learning-based Reservation of Virtualized Network Resources," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022. DOI: 10.1109/TNSM.2022.3144774.

[146] M. Kalntis, G. Iosifidis, and F. A. Kuipers, "Adaptive Resource Allocation for Virtualized Base Stations in O-RAN with Online Learning," *IEEE Transactions on Communications*, pp. 1–1, 2024. DOI: 10.1109/TCOMM.2024.3461569.

[147] Q. Liu, N. Choi, and T. Han, "OnSlicing: Online End-to-End Network Slicing with Reinforcement Learning," in *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies*, ser. CoNEXT '21, Virtual Event, Germany: Association for Computing Machinery, 2021, pp. 141–153, ISBN: 9781450390989. DOI: 10.1145/3485983.3494850. [Online]. Available: https://doi.org/10.1145/3485983.3494850.

[148] Q. Liu, N. Choi, and T. Han, "Atlas: Automate Online Service Configuration in Network Slicing," in *Proceedings of the 18th International Conference on Emerging Networking EXperiments and Technologies*, ser. CoNEXT '22, Roma, Italy: Association for Computing Machinery, 2022, pp. 140–155, ISBN: 9781450395083. DOI: 10.1145/3555050.3569115. [Online]. Available: https://doi.org/10.1145/3555050.3569115.

[149] Y. Xia, F. Kong, T. Yu, *et al.*, "Which LLM to Play? Convergence-Aware Online Model Selection with Time-Increasing Bandits," in *Proceedings of the ACM Web Conference 2024*, ser. WWW '24, Singapore, Singapore: Association for Computing Machinery, 2024, pp. 4059–4070, ISBN: 9798400701719. DOI: 10.1145/3589334.3645420. [Online]. Available: https://doi.org/10.1145/3589334.3645420.

[150] A. Tornede, M. Wever, and E. Hüllermeier, *Towards Meta-Algorithm Selection*, 2020. arXiv: 2011.08784 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2011.08784.

[151] A. N. Elmachtoub and P. Grigas, "Smart "Predict, then Optimize"," *Management Science*, vol. 68, no. 1, pp. 9–26, 2022. DOI: 10.1287/mnsc.2020.3922. eprint: https://doi.org/10.1287/mnsc.2020.3922. [Online]. Available: https://doi.org/10.1287/mnsc.2020.3922.

[152] S. Adriaensen, A. Biedenkapp, G. Shala, *et al.*, *Automated Dynamic Algorithm Configuration*, 2022. arXiv: 2205.13881 [cs.AI]. [Online]. Available: https://arxiv.org/abs/2205.13881.

[153] R. Gupta and T. Roughgarden, "Data-driven Algorithm Design," *Commun. ACM*, vol. 63, no. 6, pp. 87–94, May 2020, ISSN: 0001-0782. DOI: 10.1145/3394625. [Online]. Available: https://doi.org/10.1145/3394625.

[154] C. Dann, C. Gentile, and A. Pacchiano, "Data-Driven Online Model Selection With Regret Guarantees," in *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, S. Dasgupta, S. Mandt, and Y. Li, Eds., ser. Proceedings of Machine Learning Research, vol. 238, PMLR, Feb. 2024, pp. 1531–1539. [Online]. Available: https://proceedings.mlr.press/v238/dann24a.html.

[155] G. Pettet, A. Mukhopadhyay, M. J. Kochenderfer, and A. Dubey, "Hierarchical Planning for Dynamic Resource Allocation in Smart and Connected Communities," *ACM Trans. Cyber-Phys. Syst.*, vol. 6, no. 4, Nov. 2022, ISSN: 2378-962X. DOI: 10.1145/3502869. [Online]. Available: https://doi.org/10.1145/3502869.

[156] G. Pettet, A. Mukhopadhyay, M. J. Kochenderfer, and A. Dubey, "Hierarchical Planning for Dynamic Resource Allocation in Smart and Connected Communities," *ACM Transactions on Cyber-Physical Systems*, vol. 6, no. 4, pp. 1–26, 2022.

[157] F. Wei, G. Feng, S. Qin, Y. Peng, and Y. Liu, "Hierarchical Network Slicing for UAV-Assisted Wireless Networks With Deployment Optimization," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 12, pp. 3705–3718, 2024. DOI: 10.1109/JSAC.2024.3459055.

[158] Y. Sun, M. Peng, S. Mao, and S. Yan, "Hierarchical Radio Resource Allocation for Network Slicing in Fog Radio Access Networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3866–3881, 2019. DOI: 10.1109/TVT.2019.2896586.

[159] X. Chen, C. Wu, Z. Zhao, Y. Xiao, S. Mao, and Y. Ji, "Hierarchical Meta-Reinforcement Learning for Resource-Efficient Slicing in O-RAN," in *GLOBECOM 2023 - 2023 IEEE Global Communications Conference*, 2023, pp. 2729–2735. DOI: 10.1109/GLOBECOM54140.2023.10437350.

[160] K. Qiao, H. Wang, W. Zhang, D. Yang, Y. Zhang, and N. Zhang, "Resource Allocation for Network Slicing in Open RAN: A Hierarchical Learning Approach," *IEEE Transactions on Cognitive Communications and Networking*, pp. 1–1, 2025. DOI: 10.1109/TCCN.2024.3524641.

[161] M. A. Habib, H. Zhou, P. E. Iturria-Rivera, *et al.*, "Machine Learning-Enabled Traffic Steering in O-RAN: A Case Study on Hierarchical Learning Approach," *IEEE Communications Magazine*, vol. 63, no. 1, pp. 100–107, 2025. DOI: 10.1109/MCOM.002.2300526.

[162] Y. Mao, X. Shang, and Y. Yang, "Ant Colony based Online Learning Algorithm for Service Function Chain Deployment," in *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications*, Aug. 2023, pp. 1–10. DOI: 10.1109/INFOCOM53939.2023.10229012.

[163] Z. Wu, G. Ishigaki, R. Gour, *et al.*, "Reinforcement Learning-Based Multi-Domain Network Slice Provisioning," in *ICC 2023 - IEEE International Conference on Communications*, 2023, pp. 1899–1904. DOI: 10.1109/ICC45041.2023.10278745.

[164] H. Hojeij, M. Sharara, S. Hoteit, and V. Vèque, "Dynamic Placement of O-CU and O-DU Functionalities in Open-RAN Architecture," in *2023 20th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, 2023, pp. 330–338. DOI: 10.1109/SECON58729.2023.10287529.

[165] Y. Yue, X. Tang, W. Yang, C. Cao, and Z. Zhang, "A Deep Learning-Based VNF Placement Approach for SFC Requests in MEC-NFV Enabled Networks," in *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. New York, NY, USA: Association for Computing Machinery, 2023, ISBN: 9781450399906. [Online]. Available: https://doi.org/10.1145/3570361.3615742.

[166] Y. Bi, C. C. Meixner, M. Bunyakitanon, X. Vasilakos, R. Nejabati, and D. Simeonidou, "Multi-Objective Deep Reinforcement Learning Assisted Service Function Chains Placement," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4134–4150, 2021. DOI: 10.1109/TNSM.2021.3127685.

[167] L. Chen, J. Xu, S. Ren, and P. Zhou, "Spatio–Temporal Edge Service Placement: A Bandit Learning Approach," *IEEE Transactions on Wireless Communications*, vol. 17, no. 12, pp. 8388–8401, 2018. DOI: 10.1109/TWC.2018.2876823.

[168] L. Li, J. Shen, B. Wu, Y. Zhou, X. Wang, and K. Li, "Adaptive Data Placement in Multi-Cloud Storage: A Non-Stationary Combinatorial Bandit Approach," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 11, pp. 2843–2859, 2023. DOI: 10.1109/TPDS.2023.3306150.

[169] T. D. Tran, B. Jaumard, Q. H. Duong, and K.-K. Nguyen, "5G Service Function Chain Provisioning: A Deep Reinforcement Learning-Based Framework," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2024. DOI: 10.1109/TNSM.2024.3438438.

[170] H.-K. Lim, I. Ullah, J.-B. Kim, and Y.-H. Han, "Virtual Network Embedding Based on Hierarchical Cooperative Multiagent Reinforcement Learning," *IEEE Internet of Things Journal*, 2023.

[171] J. Ajayi, A. Di Maio, and T. Braun, "Hierarchical Placement Learning for Network Slice Provisioning," in *2025 IEEE 50th Conference on Local Computer Networks (LCN) [Accepted]*, IEEE, 2025.

[172] S. Pateria, B. Subagdja, A.-h. Tan, and C. Quek, "Hierarchical Reinforcement Learning: A Comprehensive Survey," *ACM Comput. Surv.*, vol. 54, no. 5, Jun. 2021, ISSN: 0360-0300. DOI: 10.1145/3453160. [Online]. Available: https://doi.org/10.1145/3453160.

[173] M. Leconte, G. Paschos, P. Mertikopoulos, and U. Kozat, "A Resource Allocation Framework for Network Slicing," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, Apr. 2018, pp. 2177–2185. DOI: 10.1109/INFOCOM.2018.8486303.

[174] H. Chien, Y. Lin, C. Lai, and C. Wang, "End-to-End Slicing as a Service with Computing and Communication Resource Allocation for Multi-Tenant 5G Systems," *IEEE Wireless Communications*, vol. 26, no. 5, pp. 104–112, 2019. DOI: 10.1109/MWC.2019.1800466.

[175] Q. Liu, T. Han, and N. Ansari, "Joint Radio and Computation Resource Management for Low Latency Mobile Edge Computing," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–7. DOI: 10.1109/GLOCOM.2018.8647792.

[176] H. Mahmoud, M. N. I. Farooqui, D. Mi, *et al.*, "Data-driven Approach for Optimising Resource Allocation of O-RAN Networks," in *2024 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2024, pp. 1–6.

[177] S.-p. Yeh, S. Bhattacharya, R. Sharma, and H. Moustafa, "Deep Learning for Intelligent and Automated Network Slicing in 5G Open RAN (ORAN) Deployment," *IEEE Open Journal of the Communications Society*, 2023.

[178] F. Wang, M. C. Gursoy, and S. Velipasalar, "Robust Network Slicing: Multi-Agent Policies, Adversarial Attacks, and Defensive Strategies," *IEEE Transactions on Machine Learning in Communications and Networking*, 2023.

[179] A. Kak and I. F. Akyildiz, "Towards Automatic Network Slicing for the Internet of Space Things," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 392–412, 2021.

[180] S. Mohanti, A. Malhotra, M. Umar, B. Farooq, and J. Chandrashekar, "PROMPT : Prediction of Channel Metrics for Proactive Optimization in Cellular Networks," *2024 IEEE 25th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp. 370–376, 2024. DOI: 10.1109/WoWMoM60985.2024.00065.

[181] F. Wei, G. Feng, Y. Sun, Y. Wang, and S. Qin, "Proactive Network Slice Reconfiguration by Exploiting Prediction Interval and Robust Optimization," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*, IEEE, 2020, pp. 1–6.

[182] A. Balasingam, M. Kotaru, and P. Bahl, "{Application-Level} Service Assurance with 5G {RAN} Slicing," in *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*, 2024, pp. 841–857.

[183] A. Filali, Z. Mlika, and S. Cherkaoui, "Open RAN Slicing for MVNOs With Deep Reinforcement Learning," *IEEE Internet of Things Journal*, 2024.

[184] E. Amiri, N. Wang, M. Shojafar, M. Q. Hamdan, C. H. Foh, and R. Tafazolli, "Deep Reinforcement Learning for Robust VNF Reconfigurations in O-RAN," *IEEE Transactions on Network and Service Management*, 2023.

[185] N. Van Huynh, D. Thai Hoang, D. N. Nguyen, and E. Dutkiewicz, "Optimal and Fast Real-Time Resource Slicing With Deep Dueling Neural Networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1455–1470, 2019. DOI: 10.1109/JSAC.2019.2904371.

[186] J. S. Pujol Roig, D. M. Gutierrez-Estevez, and D. Gündüz, "Management and Orchestration of Virtual Network Functions via Deep Reinforcement Learning," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 304–317, 2020. DOI: 10.1109/JSAC.2019.2959263.

[187] L. A. Garrido, A. Dalgkitsis, K. Ramantas, A. Ksentini, and C. Verikoukis, "Resource Demand Prediction for Network Slices in 5G using ML Enhanced with Network Models," *IEEE Transactions on Vehicular Technology*, 2024.

[188] A. Bura, U. Ghosh, D. Bharadia, and S. Shakkottai, "Windex: Realtime Neural Whittle Indexing for Scalable Service Guarantees in NextG Cellular Networks," *arXiv preprint arXiv:2406.01888*, 2024.

[189] I. Alawe, A. Ksentini, Y. Hadjadj-Aoul, and P. Bertin, "Improving Traffic Forecasting for 5G Core Network Scalability: A Machine Learning Approach," *IEEE Network*, vol. 32, no. 6, pp. 42–49, 2018.

[190] R. Challa, V. V. Zalyubovskiy, S. M. Raza, H. Choo, and A. De, "Network Slice Admission Model: Tradeoff Between Monetization and Rejections," *IEEE Systems Journal*, vol. 14, no. 1, pp. 657–660, 2020. DOI: 10.1109/JSYST.2019.2904667.

[191] B. Han, D. Feng, and H. D. Schotten, "A Markov Model of Slice Admission Control," *IEEE Networking Letters*, vol. 1, no. 1, pp. 2–5, 2019. DOI: 10.1109/LNET.2018.2873978.

[192] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, and A. Banchs, "Mobile Traffic Forecasting for Maximizing 5G Network Slicing Resource Utilization," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017, pp. 1–9. DOI: 10.1109/INFOCOM.2017.8057230.

[193] L. Yang, A. Zeynali, M. H. Hajiesmaili, R. K. Sitaraman, and D. Towsley, "Competitive Algorithms for Online Multidimensional Knapsack Problems," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 5, pp. 1–30, 2021.

[194] B. Han, V. Sciancalepore, D. Feng, X. Costa-Perez, and H. D. Schotten, "A Utility-Driven Multi-Queue Admission Control Solution for Network Slicing," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 55–63. DOI: 10.1109/INFOCOM.2019.8737517.

[195] J. Ajayi, A. Di Maio, and T. Braun, "Drift-Aware Policy Selection for Slice Admission Control," in *NOMS 2024-2024 IEEE Network Operations and Management Symposium*, 2024, pp. 1–9. DOI: 10.1109/NOMS59830.2024.10575766.

[196] L. Cominardi, T. Deiss, M. Filippou, V. Sciancalepore, F. Giust, and D. Sabella, "MEC Support for Network Slicing: Status and Limitations from a Standardization Viewpoint," *IEEE Communications Standards Magazine*, vol. 4, no. 2, pp. 22–30, 2020. DOI: 10.1109/MCOMSTD.001.1900046.

[197] Y. Lin, J. A. Preiss, E. Anand, Y. Li, Y. Yue, and A. Wierman, "Online Adaptive Policy Selection in Time-Varying Systems: No-Regret via Contractive Perturbations," in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36, Curran Associates, Inc., 2023, pp. 53 508–53 521. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2023/file/a7a7180fe7f82ff98eee0827c5e9c141-Paper-Conference.pdf.

[198] M. Gagliolo and J. Schmidhuber, "Learning Restart Strategies," in *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, ser. IJCAI'07, Hyderabad, India: Morgan Kaufmann Publishers Inc., 2007, pp. 792–797.

[199] Y. Liu, J. Comden, Z. Liu, and Y. Yang, "Online Resource Provisioning for Wireless Data Collection," *ACM Trans. Sen. Netw.*, vol. 18, no. 1, Oct. 2021, ISSN: 1550-4859. DOI: 10.1145/3470648. [Online]. Available: https://doi.org/10.1145/3470648.

[200] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time Analysis of the Multi-armed Bandit Problem," *Machine Learning*, vol. 47, pp. 235–256, May 2002. DOI: 10.1023/A:1013689704352.

[201] E. V. Belmega, P. Mertikopoulos, R. Negrel, and L. Sanguinetti, *Online Convex Optimization and No-Regret Learning: Algorithms, Guarantees and Applications*, 2018. arXiv: 1804.04529 [cs.LG].

[202] A. "Garivier and E. Moulines, "On Upper-Confidence Bound Policies for Switching Bandit Problems," in *Algorithmic Learning Theory*, J. Kivinen, C. Szepesvári, E. Ukkonen, and T. Zeugmann, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, 174–188, ISBN: 978-3-642-24412-4.

[203] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under Concept Drift: A Review," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346–2363, 2019. DOI: 10.1109/TKDE.2018.2876857.

[204] A. Bifet and R. Gavaldà, "Learning from Time-Changing Data with Adaptive Windowing," vol. 7, Apr. 2007. DOI: 10.1137/1.9781611972771.42.

[205] L. Baier, T. Schlör, J. Schöffer, and N. Kühl, *Detecting Concept Drift With Neural Network Model Uncertainty*, 2022. arXiv: 2107.01873 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2107.01873.

[206] Y. Dai and L. Huang, "Adversarial Network Optimization under Bandit Feedback: Maximizing Utility in Non-Stationary Multi-Hop Networks," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 8, no. 3, Dec. 2024. DOI: 10.1145/3700413. [Online]. Available: https://doi.org/10.1145/3700413.

[207] W. Attaoui, E. Sabir, H. Elbiaze, and M. Guizani, "VNF and CNF Placement in 5G: Recent Advances and Future Trends," *IEEE Transactions on Network and Service Management*, vol. 20, no. 4, pp. 4698–4733, 2023. DOI: 10.1109/TNSM.2023.3264005.

[208] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5G Network Slicing using SDN and NFV: A Survey of Taxonomy, Architectures and Future Challenges," *Computer Networks*, vol. 167, p. 106 984, 2020, ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2019.106984. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128619304773.

[209] H. P. Phyu, D. Naboulsi, and R. Stanica, "Machine Learning in Network Slicing—A Survey," *IEEE Access*, vol. 11, pp. 39 123–39 153, 2023. DOI: 10.1109/ACCESS.2023.3267985.

[210] W. da Silva Coelho, A. Benhamiche, N. Perrot, and S. Secci, "Function Splitting, Isolation, and Placement Trade-Offs in Network Slicing," *IEEE Transactions on Network and Service Management*, vol. 19, no. 2, pp. 1920–1936, 2022. DOI: 10.1109/TNSM.2021.3130915.

[211] Q. Liu, N. Choi, and T. Han, "Deep Reinforcement Learning for End-to-End Network Slicing: Challenges and Solutions," *IEEE Network*, vol. 37, no. 2, pp. 222–228, 2023. DOI: 10.1109/MNET.113.2100739.

[212] S. Asakipaam, K. Jerry, and K. Oteng Gyasi, "Resource Provisioning and Utilization in 5G Network Slicing: A Survey of Recent Advances, Challenges, and Open Issues," *International Journal of Computer Networks And Applications*, vol. 10, pp. 201–216, May 2023. DOI: 10.22247/ijcna/2023/220736.

[213] X. Li, C. Guo, L. Gupta, and R. Jain, "Efficient and Secure 5G Core Network Slice Provisioning Based on VIKOR Approach," *IEEE Access*, vol. 7, pp. 150 517–150 529, 2019. DOI: 10.1109/ACCESS.2019.2947454.

[214] H. P. Phyu, R. Stanica, and D. Naboulsi, "Multi-Slice Privacy-Aware Traffic Forecasting at RAN Level: A Scalable Federated-Learning Approach," *IEEE Transactions on Network and Service Management*, vol. 20, no. 4, pp. 5038–5052, 2023. DOI: 10.1109/TNSM.2023.3267725.

[215] R. Han, J. Wang, Q. Qi, *et al.*, "Dynamic Network Slice for Bursty Edge Traffic," *IEEE/ACM Transactions on Networking*, pp. 1–16, 2024. DOI: 10.1109/TNET.2024.3376794.

[216] Q. Zhang, F. Liu, and C. Zeng, "Adaptive Interference-Aware VNF Placement for Service-Customized 5G Network Slices," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 2449–2457. DOI: 10.1109/INFOCOM.2019.8737660.

[217] R. Chen, W. Peng, Y. Li, X. Liu, and G. Wang, "Orchid: An Online Learning based Resource Partitioning Framework for Job Colocation with Multiple Objectives," *IEEE Transactions on Computers*, vol. PP, pp. 1–14, Jan. 2023. DOI: 10.1109/TC.2023.3303959.

[218] Y. Hu, C. Zhang, E. Andert, *et al.*, "GiPH: Generalizable Placement Learning for Adaptive Heterogeneous Computing," *Proceedings of Machine Learning and Systems*, vol. 5, 2023.

[219] Z. Li, L. Zheng, Y. Zhong, *et al.*, "{AlpaServe}: Statistical Multiplexing with Model Parallelism for Deep Learning Serving," in *17th USENIX Symposium on Operating Systems Design and Implementation (OSDI 23)*, 2023, pp. 663–679.

[220] C. Zheng, H. Tang, M. Zang, *et al.*, "DINC: Toward Distributed In-Network Computing," *Proc. ACM Netw.*, vol. 1, no. CoNEXT3, Nov. 2023. DOI: 10.1145/3629136. [Online]. Available: https://doi.org/10.1145/3629136.

[221] C. Tian, H. Cao, J. Xie, S. Garg, M. Alrashoud, and P. Tiwari, "Community Detection-Empowered Self-Adaptive Network Slicing in Multi-Tier Edge-Cloud System," *IEEE Transactions on Network and Service Management*, vol. 21, no. 3, pp. 2624–2636, 2024. DOI: 10.1109/TNSM.2023.3332509.

[222] Z. Xu, M. Yu, and F. Y. Yan, "Decouple and Decompose: Scaling Resource Allocation with {DeDe}," in *19th USENIX Symposium on Operating Systems Design and Implementation (OSDI 25)*, 2025, pp. 393–409.

[223] Z. Wang, P. Li, C.-J. M. Liang, F. Wu, and F. Y. Yan, "Autothrottle: A Practical Bi-Level Approach to Resource Management for SLO-Targeted Microservices," in *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*, Santa Clara, CA: USENIX Association, Apr. 2024, pp. 149–165, ISBN: 978-1-939133-39-7. [Online]. Available: https://www.usenix.org/conference/nsdi24/presentation/wang-zibo.

[224] E. Even-dar, S. M. Kakade, and Y. Mansour, "Experts in a Markov Decision Process," in *Advances in Neural Information Processing Systems*, L. Saul, Y. Weiss, and L. Bottou, Eds., vol. 17, MIT Press, 2004.

[225] Q. Guo, S. Wang, and J. Zhu, *Regret Analysis for Hierarchical Experts Bandit Problem*, 2022. arXiv: 2208.05622 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2208.05622.

[226] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast Unfolding of Communities in Large Networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, P10008, Oct. 2008. DOI: 10.1088/1742-5468/2008/10/P10008. [Online]. Available: https://dx.doi.org/10.1088/1742-5468/2008/10/P10008.

[227] A. Sivagnanam, A. Pettet, H. Lee, A. Mukhopadhyay, A. Dubey, and A. Laszka, "Multi-Agent Reinforcement Learning with Hierarchical Coordination for Emergency Responder Stationing," in *Proceedings of the 41st International Conference on Machine Learning*, ser. ICML'24, Vienna, Austria: JMLR.org, 2024.

[228] R. Mehrotra, N. Xue, and M. Lalmas, "Bandit Based Optimization of Multiple Objectives on a Music Streaming Platform," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '20, Virtual Event, CA, USA: Association for Computing Machinery, 2020, pp. 3224–3233, ISBN: 9781450379984. DOI: 10.1145/3394486.3403374. [Online]. Available: https://doi.org/10.1145/3394486.3403374.

[229] D. Öner, A. Karakurt, A. Eryılmaz, and C. Tekin, *Combinatorial Multi-Objective Multi-Armed Bandit Problem*, 2018. arXiv: 1803.04039 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1803.04039.

[230] G. Budel and P. Van Mieghem, "Detecting The Number of Clusters in a Network," *Journal of Complex Networks*, vol. 8, no. 6, cnaa047, 2020.

[231] Z. Yang, Z. Wu, M. Luo, *et al.*, "SkyPilot: An Intercloud Broker for Sky Computing," in *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, Boston, MA: USENIX Association, Apr. 2023, pp. 437–455, ISBN: 978-1-939133-33-5. [Online]. Available: https://www.usenix.org/conference/nsdi23/presentation/yang-zongheng.

[232] J. A. Weymark, "Generalized Gini Inequality Indices," *Mathematical Social Sciences*, vol. 1, no. 4, pp. 409–430, 1981.

[233] R. R. Yager, "On Ordered Weighted Averaging Aggregation Operators in Multicriteria Decisionmaking," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 18, no. 1, pp. 183–190, 1988.

[234] R. Busa-Fekete, B. Szörényi, P. Weng, and S. Mannor, "Multi-Objective Bandits: Optimizing the Generalized Gini Index," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML'17, Sydney, NSW, Australia: JMLR.org, 2017, pp. 625–634.

[235] S. L. Cessie and J. V. Houwelingen, "Ridge Estimators in Logistic Regression," *Journal of the Royal Statistical Society Series C: Applied Statistics*, vol. 41, no. 1, pp. 191–201, 1992.

[236] X. Liu, J. Zuo, J. Wang, Z. Wang, Y. Xu, and J. C. Lui, "Learning Context-Aware Probabilistic Maximum Coverage Bandits: A Variance-Adaptive Approach," in *IEEE INFOCOM 2024-IEEE Conference on Computer Communications*, IEEE, 2024, pp. 2189–2198.

[237] N. Golrezaei, R. Niazadeh, K. K. Patel, and F. Susan, "Online Combinatorial Optimization with Group Fairness Constraints," in *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, K. Larson, Ed., Main Track, International Joint Conferences on Artificial Intelligence Organization, Aug. 2024, pp. 394–402. DOI: 10.24963/ijcai.2024/44. [Online]. Available: https://doi.org/10.24963/ijcai.2024/44.

[238] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The Internet Topology Zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, 2011. DOI: 10.1109/JSAC.2011.111002.

[239] D. Rossi and G. Rossini, "Caching Performance of Content Centric Networks under Multi-Path Routing (and more)," *Relatório técnico, Telecom ParisTech*, vol. 2011, pp. 1–6, 2011.

[240] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A Contextual-Bandit Approach to Personalized News Article Recommendation," in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 661–670.

[241] L. Qin, S. Chen, and X. Zhu, "Contextual Combinatorial Bandit and its Application on Diversified Online Recommendation," in *Proceedings of the 2014 SIAM International Conference on Data Mining*, SIAM, 2014, pp. 461–469.

[242] S. Wang and W. Chen, "Thompson Sampling for Combinatorial Semi-Bandits," in *Proceedings of the 35th International Conference on Machine Learning*, J. Dy and A. Krause, Eds., ser. Proceedings of Machine Learning Research, vol. 80, PMLR, Oct. 2018, pp. 5114–5122. [Online]. Available: https://proceedings.mlr.press/v80/wang18a.html.

[243] K. Dang, H. Khalifé, M. Sintorn, D. Lindbo, and S. Secci, "Deep Reinforcement Learning for Joint Energy Saving and Traffic Handling in xG RAN," in *ICC 2024 - IEEE International Conference on Communications*, 2024, pp. 4743–4748. DOI: 10.1109/ICC51166.2024.10622652.

[244] S. Wu, N. Chen, A. Xiao, P. Zhang, C. Jiang, and W. Zhang, "AI-Empowered Virtual Network Embedding: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, 2024.

[245] C. Zhang, M. Fiore, C. Ziemlicki, and P. Patras, "Microscope: Mobile Service Traffic Decomposition for Network Slicing as a Service," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '20, London, United Kingdom: Association for Computing Machinery, 2020, ISBN: 9781450370851. DOI: 10.1145/3372224.3419195. [Online]. Available: https://doi.org/10.1145/3372224.3419195.

[246] H. Awada, S. Berri, and A. Chorti, "Learning-Based Resource Allocation for MBRLLC and Homogeneous Slices in 6G Networks," in *2024 3rd International Conference on 6G Networking (6GNet)*, 2024, pp. 127–134. DOI: 10.1109/6GNet63182.2024.10765787.

[247] Z. Xu, F. Y. Yan, and M. Yu, "Zeal: Rethinking Large-Scale Resource Allocation with "Decouple and Decompose"," 2024. [Online]. Available: http://arxiv.org/abs/2412.11447.

[248] R. Panwar and M. Supriya, "RLPRAF: Reinforcement Learning-Based Proactive Resource Allocation Framework for Resource Provisioning in Cloud Environment," *IEEE Access*, vol. 12, pp. 95 986–96 007, 2024. DOI: 10.1109/ACCESS.2024.3421956.

[249] O. Poppe, P. Castro, W. Lang, and J. Leeka, "Proactive Resource Allocation Policy for Microsoft Azure Cognitive Search," *SIGMOD Rec.*, vol. 52, no. 3, pp. 41–48, Nov. 2023, ISSN: 0163-5808. DOI: 10.1145/3631504.3631516. [Online]. Available: https://doi.org/10.1145/3631504.3631516.

[250] A. Staffolani, V.-A. Darvariu, L. Foschini, M. Girolami, P. Bellavista, and M. M. Foschini, "PRORL: Proactive Resource Orchestrator for Open RANs Using Deep Reinforcement Learning," *IEEE Transactions on Network and Service Management*, vol. 21, no. 4, pp. 3933–3944, 2024. DOI: 10.1109/TNSM.2024.3373606.

[251] W. Rafique, J. Rani Barai, A. O. Fapojuwo, and D. Krishnamurthy, "A Survey on Beyond 5G Network Slicing for Smart Cities Applications," *IEEE Communications Surveys & Tutorials*, vol. 27, no. 1, pp. 595–628, 2025. DOI: 10.1109/COMST.2024.3410295.

[252] S. Vassilaras, L. Gkatzikis, N. Liakopoulos, *et al.*, "The Algorithmic Aspects of Network Slicing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 112–119, 2017. DOI: 10.1109/MCOM.2017.1600939.

[253] D. Bega, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez, "DeepCog: Cognitive Network Management in Sliced 5G Networks with Deep Learning," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 280–288. DOI: 10.1109/INFOCOM.2019.8737488.

[254] F. Wei, G. Feng, Y. Sun, Y. Wang, S. Qin, and Y.-C. Liang, "Network Slice Reconfiguration by Exploiting Deep Reinforcement Learning With Large Action Space," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2197–2211, 2020. DOI: 10.1109/TNSM.2020.3019248.

[255] M. M. H. Qazzaz, L. Kulacz, A. Kliks, S. A. Zaidi, M. Dryjanski, and D. McLernon, "Machine Learning-based xApp for Dynamic Resource Allocation in O-RAN Networks," in *2024 IEEE International Conference on Machine Learning for Communication*

and Networking (ICMLCN), 2024, pp. 492–497. DOI: 10.1109/ICMLCN59089.2024.10625184.

[256] L. L. Schiavo, J. A. Ayala-Romero, A. Garcia-Saavedra, M. Fiore, and X. Costa-Perez, "YinYangRAN: Resource Multiplexing in GPU-Accelerated Virtualized RANs," in *IEEE INFOCOM 2024-IEEE Conference on Computer Communications*, IEEE, 2024, pp. 721–730.

[257] L. L. Schiavo, G. Garcia-Aviles, A. Garcia-Saavedra, *et al.*, "CloudRIC: Open Radio Access Network (O-RAN) Virtualization with Shared Heterogeneous Computing," in *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, 2024, pp. 558–572.

[258] F. Tonini, C. Natalino, M. Furdek, C. Raffaelli, and P. Monti, "Network Slicing Automation: Challenges and Benefits," in *2020 International Conference on Optical Network Design and Modeling (ONDM)*, 2020, pp. 1–6. DOI: 10.23919/ONDM48393.2020.9133004.

[259] J. Ajayi and T. Braun, "Proactive Resource Optimization for Heterogeneous 5G Network Slicing," in *GLOBECOM 2025 - 2025 IEEE Global Communications Conference*, 2025.

[260] O. Stenhammar, G. Fodor, and C. Fischione, "A Comparison of Neural Networks for Wireless Channel Prediction," *IEEE Wireless Communications*, vol. 31, no. 3, pp. 235–241, 2024. DOI: 10.1109/MWC.006.2300140.

[261] T. Ouyang, K. Zhao, G. Hong, X. Zhang, Z. Zhou, and X. Chen, "Dynamic Edge-Centric Resource Provisioning for Online and Offline Services Co-Location via Reactive and Predictive Approaches," *IEEE Transactions on Networking*, pp. 1–16, 2025. DOI: 10.1109/TON.2025.3535499.

[262] Z. Zhao, N. Emami, H. Santos, *et al.*, "Reinforced-LSTM Trajectory Prediction-Driven Dynamic Service Migration: A Case Study," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 4, pp. 2786–2802, 2022. DOI: 10.1109/TNSE.2022.3169786.

[263] M. Tsampazi, S. D'Oro, M. Polese, *et al.*, "PandORA: Automated Design and Comprehensive Evaluation of Deep Reinforcement Learning Agents for Open RAN," *IEEE Transactions on Mobile Computing*, vol. 24, no. 4, pp. 3223–3240, 2025. DOI: 10.1109/TMC.2024.3505781.

[264] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[265] T. Tsourdinis, I. Chatzistefanidis, N. Makris, and T. Korakis, *UE Network Traffic Time-Series (Applications, Throughput, Latency, CQI) in LTE/5G Networks*, 2022. DOI: 10.21227/4ars-fs38. [Online]. Available: https://dx.doi.org/10.21227/4ars-fs38.

[266] G. Brockman, V. Cheung, L. Pettersson, *et al.*, "OpenAI Gym," *arXiv preprint arXiv:1606.01540*, 2016.

[267] H. Zhang, A. Zhou, J. Lu, *et al.*, "OnRL: Improving Mobile Video Telephony via Online Reinforcement Learning," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '20, London, United Kingdom:

Association for Computing Machinery, 2020, ISBN: 9781450370851. DOI: 10.1145/3372224.3419186. [Online]. Available: https://doi.org/10.1145/3372224.3419186.

[268] L. Bonati, P. Johari, M. Polese, *et al.*, "Colosseum: Large-Scale Wireless Experimentation Through Hardware-in-the-Loop Network Emulation," in *2021 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2021, pp. 105–113. DOI: 10.1109/DySPAN53946.2021.9677430.

[269] V.-A. Darvariu, S. Hailes, and M. Musolesi, "Graph Reinforcement Learning for Combinatorial Optimization: A Survey and Unifying Perspective," *arXiv preprint arXiv:2404.06492*, 2024.

[270] J. Wang, L. Zhang, Y. Yang, *et al.*, "Network Meets ChatGPT: Intent Autonomous Management, Control and Operation," *Journal of Communications and Information Networks*, vol. 8, no. 3, pp. 239–255, 2023. DOI: 10.23919/JCIN.2023.10272352.

[271] M. Ameur, B. Brik, and A. Ksentini, "Leveraging LLMs to eXplain DRL Decisions for Transparent 6G Network Slicing," in *2024 IEEE 10th International Conference on Network Softwarization (NetSoft)*, 2024, pp. 204–212. DOI: 10.1109/NetSoft60951.2024.10588921.

[272] S. El-Zahr, P. Gunning, and N. Zilberman, "Exploring the Benefits of Carbon-Aware Routing," *Proc. ACM Netw.*, vol. 1, no. CoNEXT3, Nov. 2023. DOI: 10.1145/3629165. [Online]. Available: https://doi.org/10.1145/3629165.

[273] P. Li, N. Christianson, J. Yang, A. Wierman, and S. Ren, "Learning for Sustainable Online Scheduling with Competitive Fairness Guarantees," in *Proceedings of the 16th ACM International Conference on Future and Sustainable Energy Systems*, ser. E-Energy '25, Association for Computing Machinery, 2025, pp. 63–79, ISBN: 9798400711251. DOI: 10.1145/3679240.3734615. [Online]. Available: https://doi.org/10.1145/3679240.3734615.

[274] C. Elsworth, K. Huang, D. Patterson, *et al.*, "Measuring the Environmental Impact of Delivering AI at Google Scale," *arXiv preprint arXiv:2508.15734*, 2025.

[275] N. Shumba, O. Tshekiso, P. Li, G. Fanti, and S. Ren, "A Water Efficiency Dataset for African Data Centers," in *Proceedings of the 2025 ACM SIGCAS/SIGCHI Conference on Computing and Sustainable Societies*, ser. COMPASS '25, Association for Computing Machinery, 2025, pp. 453–460, ISBN: 9798400714849. DOI: 10.1145/3715335.3735483. [Online]. Available: https://doi.org/10.1145/3715335.3735483.

[276] C. Yeh, V. Li, R. Datta, *et al.*, "SustainGym: Reinforcement Learning Environments for Sustainable Energy Systems," in *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, New Orleans, LA, USA, Dec. 2023. [Online]. Available: https://openreview.net/forum?id=vZ9tA3o3hr.

[277] A. H. Mahmud and S. Iyengar, "A Distributed Framework for Carbon and Cost Aware Geographical Job Scheduling in a Hybrid Data Center Infrastructure," in *2016 IEEE international conference on autonomic computing (ICAC)*, IEEE, 2016, pp. 75–84.

[278] D. Xenakis, E. Samikwa, J. Ajayi, A. D. Maio, T. Braun, and K. Schlegel, "Towards Personality Detection and Prediction using Smartphone Sensor Data," in *2023 21st*

Mediterranean Communication and Computer Networking Conference (MedComNet), 2023, pp. 121–130. DOI: 10.1109/MedComNet58619.2023.10168869.