ISSN 1688-2784

Universidad de la República
Facultad de Ingeniería

Learning-Based Resource Allocation for Fair and Efficient Mobile Networks

Tesis presentada a la Facultad de Ingeniería de la Universidad de la República por

Martín Randall Carlevaro

en cumplimiento parcial de los requerimientos para la finalización de la carrera de Doctorado en Ingeniería Eléctrica.

Dirección de Tesis

Pablo Belzarena	Universidad de la República
Federico La Rocca	Universidad de la República

Tribunal

Miguel Calvo-Fullana	Universitat Pompeu Fabra
Marcelo Fiori	Universidad de la República
María Simon	Universidad de la República
Stefano Secci (Revisor Externo)	Conservatoire National des Arts et Métiers
José Suárez-Varela (Revisor Externo)	Telefónica Innovación Digital

Dirección Académica

Pablo Belzarena	Universidad de la República

Montevideo
Wednesday 29th April, 2026

Learning-Based Resource Allocation for Fair and Efficient Mobile Networks, Martín Randall Carlevaro.

ISSN 1688-2784

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.2).
Contiene un total de 136 páginas.
Compilada el Wednesday 29th April, 2026.
http://iie.fing.edu.uy/

El hombre de los países industriales ha llegado a la luna dominando la naturaleza. ¿Es justo que el hombre ponga un pie sobre la luna? ¿O no sería más justo que los grandes países pongan los pies sobre la tierra y se den cuenta de que hay millones de personas que no tienen trabajo y que sufren de hambre?

Salvador Allende

1. Each person is to have an equal right to the most extensive total system of equal basic liberties compatible with a similar system of liberty for all.

2. Social and economic inequalities are to be arranged so that they are both:

   (a) to the greatest benefit of the least advantaged, consistent with the just savings principle, and

   (b) attached to offices and positions open to all under conditions of fair equality of opportunity.

John Rawls, A Theory of Justice

This page has been intentionally left blank.

Agradecimientos

Quiero agradecer a la Agencia Nacional de Investigación e Innovación y a la Comisión Sectorial de Investigación Científica de la Universidad de la República. Gracias al respaldo de ambas instituciones conté con los recursos necesarios (particularmente tiempo, el más importante) para llevar adelante este trabajo. A mis tutores, Pablo Belzarena y Federico La Rocca, muchas gracias por el acompañamiento, las lecturas atentas, los comentarios, las sugerencias y la generosidad para sostener un proceso tan largo y demandante como el de una tesis de doctorado. A los integrantes del tribunal, que tan amablemente se tomaron el tiempo y el trabajo de leer y corregir esta tesis.

Muchos colegas y compañeros merecen incluirse en este agradecimiento por aportes similares, entre los que agradezco especialmente a Pedro Casas y Santiago Paternain. Finalmente, a familiares, amigos y compañeros, que se encontrarán comprendidos en la dedicatoria.

This page has been intentionally left blank.

Dedicatoria

Cosa rara es la justicia,
que se mide por su opuesto,
el sistema lleva puesto,
mientras premia la codicia,
lo desigual con pericia.
Por eso al que es olvidado,
que el sistema ha desechado,
lo incluyo en estos versos,
que dedican mis esfuerzos,
hechos en el doctorado.

Disculpeme lo imprudente
de que se me vaya largo,
el saludo y me hago cargo,
permitame que yo intente,
mencionar ahora a mi gente.
Empiezo por los tutores,
colegas y otros actores,
'Hallo!' y me voy pa Viena,
fundamental pa la faena,
Al Pedro sus tres vitores!

A cada vuelta un sobrino,
la familia va creciendo!
El perro me está sonriendo:
me fue asignado con tino,
cuidarlo es hoy mi destino.
Ante el pasar de la horas,
'a jugar' Julia me imploras,
mientras el Guille te explica
el título que Emma aplica:
'dotor de computadoras'...

Amigos ya llegué a ustedes,
compañeros de mil luchas,
las esperanzas son muchas,
se siguen tejiendo redes:
para tumbar las paredes.
Ya por acá me despido,
con el saludo más sentido:
que se comparte entre el lector,
y el gran amor de este autor,
que se va con un Chasquido!

This page has been intentionally left blank.

Summary

This thesis explores the application of artificial intelligence to resource allocation in wireless networks. Next-generation cellular systems, with 5G already deployed and 6G under development, drastically increase both the capacity and complexity of the network, demanding more careful design for improved resource utilization and operational efficiency.

Traditional methods based on network overprovisioning are no longer sufficient. The use of mid- and high-frequency bands introduces new challenges in urban environments and densely populated areas. Likewise, classical resource allocation schemes based on fixed rules struggle to ensure both efficiency, fairness, and adaptability. To address these limitations, modern networks increasingly rely on intelligent management mechanisms capable of dynamically adapting to heterogeneous traffic conditions, with the dual objective of maintaining service quality and reducing operational costs.

This work focuses precisely on time–frequency resource allocation by base stations, considering two complementary levels: user association at the inter-cell level and internal resource scheduling at the intra-cell level.

The first part of this thesis considers the user association problem, which seeks to determine to which base station a mobile device should connect upon entering the network. Differently from previous efforts, the focus is on fairness and stability in a dynamic setting where users arrive randomly and have finite session durations. To this end, the fair user association problem is formalized and studied under realistic system dynamics, including user departures and the possibility of admission rejection. The problem is modeled as a partially observable Markov Decision Process, with several variants reflecting different modeling assumptions and optimization objectives. Reinforcement learning techniques are employed, leveraging graph neural networks to exploit the relational structure of the network and to learn decentralized yet stable association policies. A principled method is proposed to incorporate fairness criteria into the decision-making process, adapted to partially observable states and time-varying user populations. The proposed approach is validated using real data from a 5G network in Paris, demonstrating improvements in both load balancing and user experience.

In the second part of the thesis, we examine the problem of resource distribution within a base station. The 5G standard defines three service categories: enhanced Mobile Broadband (eMBB), massive Machine-Type Communications (mMTC), and Ultra-Reliable Low-Latency Communications (URLLC). This work focuses on the coexistence of eMBB and URLLC services, where URLLC traffic has

strict latency and reliability requirements and is granted priority access to spectrum resources, potentially preempting resources previously allocated to eMBB users. Given that these services operate on different time scales, the proposed approach combines classical optimization techniques with supervised learning methods to design resource allocation policies that satisfy URLLC requirements while minimizing the performance degradation experienced by eMBB users.

The proposed solutions contribute to the advancement of artificial intelligence applied to resource management in wireless networks. Efficient resource optimization is not only desirable but necessary given the growing pressure on the planet's natural resources. This thesis emphasizes the use of machine learning not only to maximize efficiency, but also as a means to promote fairness in access to shared resources.

Resumen Público

Esta tesis explora la aplicación de la inteligencia artificial a la asignación de recursos en redes inalámbricas. Los sistemas celulares de próxima generación, con 5G ya desplegado y 6G en desarrollo, incrementan significativamente tanto la capacidad como la complejidad de la red, lo que exige un diseño más cuidadoso orientado a una mejor utilización de los recursos y a una mayor eficiencia operativa.

Los enfoques tradicionales basados en el sobredimensionamiento de la red ya no resultan suficientes. El uso de bandas de frecuencia medias y altas introduce nuevos desafíos, en particular en entornos urbanos y áreas densamente pobladas. Asimismo, los esquemas clásicos de asignación de recursos basados en reglas fijas tienen dificultades para garantizar simultáneamente eficiencia, equidad y adaptabilidad. Para superar estas limitaciones, las redes modernas incorporan mecanismos de gestión inteligente capaces de adaptarse dinámicamente a condiciones de tráfico heterogéneas, con el doble objetivo de mantener la calidad de servicio y reducir los costos operativos.

Este trabajo se centra precisamente en la asignación de recursos tiempo-frecuencia por parte de las estaciones base, considerando dos niveles complementarios: la asociación de usuarios a nivel intercelda y la planificación interna de recursos a nivel intracelda.

La primera parte de esta tesis aborda el problema de asociación de usuarios, cuyo objetivo es determinar a qué estación base debe conectarse un dispositivo móvil al ingresar a la red. A diferencia de trabajos previos, el enfoque se pone en la equidad y la estabilidad en un entorno dinámico donde los usuarios llegan de forma aleatoria y tienen sesiones de duración finita. Con este fin, se formaliza el problema de asociación justa de usuarios y se estudia bajo dinámicas de sistema realistas, incluyendo la salida de usuarios y la posibilidad de rechazo en la admisión. El problema se modela como un Proceso de Decisión de Markov parcialmente observable, con varias variantes que reflejan diferentes supuestos de modelado y objetivos de optimización. Se emplean técnicas de aprendizaje por refuerzo, aprovechando redes neuronales basadas en grafos para explotar la estructura relacional de la red y aprender políticas de asociación descentralizadas y estables. Se propone un método principiado para incorporar criterios de equidad en el proceso de toma de decisiones, adaptado a estados parcialmente observables y a poblaciones de usuarios dinámicas. El enfoque propuesto se valida utilizando datos reales de una red 5G en París, mostrando mejoras tanto en el balance de carga como en la experiencia de los usuarios.

En la segunda parte de la tesis se examina el problema de distribución de recursos dentro de una estación base. El estándar 5G define tres categorías de servicio: banda ancha móvil mejorada (eMBB), comunicaciones masivas de tipo máquina (mMTC) y comunicaciones ultra confiables y de baja latencia (URLLC). Este trabajo se centra en la coexistencia de servicios eMBB y URLLC, donde el tráfico URLLC presenta requisitos estrictos de latencia y confiabilidad y recibe prioridad en el acceso al espectro, pudiendo incluso reasignar recursos previamente asignados a usuarios eMBB. Dado que estos servicios operan en diferentes escalas temporales, la tesis combina técnicas clásicas de optimización con métodos de aprendizaje supervisado para diseñar políticas de asignación de recursos que cumplan con los requisitos de URLLC minimizando, al mismo tiempo, la degradación del desempeño de los usuarios eMBB.

En conjunto, las soluciones propuestas contribuyen al avance de la aplicación de la inteligencia artificial a la gestión de recursos en redes inalámbricas. La optimización eficiente de recursos no es solamente un objetivo deseable, sino una necesidad ante la creciente presión sobre los bienes naturales del planeta. Esta tesis enfatiza el uso de técnicas de aprendizaje automático no solo para maximizar la eficiencia, sino también como medio para promover la equidad en el acceso a los recursos compartidos.

Table of contents

This page has been intentionally left blank.

Chapter 1

Introduction

1.1 Societal and Technological Context

Internet has secured a vital place in modern society; the COVID-19 pandemic only made this clearer [1]. It supports critical infrastructures, drives consumption, and mediates social interactions. Its role is undeniable. Like water from a tap, it has become a necessity, but one that carries costs and risks.

Even in a world that remains deeply analog, the Internet has transformed something central to human life: interpersonal communication [2, 3]. Individuals increasingly turn to the digital realm for connection, solace, or distraction. Screens mediate affection, attention, and social bonds, creating new dependencies. Social networks, messaging apps, and streaming platforms dominate human attention, raising questions about psychological and societal resilience [4, 5].

Yet, this indispensable system is far from free. Every byte, every video stream, every connection consumes energy. The Information and Communication Technology sector currently accounts for approximately 1.5–2% of total electricity consumption [6], with projections reaching 3–4% by 2030 if current growth trends continue [7]. Data centers alone consumed over 200 TWh in 2022 [8], and wireless networks are responsible for a significant fraction of this footprint, especially as densification and massive MIMO deployments increase [9].

Infrastructure must be renewed or replaced every few years [10]. Spectrum must be densified. Hardware must be upgraded. The push for higher data rates—8K screens, ultra-low latency services, and massive IoT—raises pressing sustainability questions. Concentration of traffic through a few monopolistic platforms increases systemic fragility. Large-scale outages, such as the AWS disruption in 2020, demonstrated the vulnerability of globally interdependent networks [11, 12].

Beyond infrastructure investments, achieving coverage levels such as those in 5G presents curious facts:

1. The protocols underlying modern networks are over fifty years old, and remarkably, they still function reliably [13].

2. The data generated through the Internet has fueled the growth of artificial intelligence (AI) [14], yet AI has not contributed in turn to the improvement of Internet itself. In some cases, it may even exacerbate inefficiencies: super-concentration of traffic in a few monopolistic companies threatens the architecture of the Internet [15, 16].

It is clear that much remains to integrate AI into these systems to replace or modify protocols that, ultimately, already work. Without the pretension to bridge this gap, but with a clear objective of exploring its potential, this thesis explores this subject, addressing the use of artificial intelligence for resource allocation in wireless networks.

1.2 AI for Resource Allocation in Wireless Networks

Despite the maturity of existing communication protocols, a gap remains between their rule-based design and the adaptive, data-driven optimization capabilities enabled by artificial intelligence. New generations of communication technologies aim to fill this gap [9]. The surge of AI applications extends to telecommunications, offering the potential to optimize network performance. In this work, we focus on mobile networks, the networks that put the Internet in our pockets.

In the context of resource allocation, what we refer to as **AI** can be seen as the use of learning-based optimization techniques to automatically select actions that maximize certain notion of system performance. The goal is to optimize system performance, using learning techniques applied to terabytes of data with processors capable of handling them [17]. A critical question arises: what exactly should be optimized? Many AI applications focus on profit, advertising efficiency, or precision of military systems [18]. As a counter, beneficial applications do indeed exist, being its incorporation to healthcare the most visible, with resource management as an interesting option [19, 20].

Fortunately, and increasingly driven by high energy costs and spectrum scarcity, significant effort is being directed toward efficient resource utilization [21]. The traditional paradigm of network oversizing no longer holds. Exponential growth in traffic and connected devices, together with high deployment costs, demands a reconsideration of network design and more intelligent resource usage [9, 21].

Next-generation networks enable and encourage the application of AI for adaptation and management, but also introduce new challenges. In this thesis, we study the use of AI in 5G wireless networks, and tackle two problems that emerge or are exacerbated in this context: **user association** and **intra-cell resource allocation**. These problems exemplify the complexity, multi-scale interactions, and fairness–energy trade-offs that characterize modern mobile networks [21, 22].

1.2.1 User Association (UA)

User association, the process of determining which base station each user should connect to, has been studied for decades, yet it remains critical in modern deploy-

ments [23]. The increasing densification of cells means that each user may have multiple potential base stations in close proximity [9]. This amplifies interference patterns and requires dynamic adaptation to traffic fluctuations and mobility. Additionally, energy efficiency and fairness among users have become central objectives: simply maximizing throughput may overload certain base stations and leave others underutilized, wasting energy and spectrum [24].

Solving UA in this context involves several difficulties. The state space grows combinatorially with the number of users and base stations, and interactions between users are non-linear due to interference. In general, UA decisions must balance multiple, often conflicting objectives, such as throughput, fairness, heterogeneous user requirements, coverage, and energy efficiency, adding to the problem complexity [25]. In this work, we focus on the trade-off between throughput and fairness in resource allocation. Traditional heuristics often fail to capture this balance effectively, while static assignment policies cannot react to real-time changes in user distribution or channel conditions.

In this work, we address UA using **Graph Neural Networks (GNNs)** to represent the network topology and **Deep Reinforcement Learning (DRL)** to learn adaptive, sequential decision-making policies [26]. This combination enables capturing both the relational structure of the network and the dynamic, time-dependent aspects of user assignments. Our approach emphasizes fairness, proposing a Markov Decision Process (MDP) formulation that encourages equitable resource distribution while maintaining high performance.

Beyond performance metrics, the proposed approach has a direct impact on system stability, particularly under high-load conditions. By explicitly accounting for fairness in the decision-making process, the learned policies prevent persistent overload of individual base stations and reduce oscillatory user reassignments, leading to more stable and predictable network operation. Importantly, this is achieved in a decentralized manner, where decisions are taken locally using shared relational information, making the approach scalable and well suited for dense, large-scale deployments.

1.2.2 Intra-Cell Resource Allocation and Multi-User Coexistence

The introduction of 5G standards brings new classes of users with heterogeneous requirements [9, 27]. **Enhanced Mobile Broadband (eMBB)** users demand high throughput for applications such as streaming, gaming, and data transfer. **Ultra-Reliable Low-Latency Communications (URLLC)** users require extremely low latency and high reliability for mission-critical applications such as industrial automation and autonomous driving [28]. **Massive Machine-Type Communications (mMTC)** devices generate sporadic, low-rate connections for IoT sensing and control [29]. The coexistence of these diverse traffic types within a single base station creates complex scheduling and resource allocation challenges.

Intra-cell allocation must contend with multi-scale temporal dynamics. eMBB users are typically scheduled over regular *slots*, while URLLC transmissions occupy shorter *minislots* that can preempt ongoing eMBB transmissions [30]. URLLC

packets often arrive unpredictably, forcing the system to anticipate and mitigate conflicts across timescales. The allocation policy must balance competing objectives: minimizing throughput degradation for eMBB users, ensuring URLLC reliability and latency, and maintaining capacity for mMTC devices. Additional constraints such as energy efficiency, fairness, and spectral efficiency further complicate decision-making [31].

The difficulty of this problem arises from multiple sources. First, multi-scale temporal coupling requires coordination across scheduling intervals of different lengths. Second, heterogeneous Quality-of-Service (QoS) requirements introduce inherently conflicting objectives. Third, the stochastic and bursty arrival of URLLC traffic adds uncertainty that must be addressed in real time. Finally, scalability becomes critical: realistic deployments involve hundreds of users, antennas, and scheduling variables, yielding high-dimensional optimization spaces [21, 32].

To address these challenges, this thesis adopts a combination of classical optimization and supervised learning [20, 33]. The problem is first formalized as a constrained optimization task reflecting QoS priorities and temporal structure. Then, supervised learning methods are used to approximate these near-optimal policies efficiently, enabling real-time operation while maintaining strong performance guarantees.

1.3 Research Objectives, Contributions, and Thesis Structure

This thesis addresses two intertwined problems in 5G networks:

1. **User association (UA)** via GNN+DRL for adaptive, topology-aware policies.
2. **Intra-cell resource allocation** via optimization and supervised learning, managing heterogeneous traffic across multiple temporal scales.

Key contributions include:

- A GNN+DRL framework for UA balancing load, fairness, and energy efficiency.
- MDP formulations capturing fairness, throughput, and energy trade-offs for UA.
- Algorithms combining optimization and supervised learning for intra-cell allocation under heterogeneous traffic.
- Empirical validation demonstrating superior performance over conventional heuristics.
- Frameworks explicitly handling coexistence of eMBB and URLLC users across temporal scales.

The thesis is organized into two main parts, each devoted to a specific resource allocation problem:

- **Part I: User association.** The first part of the thesis addresses the user association problem. Chapter 2 introduces the problem setting and reviews the related literature, and the network and system models are presented. Chapter 3 develops the proposed learning-based solution, combining graph neural networks with deep reinforcement learning. Chapter 4 presents a decentralized solution for user association considering slot-fairness. In Chapter 5, an explicit MDP formulation is introduced to incorporate fairness criteria directly into the decision-making process. This chapter ends with a summary of our approaches to the UA problem.
- **Part II: Intra-cell resource allocation.** The second part focuses on resource allocation within a base station. Chapter 6 discusses the coexistence challenges of different service types and surveys related work, and ends by formulating the corresponding optimization problem. In Chapter 7, we apply the presented formulation to the URLLC and eMBB coexistence problem, using supervised learning techniques to approximate near-optimal allocation policies. Chapter 8 finishes this part by presenting experimental results validating our proposal.

The proposed methods have been validated through **peer-reviewed publications**, such as [34–36] for the User Association part and [37] for the intra-cell resource allocation. To promote transparency and reproducibility, the code used in the reported experiments is publicly available in open repositories. The study of the user association problem with a focus on horizon fairness has been submitted to a journal and is under revision (IEEE Transactions on Vehicular Technology). The thesis concludes with a synthesis of findings, discussion of broader implications, and directions for future research.

This page has been intentionally left blank.

Part I

User Association: a sequential decision problem for fair resource allocation

Chapter 2

Old problems, new solutions: looking at User Association

2.1 Introduction

User association (UA) is a central mechanism in wireless networks. It determines which connectivity provider—such as a base station, access point, satellite, or aerial platform—serves each active user at a given time. This decision directly influences throughput, latency, load balancing, and energy efficiency, and therefore has a decisive impact on overall network performance and user experience.

Optimizing UA offers substantial gains. By properly distributing users among available resources, the network can improve spectral efficiency, reduce congestion, and extend service coverage without additional infrastructure. Efficient association decisions also enhance energy utilization, allowing underused nodes to enter low-power modes while overloaded ones are relieved. From the user side, a well-designed UA policy leads to fairer service distribution and reduced blocking probability, which are critical for quality-of-service guarantees in dense deployments.

Yet UA remains a difficult problem. Its difficulty arises from three main factors:

- Combinatorial complexity: the number of possible association configurations grows exponentially with the number of users and nodes, making exact optimization intractable.
- Coupled dynamics: association decisions at one node influence the state and performance of others through interference and shared spectrum.
- Uncertainty and time variability: user mobility, fluctuating channel conditions, and nonstationary traffic patterns require continuous adaptation.

As we discuss in section 2.2, traditional approaches based on received signal strength or static biasing fail to capture these interactions. Optimization-based formulations can yield better results but depend on centralized control and full network information, which is unrealistic in large-scale or dynamic environments.

This motivates the search for learning-based, distributed methods that can infer effective association policies directly from experience, adapting autonomously to varying conditions.

In this context, reinforcement learning (RL) provides a principled framework for sequential decision-making under uncertainty. By formulating UA as a partially observable Markov Decision Process (POMDP), each network agent can learn policies that maximize a long-term performance metric derived from throughput, fairness, or system utility. Furthermore, the structure of wireless networks naturally lends itself to graph-based representations, where nodes correspond to serving entities and edges capture physical or logical relationships such as interference or cooperation. Graph Neural Networks (GNNs) can exploit this structure to learn scalable, permutation-equivariant policies suitable for decentralized operation.

This first part develops a complete line of work exploring RL- and GNN-based user association, progressively increasing the realism and scope of the model. After the introduction and related works, we present the system model and problem statement. The study advances through four main stages:

- A vanilla formulation for throughput maximization. The initial model assumes that all arriving users must be accepted by one of the serving nodes. The problem is formulated under the RL framework, and the system learns to allocate resources to maximize total throughput. This step establishes the foundation for representing UA as a graph-based learning problem, demonstrating the suitability of GNNs for distributed value estimation and decision-making.

- A decentralized slot-fair formulation. Real networks must sometimes reject users when resources are exhausted. We extend the model to propose a decentralized implementation and include rejection as an explicit decision option, represented by a dedicated node connected to all others. This modification allows the system to balance efficiency and stability under congestion while preserving the graph structure. The formulation introduces slot-time fairness, ensuring equitable service distribution within short scheduling intervals.

- An *average a-posteriori* fairness-aware MDP formulation. The final stage integrates fairness directly into the MDP design. The fairness term is incorporated into the reward and state definitions, producing more balanced decisions without compromising decentralization or scalability. This formulation considers user histories and long-term service disparities, allowing each node to learn policies that balance instantaneous efficiency with sustained equity. The approach maintains decentralized learning and remains robust under nonstationary demand.

Together, these formulations form a coherent framework we call GROWS (Graph Representation Of Wireless Systems) that evolves from a simple DRL baseline to a fairness-aware, scalable, and distributed association mechanism. GROWS

exploits the relational structure of the network to generalize across different deployments and topologies, while its modular learning process enables easy adaptation to diverse scenarios.

In this part we also present several implementations of the proposed approach, comparing architectural choices and evaluating their behavior in simulated and semi-realistic settings. Experimental analysis includes validation of learned policies, exploration of emergent properties such as low rejection rates, permutation equivariance, as well as an application to a Paris-based deployment scenario demonstrating the framework's practical potential.

The remainder of this part is organized as follows. In the rest of this chapter, we analyze related works 2.2 and the problem statement and system model 6.3. Chapter 3 presents the naive reinforcement learning approach, which is the testbed for the GROWS framework: combining RL+GNN for user association. Chapter 4 introduces a decentralized implementation for slot fairness. Finally, chapter 5 presents the complete MDP formulation for fairness inclusion along an implementation under the GROWS framework. The chapter ends with a summary on our approaches to UA, closing this first part.

2.2 Related Works

User association has long been a central challenge in wireless networks, influencing throughput, energy efficiency, and fairness in resource distribution. The rapid increase in traffic, heterogeneity of base stations, and the emergence of intelligent, adaptive networks have renewed attention on this problem, which lies at the intersection of optimization, control, and learning. This section reviews the main lines of research that have shaped UA in modern mobile systems, covering its traditional formulations, reinforcement learning perspectives, graph-based modeling, and fairness considerations both in UA and in reinforcement learning more broadly.

2.2.1 Approaches to the UA problem

UA has become a recurrent topic in most contemporary surveys of emerging wireless networks [38–41], with dedicated surveys emphasizing its relevance and complexity [42, 43]. The problem has been approached from several angles, ranging from simple received-signal–based heuristics to complex multi-objective optimization frameworks where UA interacts with other layers of the system, such as power control, beamforming, or handover management.

2.2.2 UA in mobile networks

The earliest and still widely used approach to UA is received-power maximization, where each user connects to the base station (BS) offering the strongest signal. To mitigate load imbalance, biasing strategies were introduced, effectively modifying the received power metric to favor underloaded cells [42, 44]. Despite the

rise of learning-based methods, heuristic approaches remain competitive for their simplicity and speed [45, 46].

As networks evolved, UA began to be treated explicitly as an optimization problem. Several mathematical frameworks emerged, such as combinatorial optimization [47], game theory [48–50], and stochastic geometry [51], each emphasizing a different aspect of system structure or uncertainty. A common challenge in these formulations lies in the discrete or binary nature of association decisions, which often results in non-convex objective functions. Researchers have proposed a range of relaxations and approximations to obtain tractable solutions, including probabilistic modeling of user arrivals and loads (often as Poisson processes) [52–54], constraint relaxation [44, 55], and Lagrangian methods [56], sometimes enhanced with machine learning for efficient approximation [57, 58].

The advent of heterogeneous networks (HetNets), combining macro and small cells, further complicated the problem by invalidating simple signal-based heuristics [54, 58–60]. This led to an explosion of hybrid formulations where UA is solved jointly with other system objectives such as energy consumption [59, 61], unmanned aerial vehicles (UAV) placement [53,55], or beamforming design [60,62]. These problems are frequently decomposed into subproblems or tackled via dual optimization [44, 55, 62]. Although fairness is widely acknowledged as essential, some recent works still prioritize spectral efficiency alone [63–65].

2.2.3 DRL and resource allocation

Because UA decisions are inherently sequential and depend on time-varying network states, modeling the problem as a Markov Decision Process (MDP) has become natural. This perspective allows the application of reinforcement learning (RL) and, more recently, deep RL (DRL) algorithms to optimize long-term performance [66]. In these frameworks, each BS or a centralized agent observes system states—such as channel conditions, load, or interference—and learns policies that maximize cumulative rewards corresponding to throughput, energy efficiency, or fairness.

DRL approaches bring flexibility to non-stationary environments, but their efficiency and scalability depend critically on how the MDP is formulated. The reward structure, state representation, and the degree of decentralization play key roles. Some proposals train centralized agents with global observability, while others distribute the decision-making across BSs or users to better reflect real-world constraints [58,67]. However, the black-box nature of DRL introduces interpretability and stability challenges, motivating the use of structured inductive biases such as graph representations.

2.2.4 GNNs and mobile networks

Graph Neural Networks (GNNs) have recently emerged as powerful tools for modeling wireless systems, where the relationships among nodes (e.g., BSs, users, or relays) can be naturally represented as graphs. Their ability to capture spatial

and relational dependencies makes them particularly suitable for problems like interference management, routing, or scheduling [68]. Within UA, GNNs can enable message-passing–based coordination between BSs, allowing each node to make decisions using local and aggregated information. When combined with RL, they provide scalable and generalizable policies that adapt to changes in network topology or traffic.

2.2.5 Fairness in UA

The definition and integration of fairness into UA is both a theoretical and practical challenge. Fairness provides a bridge between efficiency—maximizing system performance—and equity—ensuring an acceptable distribution of resources among users. The concept dates back to philosophical and economic formulations, notably the works of Rawls [69, 70], and was later formalized for network optimization through the notions of proportional and α-fairness [71, 72]. The axiomatic framework of [73] offers a systematic way to embed fairness into utility functions.

In UA optimization, fairness is often introduced via utility functions that reward balanced allocations, which can be represented as convex optimization problems. These can be addressed using dual methods [74, 75], approximations [76, 77], or learning-based optimization [58]. Other formulations impose fairness constraints explicitly, ensuring the system does not deviate beyond a predefined unfairness threshold [75, 78, 79]. Depending on system dynamics, fairness can be enforced instantaneously (slot-time fairness) or across episodes (horizon- or long-term fairness) [67, 75].

2.2.6 Fairness in RL

The incorporation of fairness into RL extends beyond telecommunications. In MDP settings, fairness can be encouraged through reward shaping, which modifies immediate rewards to align with equitable outcomes [80–83], or through loss shaping, where the learning objective penalizes unfair behaviors [84]. Surveys such as [85, 86] summarize various strategies for embedding fairness into learning systems.

Advanced approaches modify the structure of the MDP itself. For instance, [87] introduces a feedback-MDP to track deviations from fairness, while [88] adapts the actor–critic framework to account for long-term fairness, addressing the mismatch between short-term rewards and fairness objectives. State-augmentation techniques have also been explored [89, 90], where fairness-related information is encoded directly into the state or the value function. When fairness is formulated as a constraint, constrained or safe RL methods [62, 91, 92] are employed to ensure compliance during learning.

2.2.7 Summary and positioning of this work

Our approach builds upon these developments but aims to simplify and unify them. Rather than performing reward engineering, state augmentation, or architecture-specific modifications, we focus on a careful formulation of the MDP itself, such that fairness arises naturally from the reward aggregation process. The proposed model extends traditional UA settings by incorporating user departures, decentralized BS policies, and the explicit possibility of rejecting users. This enables a decentralized yet fair resource allocation process solvable through standard Q-Learning and DDQN algorithms. We validate both a classical neural and a graph-based GNN implementation, showing that the MDP structure itself enforces efficiency and fairness, aligning with the goals identified across the literature.

2.3 User Association in 5G and beyond

Let us briefly describe how user association operates in mobile networks from 4G/LTE onward. When a mobile user (user equipment, UE) attempts to connect, it measures the signal quality of candidate cells. The relevant indicators are the Reference Signal Received Power (RSRP), the Received Signal Strength Indicator (RSSI), and the Signal-to-Interference-plus-Noise Ratio (SINR). Each base station typically serves multiple cells, each corresponding to a distinct antenna sector or beam. Based on these measurements and its own policy, the network selects the serving cell and instructs the UE to attach to it.

Signal quality determines the modulation and coding options available, and thus the achievable rate per time-frequency resource unit. While RSRP is typically used for cell selection, in our work we will consider the SINR because it directly determines the Channel Quality Indicator (CQI), from which the base station selects the modulation and coding scheme (MCS) that defines the bit rate of each time–frequency resource unit. Our time-frequency resource unit will be the physical resource block (PRB, or just RB), as it is the minimum scheduling unit, defined as a set of frequencies over a time lapse. In 5G-NR, the PRB is formed by 12 subcarriers during a time slot. Note that both the subcarrier bandwidth and the time slot duration depend on the numerology: there are 5 different possibilities, ranging from 15kHz to 240kHz, and from 1 ms down to 0.0625 ms.

The connection (or association) policy, which is the object of our studies, is usually implemented through a biased maximization of the signal quality indicators, subject to minimum-level constraints. A second policy, executed within each base station, governs resource allocation among already-connected users. In this first part we address the association problem (user–base-station assignment) and defer intra-cell resource allocation to the second part of the thesis.

2.4 System Model and Problem Statement

We will now introduce the system model and problem statement: how are we going to describe what we know, and what are we looking to optimize.

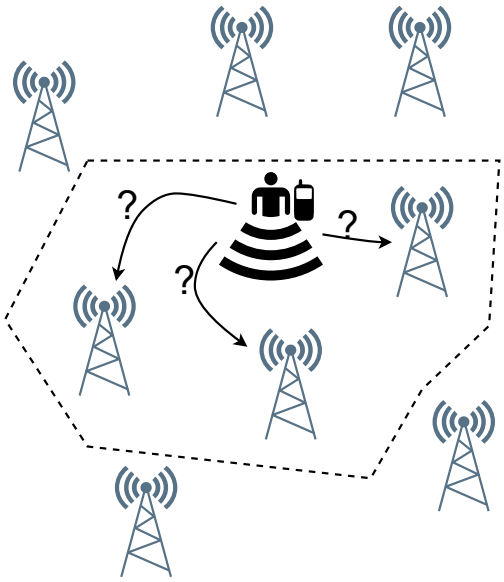


Figure 2.1: We consider the system formed by the available base stations to the arriving user.

Let $u \in \mathbb{N}$ be an index representing a set of users that arrive to the system at time t_u^i. Time can be either discrete (slotted) or continuous, although we will assume the former for clarity. The system consists of a set of base stations indexed by $b \in \{0, \ldots, N\}$, and the problem is choosing to what base station each incoming user will connect to, if accepted (index $b = 0$ will denote a rejection). While base stations positions and capacity is fixed, users' loads and positions are randomly picked, and their maximum rates with each base station is therefore also stochastic. Furthermore, we will assume that no handover is possible and users are static, so that this choice will be definitive throughout the complete connection of user u. This is a reasonable assumption when users have relatively short connections.

Users arrive with a certain load l_u that they want to transmit. If the user is connected to base station b at time t, it will obtain a rate of $r_b^u(t)$ during that time-slot (i.e. their pending load will decrease by $r_b^u(t)$ per slot), which will depend on the SINR between b and u, but also on the other users concurrently served by base station b. As base stations have a finite set of resources to distribute, the achieved user rate will depend on the number of connected users and the policy in use (e.g. round-robin or equally shared spectrum). Once the pending load goes to zero, the user disappears and the corresponding time will be denoted as t_u^e.

We will observe a certain time-window $[0, T]$ where a sequence of users $\mathcal{U} = \{0, \ldots, U\}$ entered the system (i.e. $0 \leq t_u^i \leq T$ for $u \in \mathcal{U}$), and we define a so-called *performance function* $f(\mathcal{B})$ which measures how good was the corresponding sequence of chosen base stations for each user $\mathcal{B} = \{b_0, \ldots, b_U\}$. The problem then becomes:

$$\max_{\mathcal{B} \in \{0,\ldots,N\}^U} f(\mathcal{B})$$

Note that any constraint on the system may be easily included in the performance function; e.g. if a certain sequence of decisions leads to a number of users connected to a base station that surpasses its capacity, then $f(\mathcal{B}) = -\infty$. However, it is practical to sometimes include these as constraints in the optimization problem. In this thesis, we will consider a constraint regarding the maximum number of connected users per base station.

We also dismiss (for now) the study of the internal resource distribution policy, which we denote as $\Phi(b, t)$. The actual throughput achieved by user u will be a function of both the user association and the internal distribution policy. This policy can be implemented through proportional shared spectrum, round robin, proportional fair, or other. For simplicity to tackle the UA problem, we assume from now on that this policy is (a) the same for all base stations, and (b) only dependent on the number of users (and not on their rates, for instance). Although we will not specify it, this policy can be viewed as a restriction on our optimization problem.

We will leave in a parenthesis the choice of the performance function, as in the next sections we will consider different optimization goals. From a practical point of view, the problem we strive at solving is sequential. That is to say, we have to decide to which base station (if any) connect each new user as they arrive to the system, and to take each of these decisions we can only observe the current (or past) state of the system. The optimal policy for this problem is, however, computationally intractable. Even in simplified user association models, finding the assignment that maximizes a global utility is known to be NP-hard [54], as it resembles variants of multi-dimensional knapsack, scheduling, and partitioning problems. Moreover, the state space grows combinatorially with the number of users, their loads, their channel conditions, and the occupancy of each base station. Exhaustively evaluating all possible association sequences $\mathcal{B}$ quickly becomes impossible even for moderate system sizes, since the number of feasible trajectories over a time window scales exponentially. As a result, exact optimization or dynamic programming approaches are not practical, which motivates the use of approximate, data-driven strategies better suited to the sequential nature of the problem. Such problems can be formally stated and studied within the framework of Markov Decision Processes (MDPs) and Reinforcement Learning (RL).

Remark 1 (On the user association model) Although the most direct application of this user association modeling is in next-generation cellular networks, which combine adaptive resource management with large-scale deployments, the proposed framework can be extended in a straightforward manner to other networked systems, such as Flying Ad-Hoc Networks (FANETs), as demonstrated in [35]. Moreover, our model naturally supports decentralized execution, which is essential in large-scale and highly dynamic networks as FANETs where centralized coordination is impractical or incurs prohibitive signaling overhead.

In the next chapter we start by introducing a naif approach to applying reinforcement learning to the user association problem. We develop a simple yet

functional RL-framework with which we are able to adapt well known algorithms to find suitable policies, and we build a first graph representation that exploits the benefits of graph neural networks to approximate the large state space.

Later, in chapter 4, we develop a more sophisticated modeling, in which our system's actions allow for strategic user rejection and achieve decentralized execution, and modify the RL definitions to tackle slot-time fairness.

We enhance our RL framework in chapter 5 in order to account for a more comprehensive fairness definition. We formalize the MDP underlying our system's state evolution and define a reward collection process that integrates fairness into the RL without the need for reward engineering or specific algorithm tailoring. We finish this chapter and close this first part by summarizing our proposals and discussing some design choices.

This page has been intentionally left blank.

Chapter 3

A naif Reinforcement Learning approximation

As we presented in the last section, we are looking for a series of actions that maximize a performance function. These problems can be addressed through Reinforcement Learning, which we will formally introduce next. Reinforcement Learning refers to a family of learning algorithms belonging to the machine learning world, in which an agent learns from interactions with the environment, choosing actions and receiving feedback (reward). By balancing exploration (new actions) and exploitation (known actions with high returns), the agent is able to learn from trial and error a series of actions in order to maximize the achieved reward through time.

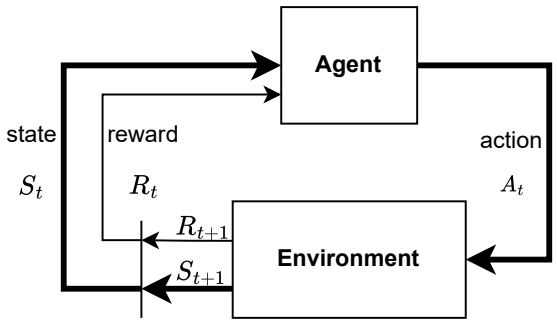


Figure 3.1: Classic RL interaction: the agent observes the environment's state, takes an action, and receives a reward, observing the environment's evolution to the next state.

Reinforcement Learning: a (micro-)Introduction

As seen in figure 3.1, we consider an agent taking action A_t over an environment in a present state S_t, which results in an update of the environment's state S_{t+1} and a reward R_t for the agent. The RL framework requires the agent-environment system to be defined by the elements $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$: state, action, transition, reward and discount factor.
A policy establishes the correspondence between state and action: $\pi(s) = a$ if deterministic, or as a probability over actions if stochastic: $\pi(a|s) = P(A_t = a|S_t = s)$.
The RL goal is to find the policy that maximizes our cumulative return. Usually, the target is the discounted cumulative reward G_t.

$$G_t = \Sigma_{k=0}^{\infty} \gamma^k R_{t+k+1}.$$

Given the stochastic nature of the problem, we whish to maximize its expectation following policy π. We denote by V_π the state-value function following policy π and by Q_π the action-value function following policy π:

$$V_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s].$$
$$Q_\pi(s, a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a].$$

These value functions can be optimized by updating an initial policy through one of Bellman's equations, as the action-value function for policy π, defined as [93]:

$$Q_\pi(s, a) = \mathbb{E}_\pi\left[\Sigma_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s, A_t = a\right]$$

which can be updated until convergence following the rule given by the optimality equation for the state-action value function:

$$Q_\pi(s, a) = r + \gamma Q_\pi(s', \pi(s'))$$

The mathematical structure guaranteeing convergence of RL algorithms require that the environment's state evolution satisfies the Markov property, that is, it can be represented as a well-defined Markov Decision Process (MDP). As we will see in chapter 5, MDPs can grow very large in order to fully represent the environment's state. This is challenging as RL needs exploration, for which states have to be visited and revisited, which is unfeasible for a large MDP. There are several approaches to cope with this complexity: some methods learn or approximate the state representation by using parametric models expected to extend to unvisited states; others operate under partial observability; and model-free methods avoid learning a transition model and rely only on observed states, actions, and rewards. A very interesting discussion on whether full-state representations are necessary, and on the central role of the reward signal, is given by [94] and its responses [95].

In our approach to the user association problem, we address the aforemen-

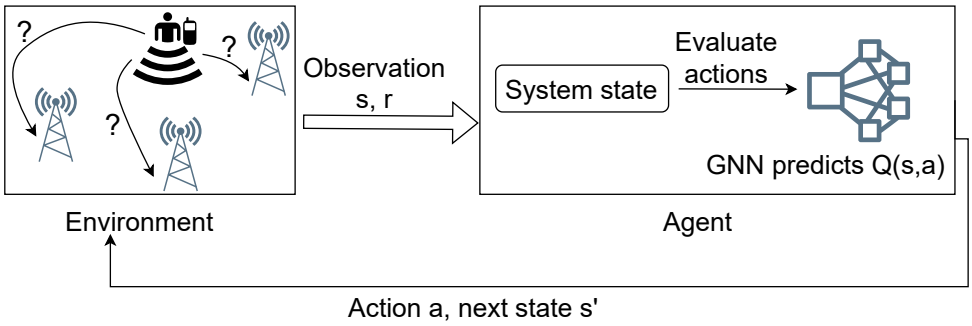


Figure 3.2: System model. We consider at most one arrival at each time-step. The choice of which base station associates with the currently arrived user is the action. After executing an action, a new state is observed and a reward is obtained. The GNN, introduced further ahead, approximates the action-value function.

tioned challenges using a Partially Observable Markov Decision Process (POMDP) framework. In a POMDP, the underlying system state evolves according to a well-defined Markov process, but the agent has access only to partial observations of this state. This reduction in state information generally leads to the loss of theoretical guarantees for optimality [93]. Nevertheless, reinforcement learning agents are still able to learn stable and effective policies in practice. Through the exploration–exploitation trade-off and the reward signal, the agent iteratively updates its policy to maximize the expected cumulative discounted reward, potentially converging to a suboptimal but robust solution.

In our case, and as a first exploration, we focus on building a reinforcement learning framework that will enable tackling our user association problem. Our RL model is then schematized by figure 3.2, and we next define the state s, action a, reward r, transitions P and discount factor γ.

We start by defining the **system's state** s. We consider a combination of the incoming user's characteristics and the base station's states: in the base stations we carry information of past arrivals and actions, and with the incoming user's information we complete the system state.

We describe the applicant's characteristics by including the rate per physical resource block $\overline{r_b^u}$ with each candidate base station b, and a certain load l_u to be satisfied. As the user's position is randomly selected, the signal quality indicators (depending on the signal strength with each base station) is random, and therefore the achievable rate is also stochastic. The load is randomized as well, which makes the user's contribution to the state completely stochastic.

Remark 2 (On knowing the user load) Although very used in the literature, the inclusion of the users' load into the state might be subject of discussion. In 5G, users do not advertise explicitly their expected load during connection for uplink, but nevertheless base stations do infer resource demand through the Buffer Status Report (number of queued bytes for transmission). For downlink, the base station already knows the load to deliver.

For each base station b, we define the state $s_b(t)$ as the number of connected users ($n_b(t)$) and their averaged rate ($\widehat{r_b}(t)$). We thus have that the state s_t is constituted by the following $2 + 2 \times B$ dimensional vector[1]:

$$s_t = [\overline{r_b^u}, l_u] \parallel_{b=1}^{N} s_b(t)$$

$$s_b(t) = [n_b(t), \widehat{r_b}(t)]$$

Although one could consider this state representation as too simple (i.e. missing out important information), it will be handy to consider a low dimensional state space to begin developing our solution.

As rejections will not be allowed in this first model, the **action** a is to select one of the possible base stations, $a \in [1, ..., B]$. Please note that not every time step involves actions: they may or may not occur, depending on the arrival of a user. To prevent adding further confusion to the system, we include the decision-making in the state, by setting the demand to 0 for the time steps on which no user arrives. In this case, only one course of action is available: when no users arrive there is no action to be taken, only updating the system's state.

Transitions occur as depicted in figure 3.2: the base station's descriptors have to be updated to include the action taken (+1 on the number of associated users, new average rate), and the effect of departures ($-n$ on the number of users associated for n users' demand being satisfied). Each time step will update the incoming user's rate and load in the event of a new arrival. Note that transitions are deterministic over the base station's features given the action a and the state s, but stochastic for the new user's features. If no user arrives (and therefore no action is taken), the transition is deterministic.

Finally, the **reward** r is defined as the instantaneous utility of executing an action for a given state. Following the literature [88,96], and as a means to promote fairness in the distribution of resources, we take as utility/reward the log-sum of the user's rate, $r_t = \sum_b \sum_u \log(1 + \overline{r_b^u}(t))$. We will discuss with more detail the reward function in the following chapters.

The parameter $\gamma \in (0, 1)$ is the discount factor. It controls the relative importance of future rewards with respect to immediate ones, allowing the decision-maker to trade off short-term and long-term performance. From a theoretical perspective, the discount factor is also essential for the convergence of many reinforcement learning algorithms, as it ensures that the expected discounted cumulative reward is finite and bounded.

Some final comments are in order for the system to be completely defined. First, the system has no queuing of users: we can only accept the incoming user, except if no resources are available (i.e. at least one physical resource block). Secondly, even if the model is presented as sequential on the arrival of users, this isn't a far fetched simplification: to have users arrive exactly at the same time is a more unrealistic situation. Third, the episode is defined as a certain (fixed) number of steps. Last, we can limit the complexity of our action and state spaces by

[1]We utilize the symbol $\parallel$ to represent vector concatenation.

considering only the k base stations with available resources and higher normalized rate, as in [97].

Summarizing, the RL formulation considered in this work is an episodic, discounted, infinite-horizon POMDP with stochastic transitions and discrete underlying state and action spaces.

3.1 Approximating the optimal User Association policy

We described the action-value function in our very brief RL introduction above: it relates the policy π our agent follows with our expected cumulative return. We will now develop on the policy optimization: how do we explore actions and use this information in order to make better choices on the future. We start by refreshing the definition of $Q_\pi(s, a)$:

$$Q_\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t \,\middle|\, s_0 = s, a_0 = a \right]$$

In order to find the optimal policy, the agent updates an initial random policy through one of Bellman's equations. In our case, as we consider Q_π, the Bellman equation for the action-value function is defined as [93]:

$$Q_\pi(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) \sum_{a' \in A} \pi(a'|s') Q_\pi(s', a')$$

Which leads to the Bellman optimality equation for the Q-value function:

$$Q_*(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) \max_{a' \in A} Q_*(s', a')$$

In tabular reinforcement learning methods as Q-learning, the action–value function is updated according to the Bellman optimality equation,

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$$

When the state and action spaces are finite and sufficiently small, so that all state–action pairs are visited repeatedly, this update rule is guaranteed to converge to the optimal action–value function under standard conditions on exploration and learning rates, thereby yielding an optimal policy.

When dealing with large state spaces in which revisiting all possible states is infeasible, standard approaches such as Deep Q-Networks (DQN) rely on function approximation of the action–value function. In this setting, the Q-function is approximated by a parametric model $\hat{Q}(s, a; \theta)$, typically implemented as a neural network with parameters θ learned from data.

Because the agent operates on an approximated and reduced representation of the state space, the theoretical guarantees of convergence to the optimal action–value function are generally lost. Nevertheless, by iteratively updating the Q-function to maximize the expected reward, DQN-based algorithms are able to

converge in practice to stable policies, which may be suboptimal but are often effective in complex environments.

In our case, as explained previously, we do not whish to keep track of the complete state, so we consider our system to be a partially observed MDP and learn the Q-function from a simpler state representation. This leads our approximation to its final form:

$$Q_*(s, a) \approx \hat{Q}(\hat{s}, a; \theta^*) \tag{3.1}$$

Our goal is to approximate the optimal policy through Deep Reinforcement Learning. We adopt deep neural networks as function approximators, and in particular employ a Graph Neural Network to model the structured interactions among base stations and users. This results in a DRL algorithm capable of capturing both the sequential nature of the decision process and the underlying graph topology. The architecture and training methodology are detailed in the next sections.

3.1.1 Algorithm Design

As stated, we refer to our combination of DRL+GNN for UA as GROWS. GROWS is based on the classic Double DQN reinforcement learning algorithm [98], an extension of Deep Q-Learning that uses two Q-functions to stabilize learning: a predictor and a target. The predictor network is updated more often, while the target network is updated more slowly. This decoupling of action selection and value estimation reduces the overestimation bias that occurs when both are computed with a single network, yielding more accurate Q-value targets and a more stable value-based learning process that improves convergence. A simplified pseudo-code of our algorithm can be found in Algorithm 1.

Integration with the GNN is inspired on previous work [97, 99]. The goal of the GNN is to learn how to best approximate the Q-function, an estimation of the value-action function for the RL problem. It is important to notice that training and execution are done separately: once the GNN is trained, prediction of the Q-function according to the state and possible actions can be done instantly. We will now develop on the use of Graph Neural Networks as Q-function approximator.

Algorithm 1 Simplified Double DQN pseudo-code

Initialize predictor network $\hat{Q}(s, a)$
Initialize target network $\hat{Q}^{-}(s, a)$ with $\hat{Q}^{-}(s, a) \leftarrow \hat{Q}(s, a)$
Initialize replay buffer
for episode e **do**
  while not done **do**
    User arrives, state s
    if ϵ-greedy exploration **then**
      Sample random action a
    else
      $a = \arg\max_a \hat{Q}(s, a)$ ▷ selection: predictor
    end if
    Observe reward r and next state s'
    Store (s, a, r, s') in replay buffer
    if update step **then**
      Sample batch $\{(s_i, a_i, r_i, s'_i)\}$ from replay buffer
      For each sample:
        $a^*_i = \arg\max_{a'} \hat{Q}(s'_i, a')$ ▷ selection: predictor
        $y_i = r_i + \gamma \hat{Q}^{-}(s'_i, a^*_i)$ ▷ evaluation: target
      Update $\hat{Q}$ by minimizing $(y_i - \hat{Q}(s_i, a_i))^2$
    end if
    if target update **then**
      $\hat{Q}^{-} \leftarrow \hat{Q}$
    end if
  end while
end for

3.1.2 A first Graph Representation

Graph Neural Networks (GNN) have attracted significant attention since their early formulation in [100] due to their ability to exploit the relational structure of graphs for distributed information exchange and representation learning. They support decentralized computation, naturally handle variable-sized and irregular topologies, and generalize well to unseen configurations. In this thesis we consider the Graph Convolutional Networks (GCN), a specific class of Graph Neural Networks. Conceptually, a GCN is composed of a sequence of layers in which each layer applies a graph filter, responsible for aggregating information from neighboring nodes, followed by a nonlinear activation.

A quick introduction to Graph Convolutional Networks

Consider a graph for which each node has an associated vector $\mathbf{x}_i \in \mathbb{R}^d$ (for $i = 1, \ldots, N$), which may regarded as the input features. Making the analogy to discrete-time convolution, a first-order convolutional layer for a GNN may be obtained as follows [101]:

$$\mathbf{x}'_i = \sigma\left(\mathbf{\Theta}^T \sum_{j \in \mathcal{N}_i \cup \{i\}} S_{j,i}\mathbf{x}_j\right), \tag{3.2}$$

where $\mathbf{x}'_i \in \mathbb{R}^{d'}$ is the output of the layer, $\sigma(\cdot)$ is a point-wise non-linearity (e.g., the ReLU function), $\mathbf{\Theta} \in \mathbb{R}^{d \times d'}$ is the learnable parameter of this layer, $\mathcal{N}_i$ is the set of neighbors of node i, and $S_{i,j}$ is the i, j entry of matrix $\mathbf{S} \in \mathbb{R}^{N \times N}$, the so-called Graph Shift Operator (GSO). This is a matrix representation of the graph, which should respect its sparsity (i.e. $S_{i,j} \neq 0$ whenever there is an edge between nodes i and j). The adjacency matrix of the graph, its Laplacian or their normalized versions are all valid GSOs.
Note that in (3.2) each node needs to linearly combine the vectors of its neighbors only. As we concatenate K such layers, the final vector representation of node i (i.e. the output of the GNN) will depend on its neighbors up to K hops away. This observation implies that a GNN may be implemented in a fully-distributed way, as long as an edge in the graph means that the corresponding pair of nodes can communicate.
We may be more general and build higher-order filters. Let us stack all nodes' vectors $\mathbf{x}_i$ into matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$, which is called a graph signal. The matrix product $\mathbf{SX} = \mathbf{Y}$ results in another graph signal, corresponding to the first-order convolution we used in (3.2) (albeit without parameter $\mathbf{\Theta}$, which we will include shortly). By writing $\mathbf{S}^K\mathbf{X} = \mathbf{S}(\mathbf{S}^{K-1}\mathbf{X})$ we may see that in this way we aggregate the information K hops away. Again, although it requires K rounds of information exchange, this operation may be performed without intervention of a central entity.
Finally, a general graph convolution is defined simply as a weighted sum of these K signals (i.e. $\sum_k \mathbf{S}^k\mathbf{X}h_k$, where scalars h_k are the taps of the filter). In this context, parameter $\mathbf{\Theta}$ in (3.2) is interpreted as a filter bank. That is to say, by considering a $d \times d'$ matrix $\mathbf{H}_k$ instead of the scalar taps, a single-layer GNN (or graph perceptron) is obtained by applying the pointwise non-linear function $\sigma(\cdot)$ to this convolution [102] [103]:

$$\mathbf{X}' = \sigma\left(\sum_{k=0}^{K-1} \mathbf{S}^k\mathbf{X}\mathbf{H}_k\right), \tag{3.3}$$

whereas a deep GNN is constructed by concatenating several perceptrons. Note that a bias may be easily included by adding a matrix $\mathbf{b} \in \mathbb{R}^{d'}$.

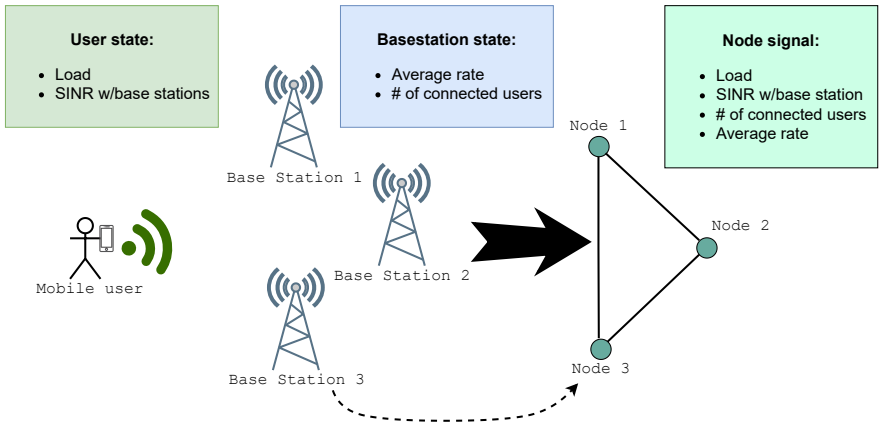


Figure 3.3: Graph representation of the system. Nodes represent base stations, and edges represent connections between these base stations. Each node signal is composed of the base stations' state and the incoming user's load and rate.

For the graph representation, at each decision time we construct a graph whose nodes correspond to the base stations involved in evaluating the newly arrived user. Edges represent exchanges between base stations. Because the considered 5G scenario assumes negligible backhaul cost, all base stations can freely exchange information, resulting in a fully connected graph. As illustrated in Fig. 3.3, the state associated with each node comprises both the current state of the corresponding base station and the features describing the user relative to that base station.

An interesting remark has to be made about the graph. When a user has strong enough signal with all base stations, the graph resulting is just the k possible nodes connected among themselves. But as the system evolves and base stations start to be depleted of resources or the new user only sees a few base stations, the graph will be taking different shapes. This evolving graph representation provides adaptability to different scenarios, all while allowing for the information passing between neighboring nodes, a key aspect of GNNs.

For the GNN model, we adopt the *LocalGNN* architecture proposed in [104], which implements the graph convolutional network (GCN) introduced earlier. A key property of this architecture is that the output associated with each node can be computed using only information from its k-hop neighborhood through localized message passing. This locality enables scalable and decentralized execution of the proposed algorithm. In Chapter 4 we enhance our proposal to fully profit its decentralized capabilities.

A final merge of the Q-function prediction for choosing which base station serves the user has to be made. This can be either decentralized (by exchanges between the base stations of their expected Q-function values), or centralized (if a single entity receives the states updates and calculates the maximum expected reward for the possible actions).

3.2 Validation of the GROWS Framework

In the follow sections we describe two sets of experiments: a small mobile network topology and a Flying Ad-hoc Networks deployment. For the sake of reproducibility we share this first implementation, see gitlab.fing.edu.uy/mrandall/grows.

3.2.1 A simple experiment with 3 base stations

Baselines

To evaluate the behavior and performance of this first version of GROWS in a mobile network use case, we use synthetic scenarios simulating 5G networks. As explained before, the goal is to learn a UA policy which maximizes the log-sum of the users' rate, potentially improving the overall system utility as compared to certain baseline policy. As stated, the optimal policy is intractable: the problem is NP-hard and the number of states grows too fast to run a search over all possible decisions. For this reason, UA in current mobile networks is generally done through a simple heuristic, selecting the BS with the highest signal strength. We refer to this policy as an *argmax* policy, and we will consider it as the *baseline*. Even if simple, this strategy achieves good performance, and is usually considered as baseline in the field [54, 105–111]. We will compare GROWS against this simple policy, representing the currently followed strategy.

In order to address the UA problem through a pure RL approach, and compare GROWS against a tabular *Q-learning* baseline, we consider a small scenario. In such settings, Q-learning can approach the optimal policy when states and actions are sufficiently explored, providing a meaningful reference point for evaluation. However, this method becomes infeasible in more realistic scenarios, where the number of states grows rapidly and cannot be enumerated or stored. In contrast, the DRL approach used by GROWS remains viable, as the deep function approximator scales to large state spaces. This comparison in controlled environments thus highlights both the accuracy and the scalability advantages of GROWS over classical tabular RL.

Experiment settings

We simulate a 5G network in which we consider that base stations are interconnected through a backhaul. The scenario consists of three base stations, each disposing of 5 physical resource blocks. At each time step t, a user arrives with probability p. We will compare the algorithm's performance with values from the set $p \in \{0.5, 0.7, 1\}$. Users have three possible downlink demand values to be satisfied: *low* (2 Mbits), *medium* (10 Mbits), and *high* (20 Mbits). These load values are taken following a probability distribution, with which we can impose either heavier or lighter demands.

Users are associated with a discrete set of achievable data rates for each base station, that are determined by the received SINR and physical layers choices, as explained in section 2.3. In this very first approximation to our problem, we sim-

plify the synthetic scenario by establishing a set of possible rates and randomly picking the resulting rate with each base station. We define probabilities for each possible rate, and one of the base stations has a slightly better chance to provide a higher rate to incoming users. As we advance in our experiments on the next chapters, synthetic users will be generated with geographically grounded location, and their possible data rates will be estimated through classical wireless communication formulas. For now, we consider four possible per-PRB data rates: 360 kbps, 720 kbps, 1.08 Mbps, and 1.44 Mbps.

Episodes consist of $T = 40$ time steps. Action selection follows an ε-greedy exploration-exploitation strategy, with exploration emphasized during the first 40,000 episodes and exploitation during the final 10,000 episodes, according to an exponentially decaying ε.

The hyperparameters of the LocalGNN model are tuned by grid search, resulting in a learning rate of 10^{-4}, a batch size of 32, and a lightweight architecture composed of two graph convolutional layers each considering information from 1-hop neighbors, with which we obtain a prediction per node. The hyperbolic tangent is used as the pointwise nonlinearity, and to mitigate vanishing or exploding gradients, the GCN inputs are normalized.

The policy network is updated every 20 steps, while the target network is updated every 200 steps, using an experience replay buffer containing 1×10^4 samples. The discount factor is set to $\gamma = 0.5$ to accelerate convergence in this simplified experimental setting. The Q-learning update employs a learning rate of 0.5. Finally, the action-value function is initialized to zero and policies are initialized randomly for all reinforcement learning algorithms.

Experiment analysis

To analyze how GROWS behaves in different user/traffic conditions, we simulate different network-load scenarios, varying the average user load ($\overline{\mathcal{D}}$) and the arrival rate (p). We assess performance in terms of the mean utility and the mean number of user rejections realized for the different experiments, comparing the three different approaches: ***baseline*** (**B**), ***Q-learning*** (**Q**), and **GROWS**. Results are summarized in Tables 3.1 and 3.2. For the small topology and experimental settings, the Q-learning policy is able to explore the system states enough to improve utility, but the GROWS algorithm still achieves better results for many scenarios, meaning the GCN was able to learn a good approximation of the Q value function. When demand is low, there are very few rejections, and all algorithms achieve similar results. However, as mean demand increases, GROWS is able to increase the gained utility over the baseline, proving its ability to handle traffic over more stressed situations. Regarding both mean utility and user rejection, either GROWS or the Q-learning fare better in almost all scenarios. In some cases, GROWS is able to reject a 20% less of users.

It is interesting to observe that the pure Q-learning algorithm is able to achieve better results (i.e. find a closer to optimal policy) as p grows. More frequent arrivals involve more actual decisions per episode, thus a higher exploration and a better approximation to the optimal Q-function. When the exploration over visited

	Mean Utility per Episode								
	$p = 0.5$			$p = 0.7$			$p = 1$		
$\overline{\mathcal{D}}$	**Q**	**B**	**GROWS**	**Q**	**B**	**GROWS**	**Q**	**B**	**GROWS**
6	2.67	2.64	**2.70**	**2.94**	2.86	2.92	**3.39**	3.15	3.24
8	2.81	2.73	**2.87**	3.07	2.93	**3.08**	**3.50**	3.20	3.42
10	**2.86**	2.76	2.80	3.15	2.96	**3.16**	**3.56**	3.21	3.54
12	2.92	2.77	**2.95**	3.19	2.96	**3.23**	**3.57**	3.21	3.55
14	2.92	2.78	**2.99**	**3.20**	2.95	**3.20**	**3.57**	3.21	3.52

Table 3.1: Mean utility per episode for different experiments, varying average demand ($\overline{\mathcal{D}}$) and arrival rate (p).

	Mean Number of Rejected Users per Episode								
	$p = 0.5$			$p = 0.7$			$p = 1$		
$\overline{\mathcal{D}}$	**Q**	**B**	**GROWS**	**Q**	**B**	**GROWS**	**Q**	**B**	**GROWS**
6	0.01	0.01	**0.01**	0.25	0.26	**0.34**	2.86	3.12	**2.65**
8	**0.08**	0.09	0.09	**0.99**	1.07	1.10	5.59	5.63	**5.35**
10	0.24	0.26	**0.21**	1.93	2.00	**1.77**	7.04	7.32	**7.03**
12	**0.45**	0.52	0.48	2.75	2.86	**2.66**	8.34	8.38	**8.17**
14	0.69	0.83	**0.63**	3.44	3.62	**3.38**	9.15	**9.08**	9.13

Table 3.2: Mean user rejections per episode for different experiments, varying average demand ($\overline{\mathcal{D}}$) and arrival rate (p).

states is not enough, our GNN implementation achieves better results, meaning a better approximation to the optimal policy.

Figure 3.4 reports the obtained results in terms of utility for one of the experiments ($p = 0.5$ and average demand $\overline{\mathcal{D}} = 14$), where the cumulative reward is averaged every 300 episodes. For a better visualization and interpretation of results, utility is normalized to a random UA policy, where users are assigned to BSs randomly; this means that a value of 1 on the normalized utility is equivalent to a random UA policy. Results are encouraging: this version of GROWS learns a policy that outperforms the *argmax* heuristic significantly (by more than 10% in this set up). During the first episodes, exploration is still dominant for both GROWS and Q-learning, and the greedy exploration/exploitation policy is strongly noticeable. After 1,000 episodes, GROWS learns a better UA policy with the same exploration as Q-learning, suggesting it was able to quickly approximate the Q-value function, as expected.

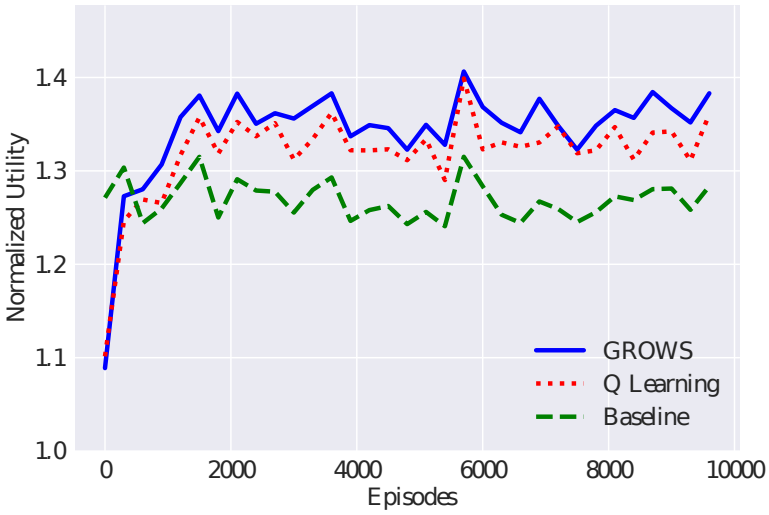


Figure 3.4: Already in a small mobile network topology (three base stations) with a simple traffic demand, GROWS outperforms current UA policies. The more complex the scenario, the highest the benefits we expect from GROWS as compared to the baseline *argmax* approach.

3.2.2 Applying GROWS in a distributed scenario: Flying Ad-hoc Networks

Flying Ad-hoc Networks: FANETs

Consider the scenario where a group of UAV are deployed in order to serve as a complementary or substitute communication infrastructure (often referred to as flying ad-hoc networks - FANETs) [112]. In order to satisfy stringent deployment situations as an emergency backup network (e.g. floods, hurricanes), the optimization of available resources plays a key role.

Regarding user association for an UAV deployment, most of the existing work focus on the problem of UAV placement in order to, for instance, maximize coverage [113–115]. We will assume that this deployment phase has already taken place and we focus on how to associate users to each UAV base station. In this case, most existing works propose a centralized scheme or assume complete knowledge of all nodes [112, 116, 117]. On the other hand, the proposal most similar to ours is the distributed approach based on multi-agent learning described in [111]. However, and very differently to our work, each base station may only use a single subcarrier per time slot (i.e. only one user is served per slot), a strong restriction they impose mostly for scalability purposes.

For a FANET deployment, the distributed nature of GNNs is vital to the algorithm's implementation: the real advantage of using a graph-based learning method comes in a scenario where no central entity may be assumed to exist, and base stations can only exchange information with their neighbors. Actually, all deployments where the infrastructure is provided by an ad-hoc network fall into this scenario. We will in this section consider GROWS behavior and performance

in a Flying Ad-hoc Network (FANET) deployment use case.

The main difference between the 5G deployment and the FANETs use cases is the graph construction. In a FANET there usually is no central entity or backhaul 'cost free' communication among nodes, so the resulting graph will not be the fully connected graph as in the 5G model. This results in different graphs according to which neighbors each UAV perceives.

To carry out the experimental evaluation, and very similar than in last section, we build a synthetic scenario for evaluations. As in the past experiment, we compare GROWS against the classic *argmax* policy, representing the currently followed strategy, and against our classical RL implementation with *Q-learning*.

Experiment settings

Regarding parametrization of the experiments and testing conditions, the following list describes the different components. We consider four BSs randomly located on a plane, with the condition of being each of them connected to at least another BS; i.e. all UAVs have at least one connection to another UAV.

Regarding user arrivals, at each time step t a new user arrives with probability p, where $p \in \{0.5, 0.7, 0.9\}$. Different than in the previous experiment, users are now uniformly distributed over the plane, subject to the constraint that each user is within coverage of at least one base station, defined by a minimum downlink SINR sufficient to support QPSK modulation. As users are now geographically grounded, their possible rates with each base station will be based on a selected 5G physical layer configuration, and estimated using the Friis propagation model, as is standard in these experiments. For this experiments we consider shared frequency and modulation settings consistent with 4G and 5G systems, namely a 20 MHz bandwidth with 15 kHz subcarrier spacing over a 3 GHz carrier, assuming FDD operation, no MIMO, and ideal modulation without coding overhead. Depending on the SINR, the selected modulation is QPSK (-95 to -85 dB), 16-QAM (-85 to -75 dB), 64-QAM (-75 to -65 dB), or 256-QAM (above -65 dB). Each physical Resource Block occupies 180 kHz, resulting in four possible per-PRB data rates: 360 kbps (QPSK), 720 kbps (16-QAM), 1.08 Mbps (64-QAM), and 1.44 Mbps (256-QAM).

As in previous experiments, users have three possible downlink demand levels—2, 10, and 20 Mbits—drawn according to a predefined probability distribution. By varying this probability distribution we obtain different values of the average traffic demand for an incoming user, denoted by λ, which takes values in $\{7.2, 10.4, 14.4\}$.

Episodes consist of 100 time steps, and action selection follows an ε-greedy exploration–exploitation policy. Exploration is emphasized during the first 15,000 episodes, after which the policy gradually shifts toward exploitation over the final 5,000 episodes according to an exponentially decaying ε. Hyperparameters are calibrated using Bayesian optimization implemented via the `Optuna` library. A relatively low discount factor, $\gamma = 0.5$, is adopted to accelerate convergence in this experimental setting.

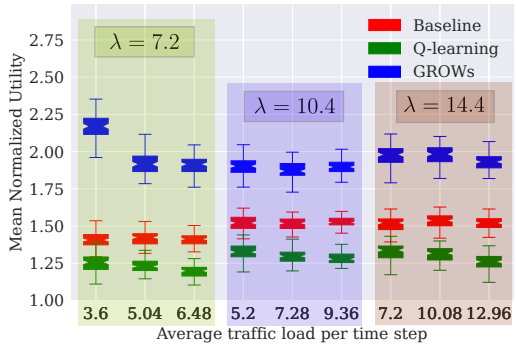


Figure 3.5: Mean normalized utility per episode for the compared algorithms over different expected traffic loads and arrival rates. GROWS clearly outperforms both the baseline as the traditional q-learning algorithm.

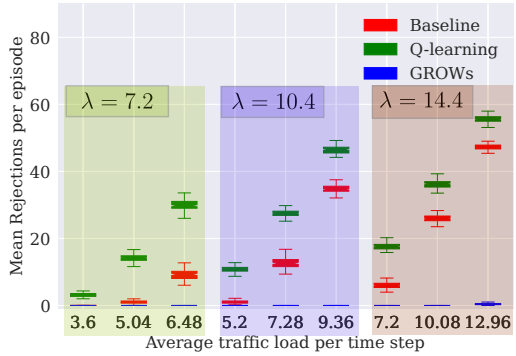


Figure 3.6: Mean rejections per episode for different traffic loads. For the same expected traffic demand λ, user rejections increase with higher user arrivals (0.5, 0.7 and 0.9), except for GROWS, which is able to accept almost all users at every episode.

Experiment analysis

To study the operation of GROWS under different traffic loads, we compare results for different combinations of p and λ. Figure 3.5 reports the obtained results in terms of utility for nine different UA policies, learned for different *traffic loads*. The x-axis corresponds to $p \times \lambda$, where the cumulative reward is averaged every 50 episodes over the 5,000 exploitation episodes, resulting in 100 values reported as boxplots.

Additionally, Fig. 3.6 reports the mean number of rejections per episode. GROWS is not only able to improve utility as compared to the other strategies, but most importantly, it does so while achieving a close to null rejection rate, even in highly congested scenarios. The steep curve for each λ value follows the user arrival rate increase. GROWS realizes an increase of 30% to 50% over the baseline utility, accepting almost all users, while the baseline has rejection rates up to 50%.

As a remark, the tabular Q-learning algorithm does not perform well in these experiments. This behavior can be attributed to the rapid growth of the state space with the number of nodes and resource blocks, which renders exhaustive

exploration impractical. However, the performance of tabular Q-learning is not the focus of this study. The algorithm is included for completeness, and was used in earlier validation stages to test the approximation the Q-function in very simple experiments, in which the algorithm could effectively revisit all possible states and actions. Without dedicated hyperparameter tuning or extended training, its lack of convergence to the optimal policy in this setting is therefore not unexpected. There is another important reason that explains a slow convergence for our proposed algorithms: since we collect observations and reward at every time slot, there is potentially misleading information regarding effectively taken decisions and the system's transitions. We will address this issue in the next chapter, by only considering observations on the decision instants.

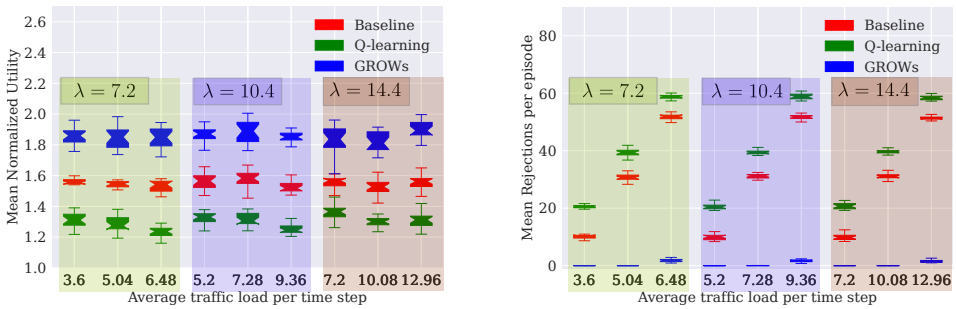
Figure 3.5 also reveals that utility is mostly dependent on the expected traffic load λ. Once users arrive and get connected to a BS, utility remains rather stable, but as traffic load increases, connections last longer and new arrivals are rejected if no resources are available. Following this reasoning, arrival rates p will have a great impact in the number of rejections per episodes, which is clearly observed in Figure 3.6. Indeed, for the same expected load, the rejection rate increases directly with the increase of arrival rates. GROWS UA policy counterbalances this performance degradation through a smarter assignment of available resources, resulting in an almost negligible user rejection rate.

Performance when facing changing traffic loads

An important characteristic for an AI/ML driven system is being able to cope with co-called Concept Drifts (CDs) in the analyzed data; i.e., modifications in the properties of the underlying probability distribution. A CD can manifest itself as a shift in the mean, an increase or decrease in the variance, or even as complete data modifications. For the specific problem of UA through learning, a desirable property is being able to handle a significantly higher load of traffic and users, not seen at training time, where the standard heuristics such as *argmax* would saturate (cf. Figure 3.6). For example, traffic load could surge in the event of flash-crowds, during emergency situations, or even due to major social events requiring higher connectivity and access resources .

To analyze how GROWS behaves in the event of a strong surge of traffic load, we explore its application to higher traffic loads not observed at training time. More specifically, we take the nine UA policies learned for the different values of p and λ (cf. Figures 3.5 and 3.6), and apply them to a significantly higher user demand, taking $\lambda = 16.2$. For reference, this value represents a traffic demand surge of $\times 2.25$, $\times 1.56$, and $\times 1.13$ for the UA policies learned with the original λ values (7.2, 10.4, and 14.4). Figure 3.7 reports the obtained results in terms of (a) utility and (b) rejections. Note that the x-axis indicates the traffic loads used in training; i.e., $p = \{0.5, 0.7, 0.9\}$ and $\lambda = \{7.2, 10.4, 14.4\}$, whereas the actually tested traffic loads correspond to $p = \{0.5, 0.7, 0.9\}$ and $\lambda = 16.2$.

GROWS is able to properly adapt to the new unseen traffic load, maintaining utility close to previous results and clearly outperforming the other strategies. Most importantly, in the event of a surge in traffic load, GROWS rejection rates



(a) Normalized utility. (b) Rejections per episode.

Figure 3.7: UA performance for unseen scenarios with higher traffic load, taking $\lambda = 16.2$. Note that the average traffic load values (i.e., the x-axis) correspond to the traffic loads used during training, whereas the actually tested traffic loads are higher. GROWS outperforms the other strategies in utility, accepting almost every user.

are still negligible, whereas they significantly increase for both *argmax* and *Q-learning*. We conclude that GROWS' found policy is able to avoid bottlenecks, even when confronted to unexpectedly high traffic loads, which certainly represents a desirable feature for any RA/UA strategy. Moreover, this experiment brings to light the good generalization properties of the GNN model, which learns a stable policy and extends it over unseen states.

This initial exploratory formulation successfully produces effective user association policies for simple experiments, showing that combining RL with GNNs is a promising approach for this problem. However, this first formulation relies on several simplifying assumptions, notably that only with the number of connected users and their mean rate as base station descriptors it is sufficient to infer a belief state and Q-function for the system. The underlying MDP/POMDP structure is not explicitly defined, and even though the RL algorithm used is model-free, there might be better partial state representations that are still simple and robust, yet better describe our system's state and therefore facilitate the learning stage.

As we move forward on our User Association problem, we develop a fully decentralized implementation that explicitly incorporates rejection as a valid action. We adapt the reward definition and the observation gathering process to align with effective decision instants, and derive the corresponding POMDP and graph representations to account for these modifications.

This page has been intentionally left blank.

Chapter 4

A fully decentralized user association scheme

In this chapter, we address the user association problem from a fully distributed perspective. Our focus is on a setting in which association decisions are taken locally at user arrival instants, without centralized coordination and with limited information exchange. This perspective is motivated by scalability considerations and by scenarios in which global state aggregation is either impractical or undesirable.

To enable decentralized execution, we restrict decision-making to arrival events and explicitly incorporate user rejection as an available action. These choices lead naturally to a formulation of the problem as a partially observable Markov decision process (POMDP) evolving over discrete decision instants. While this abstraction simplifies the system dynamics, it preserves the sequential structure of the problem and supports local decision-making based on partial observations.

The resulting framework allows user association policies to be implemented in a fully distributed manner, making it particularly suitable for large-scale and dynamic network scenarios.

4.1 A partially observable Markov Decision Process for Slot Fairness

We still consider the problem of user association in a system with N base stations, as presented in the previous section. Time is slotted and indexed by $t \in \{1, \dots, T\}$. At each time slot t, a user arrives with probability p.

Let $u \in \mathbb{N}$ denote the index corresponding to the order of user arrivals, and let t_u^i and t_u^f represent the arrival and departure times of user u, respectively. Each user u is characterized by a random discrete traffic demand $l_u \in \mathbb{N}$ and by a signal-to-interference-plus-noise ratio (SINR) with each base station b. This SINR determines an achievable normalized rate per physical resource block, denoted by $\bar{r}_b^u$, for $b = 1, \dots, N$. Each base station is equipped with a finite number of physical

resource blocks, denoted by $RB \in \mathbb{N}$, which must be allocated among its associated users.

The first novelty of this formulation is the explicit introduction of user rejection. When a candidate user u arrives, the system selects an action $a_u \in \{0, 1, \ldots, N\}$. If $a_u = b \in \{1, \ldots, N\}$, the user is associated with base station b; if $a_u = 0$, the user is rejected. Rejection may occur either due to a lack of available resources or as a deliberate decision of the control policy.

A second key aspect of the formulation is that decisions are taken exclusively at user arrival times t_u^i, which we refer to as *decision instants*. This choice is motivated by the operational structure of the problem: user association decisions are only required when a new user enters the system, at which point the user collects observations from the base stations and selects an action accordingly. As no decision is taken between arrivals, actions, rewards, and state transitions are defined only at these decision instants. In particular, the observed state and collected reward correspond to $s_t = s(t_u^i)$, and experience tuples (s, a, r, s') are stored only when an effective decision is made. This formulation naturally reduces the number of decision steps and improves learning efficiency, while preserving the sequential decision-making nature of the problem.

There are some key aspects to consider in order to be able to switch the MDP transition times. The most important is the discount factor: a future reward is discounted per every time slot elapsed from now. But the discount factor's effect, helping as it is for convergence, is not necessarily desirable when looking at our problem at hand: we wish for every user to be taken into account all the same, wether now or in the future. This would mean a $\gamma = 1$, which is actually possible to establish since we have finite episodes, but makes the simple algorithms we use very hard to converge. A very interesting article [118] helped us out of this conundrum: authors prove that a γ sufficiently high makes the optimal policy of the discounted expected reward maximization equivalent to the one solving the averaged reward maximization. This ties our problem together: we can still use a $\gamma < 1$ to help our algorithms' convergence but consider the average reward as our problem's target.

We build upon our system **state** s, still presented as the combination of the user's and base stations' features. We keep the user descriptors, SINR with each base station and load, but we modify the base stations' state to account for the users' dynamics. As now the elapsed time between observations is not fixed, there could be high variations of the number of connected users. Yet our past state definition gave no information regarding next departures. Although not explicit in our past formulation, user departure has a big incidence on resource distribution. When our POMDP observation time was every time slot, one could consider that the probability of having more than one user departing in a single time slot was, at least, small. The deep learning approximator had to learn the belief of having a next state in which no user left, a single user left, but the event of having more than one user leaving was unlikely.

This motivates the inclusion of descriptors that capture the remaining load or connection time of currently connected users. Accordingly, the state of each

base station is summarized by the number of associated users and by time-related statistics of the remaining sojourn times of the connected users, namely their mean $\widehat{\epsilon_b}$, variance var(ϵ_b), and minimum min(ϵ_b). We then have the base station's state $s_b \in \mathbb{R}^4$:

$$s_b(t_u^i) = [n_b(t_u^i), \widehat{\epsilon_b}, \min \epsilon_b, var(\epsilon_b)] \tag{4.1}$$

with $n_b(t_u^i)$ the number of users connected to base station b on decision instance u. The complete state is then the combination of the incoming users' state and the base stations' state as described, indexed on decision instant for arriving user u:

$$s_u = s(t_u^i) = [l_u, \overline{r_1^u}, ..., \overline{r_b^u} ..., \overline{r_N^u}, \|_{b=1}^N s_b(t_u^i)]$$

As discussed previously, the **transition** to the next state s_{u+1} is driven by the stochastic arrival process and by the action taken at the current decision instant. In particular, the transition captures the updated number of users connected to each base station (+1 if the user is accepted) as well as the evolution of the remaining sojourn times of all connected users under the current resource allocation. One could argue that only the remaining sojourn times are not enough, since we do not know the rates of connected users, but this information will be obtained through the reward collection, as we show next.

Let us then switch over to the reward in order to impose **Slot Fairness** [67], where the objective is precisely to ensure fairness at each time-slot. Recall that our problem statement was to maximize a utility function of the association decisions:

$$\max_{\mathcal{B} \in \{0,...,N\}^U} f(\mathcal{B})$$

The objective of the system is the same as in the previous chapter: rate maximization under a slot-fairness criterion. We then consider a per–time-slot performance function f, which depends on the rates achieved in a single time slot t, that is, on the set $\{r_b^u(t)\}$ for all users u active at time t. Typically, f is defined as a sum over all active users at that time slot, namely those users u such that $t_u^i \leq t \leq t_u^f$. The overall goal is to maximize the long-term average of this per–time-slot performance function; that is to say:

$$\max_{\mathcal{B} \in \{0,...,N\}^U} \frac{1}{T} \sum_{t=1}^{T} f((r_b^u(t))_{u:t_u^i \leq t \leq t_u^e}) = \max_{\mathcal{B} \in \{0,...,N\}^U} \frac{1}{T} \sum_{t=1}^{T} \sum_{u:t_u^i \leq t \leq t_u^e} f(r_b^u(t)) \tag{4.2}$$

where we have abused the notation by using f for the per-slot performance function and its form as a sum over all active users. The idea is that $f(x)$ in the right-hand of Eq. (4.2) should be concave in order to represent the diminishing returns principle, although other ways of justifying this form are possible [72]. This utility function could be any desired fairness function the ones defined by the α-fair family of functions. We will further develop on these functions in the next sections, for now we consider $\log(1 + r_v(t))$ with $r_v(t)$ the rate for user v, and we base this utility and reward formulation on [119].

Observe that from [118], a discount factor γ close enough to 1 leads to finding the same policy for the expected discounted cumulative reward and the expected average reward:

$$\arg\max_{\pi} \mathbb{E}\Big[\frac{1}{T}\sum_{t=1}^{T} f((r_b^u(t))_{u:t_u^i \leq t \leq t_u^e})\Big] = \arg\max_{\pi} \mathbb{E}\Big[\sum_{t=1}^{T} \gamma^t \sum_{u:t_u^i \leq t \leq t_u^e} f(r_b^u(t))\Big] \quad (4.3)$$

We have so far presented a per-time-slot function f, but different than in our previous formulation, reward collection now occurs only on decision instants, and not on a per-time-slot basis. We therefore define the instantaneous **reward** of the system as the mean utility achieved for connected users between this decision instant and the following, after taking action a_u.

$$R_{u+1} = \frac{1}{t_{u+1}^i - t_u^i} \sum_{v \in \mathcal{U}/t_v^f > t_u} \sum_{t \in [t_u^i, t_{u+1}^i]} \log(1 + r_v(t))$$

Similar to chapter 3, we want to find the policy that maximizes the expected discounted cumulative reward, for which we use the Q-learning paradigm and the Double DQN algorithm described in 1.

This set of definitions is sufficient to apply the GROWS framework. As in the previous chapter, our objective is to learn a policy that maximizes the expected discounted cumulative reward, which can be achieved using Algorithm 1. Unlike the previous formulation, the modification of the action space entails a corresponding change in the graph representation, which we describe in the next section. The approximation to the Q-function will still be achieved through the use of GNN, but also a classic Fully Connected Neural Network (FCNN) will be developed for comparison purposes.

4.2 A fully distributed UA algorithm

In order to learn the parametrized Q-value function, classical deep reinforcement learning approaches typically rely on multilayer perceptrons (MLPs), which are known to be universal function approximators. Yet, this architectures usually require aggregating information from the entire system into a fixed-size vector representation, leading to centralized computation, and possibly involving costly exchanges and high computation requirements. Moreover, if the policy is approximated using a function with a vector input, such as a Fully-Connected Neural Network, certain limitations arise. This approach requires a fixed ordering of the nodes and becomes infeasible if the input vector's dimensionality changes, for example, when the number of nodes in the network varies or increases.

From the classic neural networks different flavors have arisen to leverage for these difficulties, among which the GCN introduced in 3.1.2 propose a graph-based

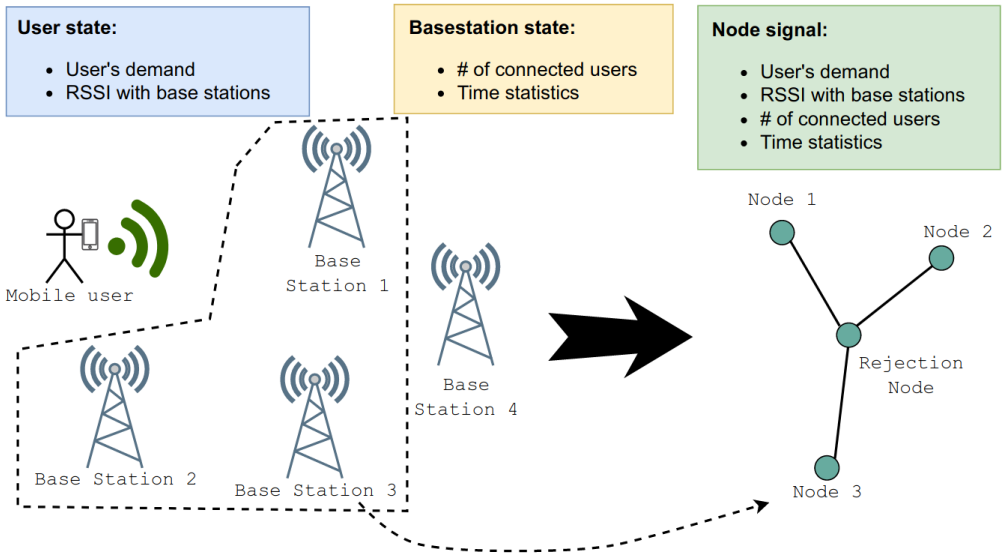


Figure 4.1: Graph representation of the system, where the star graph is constructed with the rejection node at the center. The rest of nodes represent base stations and the signal of each node is a composition of the base station and the user's state.

representation that supports decentralized computation through local information exchanges.

A key feature of the proposed user association and access control mechanism is its distributed nature, meaning that the algorithm is designed to operate without relying on a centralized decision-making entity for each incoming user. Instead, upon arrival, each user acquires local state information from the base stations, represented by the matrix $\mathbf{X}$, and independently selects an action, namely the base station to associate with or the rejection option.

The algorithm is executed by the incoming user. To build the graph representation, each incoming user constructs a graph where the base stations are nodes and the signal is the base stations' state. Instead of considering the whole network, the user only includes those k base stations it receives with the highest quality. Finally, and to allow the policy rejecting users, we also include an extra node representing this decision. In this case, the graph is constructed as a star, with this "reject node" at the center, see Figure 4.1.

This star graph has both drawbacks and virtues: although the center node could be a bottleneck for learning, the eigenvectors of a star graph act like a "map" that clearly marks the hub (central node) versus the leaves (other nodes), letting the GNN distinguish nodes and better capture global structure that simple neighbor-averaging would miss. In order to avoid these potential bottlenecks at the rejection node, self-loops are added to the shift operator, a common approach in graph convolutional networks.

This graph representation enables a fully distributed algorithm without requiring any communication between base stations. Instead, the incoming user independently gathers all necessary information. Notably, the time and energy

required for the user to compute the optimal policy are minimal, as this process involves only collecting the states of the base stations and performing a forward pass through a pre-trained GNN. As an example of how the proposed approach could be deployed in a 5G system, existing information exchanges between base stations and users, such as those described in 2.3, can be leveraged to construct the graph representation. These same exchanges could also be used in a straightforward manner by a fully connected neural network (FCNN) baseline, which would take as input the concatenation of the node-level signals.

Furthermore, by leveraging a GNN to define the policy, the algorithm becomes permutation equivariant: what is learned in a context can be transferred to unseen situations with similar characteristics. This ensures that decisions remain consistent regardless of how the nodes are reordered. The advantages of this property will be highlighted and discussed in the following section.

4.3 Experiments

We conduct a series of experiments, focusing on two specific settings that highlight the strengths of our proposal. First, we aim to demonstrate a key aspect of our algorithm: its ability to adapt to unseen yet similar scenarios, leveraging the permutation equivariance of the GCN. Second, we evaluate the algorithm's performance in a real-world 5G deployment in the city of Paris. All examples and their selected parameters and hyper parameters can be found at https://gitlab.fing.edu.uy/mrandall/growth.

To evaluate our proposal, we compare it against three different approaches: (a) our previously considered **baseline**, selecting the base station with the strongest SINR; (b) a **random** policy; and (c) the FCNN version of our algorithm. We continue referring to our proposal as **GROWS**.

4.3.1 Permutation equivariance

We will now develop on the permutation equivariance property, a key advantage of using GCN. Firstly, note that a GNN may be executed in a graph with any given number of nodes (since it is characterized by its filter taps $\mathbf{H}_k$). Secondly, in the construction of the Graph Shift Operator (GSO) $\mathbf{S}$ we have also arbitrarily chosen an order for the nodes.

However, unlike a vector-based representation of the problem, a GNN is independent of the node ordering. To understand this, consider a permutation matrix $\mathbf{P}$, which is a binary matrix satisfying $\mathbf{P1} = \mathbf{1}$ and $\mathbf{P}^\top\mathbf{1} = \mathbf{1}$. Since this permutation matrix is orthonormal, reordering the nodes and then computing the filter's output, represented as $(\mathbf{P}^\top\mathbf{SP})(\mathbf{P}^\top\mathbf{X})$, yields the same result as first computing the output $(\mathbf{SX})$ and then reordering it, represented as $\mathbf{P}^\top(\mathbf{SX})$. This permutation-equivariant property is naturally inherited by a GNN, as it performs point-wise operations on the output of the filter [120].

To illustrate the usefulness of this property in our context, we first consider a simple setting of three base stations in a straight line, as shown in Figure 4.2.

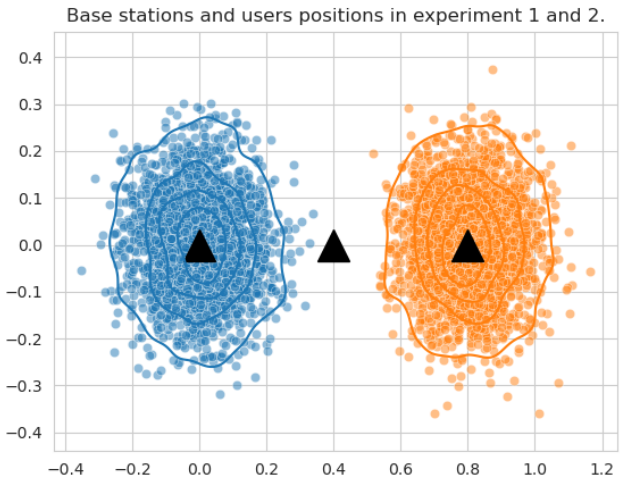
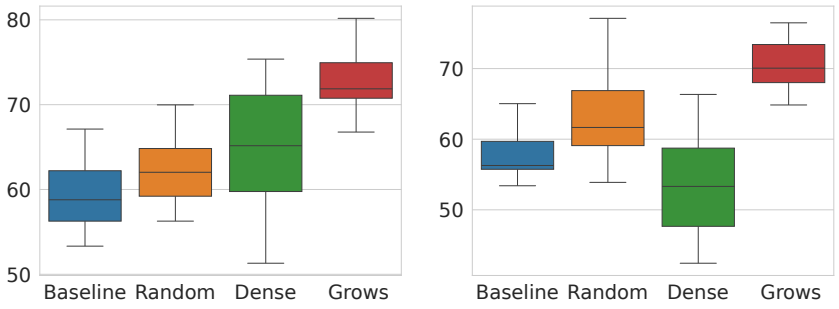


Figure 4.2: During training, users arrival is centered on the leftmost base station, whereas during test, users' arrival has shifted towards the base station to the right.

During training, users arrive according to a normal distribution centered around the leftmost base station. We subsequently test two scenarios, referred to as Experiments 1 and 2: one based on the initial setup and another where users cluster around the rightmost base station.

Similar to past experiments, the incoming user's characteristics (arrival, load and normalized rate with each base station) are stochastic. Each time slot a user arrives with probability $p = 0.5$, and the user's normalized rate with each base station is estimated according to the Friis equation as explained in earlier chapters. As explained, users arrive following a normal distribution centered on one of the outer base stations. This generates a scenario on which, on average, incoming users see the leftmost base station providing high rates, the center base station offering a medium rate, and the farthest base station most likely only supporting QPSK (the lowest possible rate). To generate the load requests we define three possible demand values: $\{80, 251, 503\}$, and randomly select the value according to the probability weights $\{0.6, 0.2, 0.2\}$. This leads to an average demand request of 99.4 Mbps, while the base stations have 10 physical resource blocks to distribute. The GROWS implementation relies on the `pytorch geometric` libraries, and hyperparameter search is done through the use of random search over sweeps. The GCN architecture is very simple, as in this setting there is only 1-hop, so we enhance the expressiveness of the node embeddings by using a single layer of 32 features, while the FCNN demanded two 64 sized layers. Numerical results from these experiments are presented in Table 4.1, in the columns marked as **Exp 1** and **Exp 2**, and results for utility are reported in Figures 4.3(a) and 4.3(b).

As expected, the FCNN version performs well in scenarios similar to those it was trained on; however, it struggles to adapt even to minor changes. In contrast, the permutation equivariance property of the GNN allows GROWS to adjust ef-



(a) Utility achieved, 1st scenario (Exp 1). (b) Utility achieved, 2nd scenario (Exp 2).

Figure 4.3: Comparison of both scenarios in the permutation experiment. When tested in scenarios similar to those encountered during training, both versions of the DDQN perform well (left figure). However, when the arrival patterns change and users come close to the opposite base station, as GROWS maintains its performance, the fully connected version of our algorithm struggles to adapt and performs poorly (right figure). Unsurprisingly, both random and baseline achieve the same results for both scenarios.

Table 4.1: Summary of the results achieved for the two scenarios describing the permutation experiment, and for the Paris experiment.

	Mean Utility			**Mean # of Rejections**		
	Exp 1	**Exp 2**	**Paris**	**Exp 1**	**Exp 2**	**Paris**
Baseline	59.8	58.5	47.1	76	72	97.5
Random	61.9	63	44.3	61	**56.2**	90
Dense	64.9	53.4	40.9	**59**	60.6	107.4
GROWS	**73.2**	**70.5**	**51.4**	62	57.5	**84.5**

fectively, yielding results in experiment 2 comparable to those achieved with the training realized in the first experiment. Naturally, both the random and the baseline policy achieve similar results in both scenarios, but systematically inferior to GROWS.

4.3.2 Performance on a 5G deployment derived from real base-station layouts

Next, we consider an evaluation scenario characterized by high user density and traffic load, which can significantly degrade system performance, particularly during large gatherings. To illustrate this, we focus on Paris, the host city for the 2024 Olympic Games, as a case study for managing user association in crowded environments. We utilize the 5G deployment data from a selected operator to determine the positions of the base stations, as shown in Figure 4.4, and we distribute users following a gaussian distribution around the center of the selected area to simulate

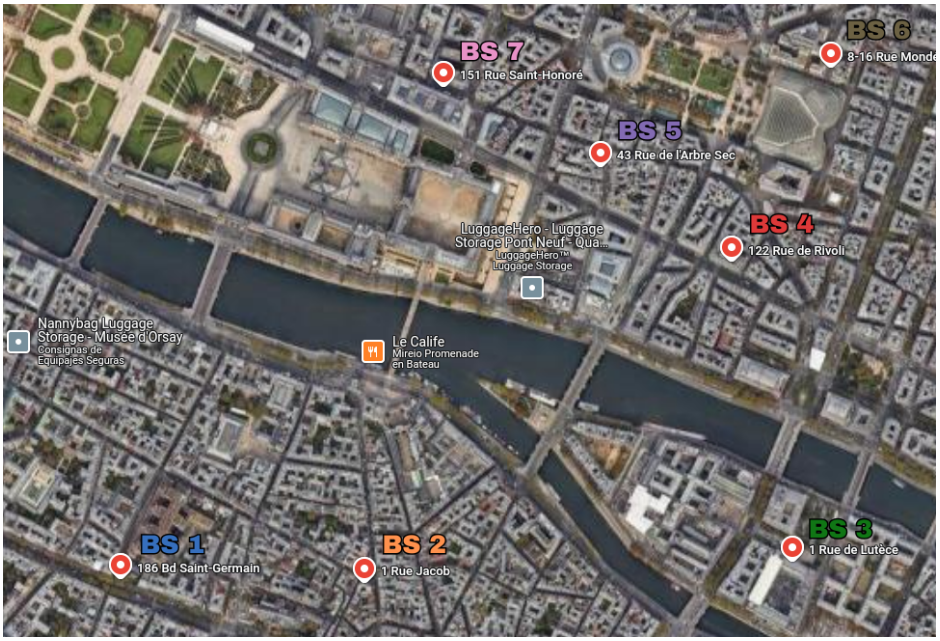


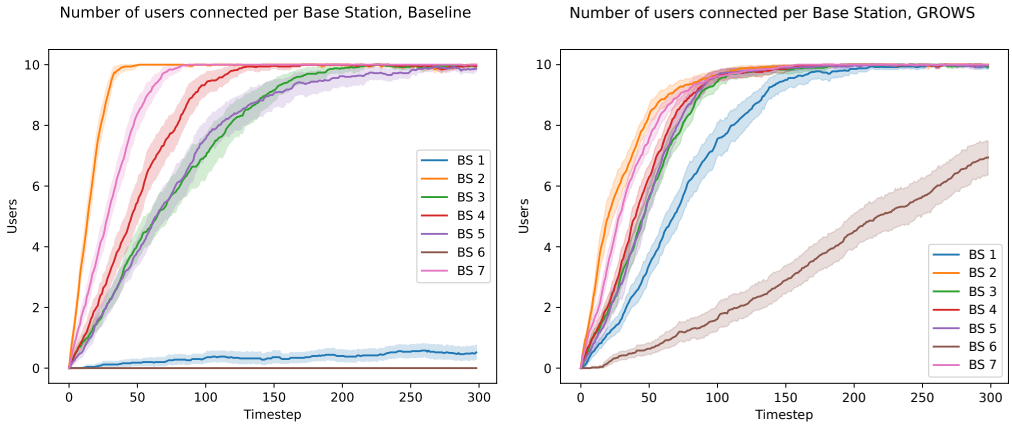
Figure 4.4: We select a densely populated area where large crowds gather, and select the 5G base stations deployed by a specific mobile operator. We randomize mobile users with higher density around the center of the figure, and random arrival times and demands.

a crowd gathering.[1] The seven different base stations are individually identified by an ID and a color, matching the results depicted in Figures 4.7.

The synthetic generation of users is very similar to the permutation experiment, although now arrivals follow a normal distribution centered on the center of the map shown in Fig. 4.4. Another substantial difference is that the map is bigger: an incoming user does not see possible links with all of the base stations, although we ensure any incoming user will see at least $k = 3$ base stations. As introduced at the beginning of this section, we desire a saturated scenario, for which we generate a higher resource demand than is available. Users arrive each time step with probability $p = 0.8$, with rates estimated according to their position and the Friis equation, and possible loads $\{50, 100, 200\}$ Mb. We assign to each base station up to 10 physical resource blocks. In order for our algorithm to scale, in this experiment we consider only the $k = 3$ base stations with best SINR regarding the incoming user to build our state. For this larger experiment, the GCN was composed of two layers of 256 hidden features each. The results of this analysis are summarized in Table 4.1, in the columns marked as **Paris**.

As can be seen in Fig. 4.5, the baseline approach leads to the fast saturation of the base station capacities, starting with the most centrally located nodes and filling them sequentially. In contrast, GROWS is able to effectively distribute users among available candidates.

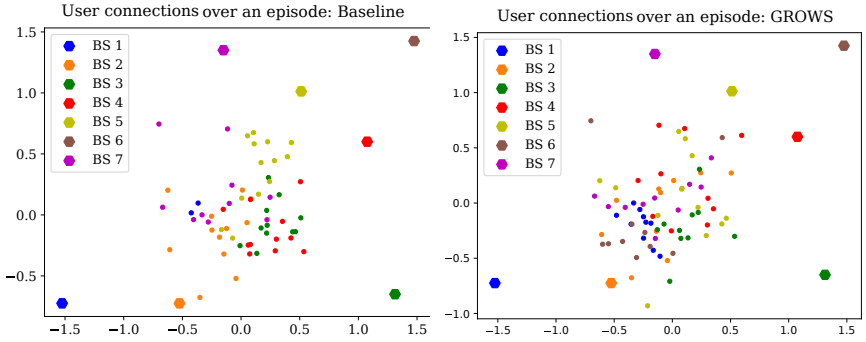
[1]Data was sourced from nperf.com, which provided information on the operator's 5G deployment, last visited on December 2024.



(a) Baseline connection evolution.

(b) GROWS connection evolution.

Figure 4.5: As expected, the baseline prioritizes base stations with stronger SINR (closer to the center), filling them sequentially. In extreme cases, while the center base stations are completely occupied, the resources of the farthest base stations remain entirely unused. Instead, GROWS has all base stations actively serving users, reflecting a more evenly distributed approach that optimizes resource utilization across the entire network.



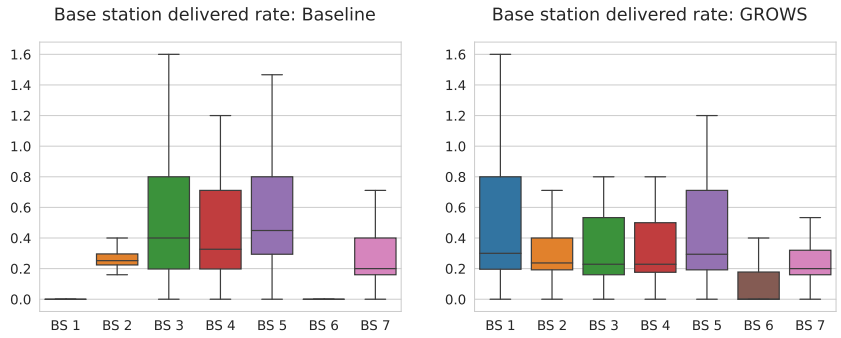
(a) Users' position and connections, baseline.

(b) Users' position and connections, GROWS.

Figure 4.6: Arrival positions of users and connection to each base station under the Baseline and GROWS. The baseline's policy is close to a "closest neighbor" policy, although not exactly due to random fading and base station saturation. GROWS policy is able to redirect incoming users to farther base stations with available resources when the closest ones are already serving a number of connections.

The users' connection to each base station during an episode are presented for the baseline and GROWS on Figure 4.5. Our proposal is able to use the farthest base stations when the closer ones are saturated, meaning our problem formulation is able to effectively improve fair rate utility, whereas the baseline performs poorly in this overcrowded scenario.

As we analyze the delivered rates per base station in Figures 4.7, the advantages of the GROWS policy become more evident. Unlike the baseline approach, which tends to saturate the nearest base stations (i.e. those positioned centrally), the GROWS policy ensures that all base stations are actively utilized. This broader



(a) Delivered rate per base station, baseline. (b) Delivered rate per base station, GROWS.

Figure 4.7: For the Paris experiment, we compare the rates delivered per base station. In this case, the baseline's saturation effect makes it so that mainly BS 3-5 are used, achieving higher rates for those but leaving unused resources in the rest of BS. In contrast, GROWS is able to distribute resource through the whole system. Notice that for the closest BS (3-5), GROWS delivered rates are slightly lower than for the baseline, but it still manages to achieve higher rewards through distribution of users to other base stations achieving therefore a better resource utilization.

engagement not only enhances overall system performance but also mitigates the risk of congestion in high-demand areas. By distributing user connections more evenly across the network, the GROWS policy maximizes rate efficiency and better accommodates varying user densities.

In Figure 4.7 we can observe that over-utilization of a few base stations by the baseline also leads to a great variance regarding delivered rates, while GROWS promotes a more equitable achieved rate per user.

Similar to the experiments in the previous chapter, the GROWS policy not only achieves higher utility, but also results in fewer user rejections compared to all alternative approaches; cf. Table 4.1. It is important to note that while our algorithm can explicitly take the 'rejection' action, the baseline and random policies do not have this capability, which could skew the comparison of rejection rates in their favor. This discrepancy is particularly evident when users receive strong signals from all base stations; in such cases, a random policy tends to distribute connections uniformly across the available stations. While this uniformity may lead to a lower rejection rate, it often results in sub-optimal utility due to poor resource allocation.

While this chapter focused on a distributed formulation of user association with a slot based fairness utility, its notion of fairness is inherently myopic. In the following chapter, we shift the focus to a more careful definition of the utility function and the associated reward structure, with the goal of capturing fairness over longer time horizons. To this end, we introduce a Markov Decision Process (MDP) formulation that embeds fairness directly into the reward collection mechanism, and we derive a partially observable MDP (POMDP) that integrates naturally with the GROWS framework.

This page has been intentionally left blank.

Chapter 5

A comprehensive fairness formulation

In previous chapters, we have adopted the reinforcement learning framework to derive user association policies, with the aim of maximizing achieved rates under a fairness criterion. Up to this point, fairness has been modeled through a per-slot utility function drawn from the α-fair family, with proportional fairness used as a representative implementation. Although this choices are widely used in literature, the actual effect of embedding such fairness definitions into learning-based algorithms under different system dynamics has not been thoroughly studied. In the next sections, we consider a careful inclusion of fairness utility functions into our dynamic user association system.

5.1 Choosing a performance function for fairness inclusion: the *average a-posteriori*

Let us begin by recalling our problem statement, in which we maximize a utility function of the association decisions:

$$\max_{\mathcal{B}\in\{0,\dots,N\}^U} f(\mathcal{B})$$

Naturally, the obtained performance will depend heavily on the choice of function f. The tradeoff between system-wide performance and fairness among users is controlled by this choice. However, and quite interestingly, the case of dynamic users that may enter and leave the system has received little attention (or none at all) when compared to the case where users are always the same and are present throughout the complete observation window (i.e. $l_u = \infty$ and $t_u^i = 0\ \forall u$).

To illustrate the necessity of considering this scenario separately, we will now consider two popular forms of functions f which were designed for the dynamic case where utilities may change (but still designed for static and infinitely long users), and show simple examples where they fail under the dynamic users case. We will end this section by presenting an effective performance function and the rationale behind it.

The first case was introduced in the last chapter and is the **Slot Fairness** [67], where the goal is to ensure fairness at time-slot level. The function f will take as argument the rates of a single time-slot t; and we maximize the average of a per time-slot performance function; that is to say:

$$\max_{\mathcal{B}\in\{0,\dots,N\}^U} \frac{1}{T}\sum_{t=1}^{T}\sum_{v:t_u^i\leq t\leq t_v^e} f(r_b^v(t)) = \tag{5.1}$$

$$\max_{\mathcal{B}\in\{0,\dots,N\}^U} \frac{1}{t_{u+1}^i - t_u^i}\sum_{v\in\mathcal{U}/t_v^f>t_u}\sum_{t\in[t_u^i,t_{u+1}^i]} \log(1+r_v(t)) \tag{5.2}$$

Consider now a very simple example where users are very scarce and there is typically at most a single user active in the time window $[0,T]$. Assume that $f(x)=\log(x)$ in (5.1) for clarity, although what follows is valid for any other concave function. Furthermore, we consider that the SINR is constant throughout the users' transmission duration, so that $r_b^u(t)=r_b^u$ is a constant once b is chosen. We thus have that the incoming user will connect to the base station using the following maximization:

$$\begin{aligned}\max_{b=\{0,\dots,B\}} \frac{1}{T}\sum_{t=t_u^i}^{t_e^u}\log(r_b^u(t)) &= \max_{b=\{0,\dots,B\}}\frac{1}{T}(t_u^e - t_u^i)\log(r_b^u)\\ &= \max_{b=\{0,\dots,B\}}\frac{1}{T}\frac{l_u}{r_b^u}\log(r_b^u),\end{aligned}$$

where in the last equality we have substituted the user's stay time $\Delta t_u = t_u^e - t_u^i$ with its load l_u divided by its rate r_b^u. Since $\log(r_b^u)/r_b^u$ is not a monotonously increasing function (in fact it is decreasing for any $r_b^u \geq e$), we may end up choosing the base station providing the least rate r_b^u to the user, a clearly undesirable situation.

Let us then consider the alternative proposed in [67], the so-called **Horizon Fairness**. Instead of considering each time-slot separately, here the mean performance throughout the observation window of each user is the argument of a per-user function f, and the overall performance of the system is the sum over all users. That is to say:

$$\max_{\mathcal{B}\in\{0,\dots,N\}^U}\frac{1}{U}\sum_u f\left(\frac{1}{T}\sum_{t=1}^{T} r_b^u(t)\right). \tag{5.3}$$

However, the same example as before shows that this performance function is unsatisfactory too. In this case, the incoming user will connect to the base base

station using the following criteria:

$$\max_{b=\{0,\dots,B\}} \log\left(\frac{1}{T}\sum_{t=t_u^i}^{t_u^e} r_b^u(t)\right) = \max_{b=\{0,\dots,B\}} \log\left(\frac{1}{T}\Delta t_u r_b^u\right)$$
$$= \max_{b=\{0,\dots,B\}} \log\left(\frac{1}{T}\frac{l_u}{r_b^u} r_b^u\right) = \max_{b=\{0,\dots,B\}} \log\left(\frac{1}{T} l_u\right),$$

meaning in this case that any base station may be chosen and the overall performance will be considered the same, another undesirable situation.

The problem with the objective function above, and the reason why all base stations were equally likely to be chosen, is that we are considering the average over the complete observation window, even when the user was inactive. We should instead consider averages over the lifespan of user u, in particular throughout its complete session (i.e. once users have finished their transmission, or in other words *a posteriori*).

We propose thus the following performance function, which we denote as **Average a-posteriori Fairness**:

$$\max_{\mathcal{B}\in\{0,\dots,N\}^U} \frac{1}{U} \sum_{u:t_u^e\leq T} f\left(\frac{1}{t_u^e - t_u^i}\sum_{t=t_u^i}^{t_u^e} r_b^u(t)\right). \tag{5.4}$$

In this case our ongoing example of a single user during the observation window results in the following optimization criteria:

$$\max_{b=\{0,\dots,B\}} \log\left(\frac{1}{\Delta t_u}\sum_{t=t_i^u}^{t_e^u} r_b^u(t)\right) = \max_{b=\{0,\dots,B\}} \log\left(\frac{1}{\Delta t_u}(t_e^u - t_i^u) r_b^u\right)$$
$$= \max_{b=\{0,\dots,B\}} \log\left(r_b^u\right),$$

which is clearly the desired solution.

This leads to a reward collection on decision instants which is the performance function applied to the achieved rate for users v that have finished since the last time we observed the system:

$$R = \sum_{v:t_{u-1}^i < t_v^f < t_u^i} f\left(\frac{1}{\Delta t_v}\sum_{t=t_v^i}^{t_v^e} r_b^v(t)\right) \qquad = \sum_{v:t_{u-1}^i < t_v^f < t_u^i} f\left(\frac{1}{\Delta t_v} l_v\right). \tag{5.5}$$

Remark 3 (On stability) There is another important reason that justifies considering only finished users to measure performance. Fairness measures assume a number of fixed users, and as such neglect the possibility of unstable systems where the users' arrival process is such that they accumulate in the system. For instance, in a processor sharing queue with a total rate of r, if the the number of new users per second is more than the inverse of the expected value of l_u/r (the

average number of users that finish their transmissions per second) it is easy to see that the number of active users will grow indefinitely. By considering only connections that have started and finished in the observation window, the proposed performance function will avoid this situation by choosing to reject certain users (i.e. choosing $b = 0$).

Remark 4 (On load fairness) It is important to note that the average a posteriori fairness criteria does not take into account l_u to weigh users. If for instance we have a single resource to allocate to two users so that one has to be rejected, we may prefer to assign it to the one that will use it the most (i.e. the one with the largest l_u). However, this resource-centric approach has been shown once and again to be extremely unfair. Furthermore, if we consider each flow to be a user, either choice will satisfy exactly one user. Our proposed performance measure chooses the user that obtains the best average rate. Last but not least, choosing users with a large l_u may produce congestion (since users will remain longer in the system) and force the decision algorithm to reject users. Again, including only finished users in (5.4) should implicitly take this aspect into account.

On the utility function f

We have presented examples and will present results based on the choice of $f(r) = \log(r)$ or variations ($f(r) = \log(1+r)$ for when $r \leq 1$), which promote proportional fairness. Our main target on this chapter is set on the MDP formulation: when to collect the reward to ensure the desired fairness is being applied to our system. This is independent of the selection of a particular utility function, and can be applied to any function of the α-fair family, or others. For example, if a policy prioritizing resource utilization is desired a smaller α is to be considered, and α has to be larger if looking for a policy that prioritizes user acceptance [86].

We know now what our optimization goal is: fair rate maximization. We will next introduce a complete MDP formulation consistent with the proposed *average a-posteriori* formulation for the user association problem.

5.2 A Markov Decision Process formulation for fairness inclusion

In order to implement the proposed *average a-posteriori* scheme for fairness inclusion on dynamic systems, a first consideration arises: to compute the average rate for a user we need to keep track of several variables, the user's arriving time being just one example. As we do not desire our state to grow unnecessarily, we next introduce a complete description of the MDP underlying our system's dynamics, which we have so far omitted as we prioritized the algorithmic construction. We now present it, from which we will derive a POMDP that we can solve by using the already introduced GROWS framework.

Markov Decision Process 101

A decision process is one in which there is a decision maker taking a sequence of actions over an environment, which in turn returns some reward to the system.
We say that a process is Markov if it is a stochastic process that follows the Markov property.[1]This property, also referred as "memory-less", indicates that the probabilities of moving to the next state depend only on the present state:

$$P(s_{t+1}|s_t, s_{t-1}, s_{t-2}, ...) = P(s_{t+1}|s_t)$$

If actions are involved, then:

$$P(s_{t+1}|s_t, s_{t-1}, s_{t-2}, ..., a_t, a_{t-1}, a_{t-2}, ...) = P(s_{t+1}|s_t, a_t)$$

An MDP is defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$, in which $\mathcal{S}$ represents the state space, $\mathcal{A}$ represents the action space, P indicates the transitions probabilities, R is the reward and $\gamma \in [0, 1)$ is the discount factor.
An MDP is said to be deterministic if the transitions are deterministic, and stochastic if p is a probability, $p(s', r|s, a)$ being the probability of moving to state s' from state s given action a is taken and receiving reward r.
The use of MDP is very direct to describe sequential decision processes, and there are several methods to find the policy $\pi(a|s) = P(a|s)$ that maximizes the expected reward, as the reinforcement learning framework introduced earlier. This maximization is usually wanted on the discounted expected return: $G_t = \mathbb{E}\Big[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}\Big]$, as in chapter 3; or as in last chapter for the average expected reward: $G_t = \lim_{t\to\infty} \frac{1}{t}\mathbb{E}\Big[\sum_{k=0}^{t-1} r_{k+1}\Big]$.

Let us then state our problem under this framework. This means finding the definition of **s, r, a, P,** γ so that the system's evolution follows a Markovian Decision Process.

Firstly, and as has been the norm in this thesis, time is discrete as we take decisions (or **actions** in the MDP jargon) as users arrive; i.e. we will consider the system at each t_u^i for $u = 1, \ldots, U$.

An action is simply the chosen base station $a_{t_u^i} = b \in \mathcal{B}$, and once we have

[1]Andrey Markov (June 14, 1856 – July 20, 1922), Russian mathematician. Among other significant contributions, his study of stochastic processes (1906) gave rise to the origin of Markov chains (a name coined in 1926). A professed atheist and activist, he was called the "militant academic" by the press for rejecting honors from the Tsar and for requesting his own excommunication from the Orthodox Church (to which he did not belong), following the excommunication of Gorky, who was his friend [121]. In 1913, when the government celebrated the 300th anniversary of the Romanov dynasty, Markov organized a counter-celebration using the 200th anniversary of Bernoulli's Law of Large Numbers [122].

taken a decision at time t_u^i we can interpret that the system responds by randomly picking when the next user will arrive (i.e. a value for δ_{u+1} so that $t_{u+1}^i = t_u^i + \delta_{u+1}$), their load ($l_{u+1}$) and the rate at which the user will be able to connect to each base station. Note that the latter has to be normalized in order to make it independent of the current and future state. We consider the bit-rate per physical resource block, already introduced as $\overline{r_b^{u+1}}$.

As discussed earlier, at each decision time we will collect as **reward** the performance function obtained from users v that have finished since the last time we observed the system:

$$R_{t_u^i} = R = \sum_{v: t_{u-1}^i < t_v^f < t_u^i} f\left(\frac{1}{\Delta t_v} \sum_{t=t_v^i}^{t_v^e} r_b^v(t)\right) = \sum_{v: t_{u-1}^i < t_v^f < t_u^i} f\left(\frac{1}{\Delta t_v} l_v\right). \tag{5.6}$$

Let us then define the **states** and, given a certain action, the **transition probabilities** between all possible state pairs. When the transition to the next state depends only on the current state and selected action, the MDP is deterministic, otherwise it is stochastic, which is our case: our next state will very much depend on current state and action, but also on the (stochastic) next arrival. Note that these transition probabilities do not have to be known or explicitly formulated to solve problem (5.4) (for instance, through Reinforcement Learning as we presented in past chapters), but they must be well-defined. That is to say, given a certain state at time t_u^i, the decision b and a certain value of both δ_{u+1} and the normalized rates $\overline{r_b^u}$, we have to be able to compute the state at time t_{u+1}^i. This way, the transition probabilities are given by the distributions of δ_{u+1} and $\overline{r_b^{u+1}}$.

The requirement above is solved by keeping track of how much time remains for each user v to finish its transmission with the current resource distribution (which we will denote as $\epsilon_v(t)$). Note that under the classical assumption of exponential distributions for both the loads and the inter-arrivals times, and invoking the so-called memory-less property of this distribution, we need to keep track of how many users are present in each base station only (and the mean duration). However, in the general case, we may need all the values of $\epsilon_v(t)$, or at least several more statistics than just the size and its mean. For clarity, let us discuss a few examples of how $\epsilon_v(t)$ is modified from time-slot t_u^i to the following t_{u+1}^i given a certain value of δ_{u+1}.

Example 1 (Base station not chosen) Consider a certain base station b with a set of values $\epsilon_v(t_u^i)$ (with v indexing the users in that particular base station), and assume the choice at t_u^i was not b. If $\epsilon_v(t_u^i) > \delta_{u+1}$ (all users had a longer remaining time than the time to the next arrival), then all these variables are simply updated to $\epsilon_v(t_u^i) - \delta_{u+1}$. If $\epsilon_v(t_u) < \delta_{u+1}$ for at least one user v', then v' will have left the system. Note that at time $t_{v'}^e$, when this user leaves, we have to down-scale all of the remaining $\epsilon_v(t_{v'}^e)$ since there are more resources for the remaining users. It may well happen that because of the increased resources a second user will also leave, even if its remaining time was longer than the time to the next arrival at the beginning of the previous slot (i.e. $\epsilon_v(t_u) > \delta_{u+1}$).

Example 2 (Chosen base station) Assume now that action $b > 0$ was chosen at t_u^i and let us discuss how the state corresponding to b changes at t_{u+1}^i. For starters, given $\overline{r_b^u}$ and the users already present in b, we may compute $r_b^u(t_u^i)$ and, in conjunction with l_u, also $\epsilon_u(t_u^i)$, which is added to the set of values that characterize the state of this base station. Note that, similarly to the case when a user leaves, the rest of the users in this base station will have their respective $\epsilon_v(t_u^i)$ up-scaled due to having less resources per users. After this step, we proceed as in the previous example to check whether a user leaves or not, and update the values of $\epsilon_v(t)$ accordingly.

As our proposed reward definition considers the average rate achieved per user $r_v = l_v/\tau_v(t_v^e))$, in order to compute the collected reward we also need to keep track of how much time each user has been on the base station, which we will denote as $\tau_v(t)$, and the initial load l_v. Updating this value between time-steps is similar to the case of $\epsilon_v(t)$, but without the scaling due to users entering or leaving the system. At the same time, the normalized rate $\overline{r_b^v}$ is no longer needed, as its effect is already included in the remaining time $\epsilon_v^t = \frac{l_v^t}{r_b^v} = \frac{l_v^t \times RB}{\overline{r_b^v} \times N_b}$, where N_b is the number of active users and RB is the number of physical resource blocks.

Remark 5 (When the reward considers the instantaneous rate) Going back to the time slot fairness we have considered in the previous chapters, observe that in that such case storing the arrival times is unnecessary. By considering the remaining time and normalized rate, if our performance function considers only the delivered rate for the interval $[t_u^i, t_{u+1}^i]$, the arrival time provides no useful information. In this case our state is reduced to:

$$s_b = \{(\overline{r_b^v}, \epsilon_v(t_u^i))\},$$

for all users v currently connected to base station b. We considered this reward formulation in the past chapter. Our last state representation included statistics summarizing ϵ_v, with the reward collection providing information related to the connected users' rates.

All in all, the system may be characterized by a set of triplets for each base station, including for each active user v the values corresponding to $(l_v, \tau_v(t_u^i), \epsilon_v(t_u^i))$, and the arriving user rates and load. Let us then summarize our state:

$$s(t_{u+1}^i) = [l_{u+1}, \overline{r_1^{u+1}}, \ldots, \overline{r_N^{u+1}}, \|_{b=1}^N s_b(t_{u+1}^i)]$$
$$s_b = \{(\overline{r_b^v}, \epsilon_v, \tau_v)\}$$

That is to say, for each use v connected to base station b, the initial load, how long it has been on the system, and their remaining sojourn time if the resource assignation remains the same. The user assignment algorithm observes this, the new user's load l_{u+1} and $\overline{r_b^{u+1}}$ (for all $b = 1, \ldots, B$) and has to decide to which base station (if any) to connect the new user. These transitions are illustrated in Fig. 5.1. We will refer to these values (the set of triplets, the new user's load and

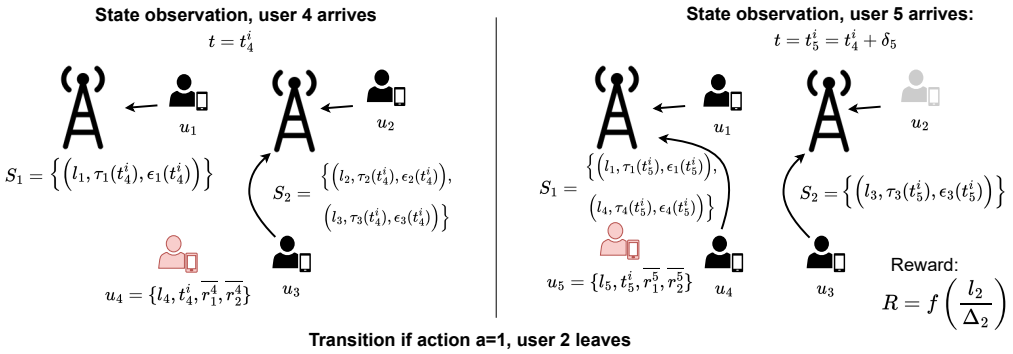


Figure 5.1: To the left, a schematic of the user assignment problem and the associated MDP representation. After observing the system's state at $t_u = t_4^i$, consisting of $(l_v, \tau_v(t_u^i), \epsilon_v(t_u^i))$ for all active users, and the incoming user's characteristics: arrival time t_4^i, the normalized rates $\overline{r_b^u}$ for $b = 1, \ldots, B$ and the load l_u. We then have to decide to which base station (if any) connect this new user. The system then draws a new value for δ_{u+1}, $\overline{r_b^{u+1}}$ for $b = 1, \ldots, B$ and l_{u+1}. To the right, one possible evolution when action $a = 1$ is selected. As user 5 arrives, base station 1 has accepted user 4, and user 2 has left the system. The system updates the state for each base station accordingly: $(l_v, \tau_v(t_{u+1}^i), \epsilon_v(t_{u+1}^i))$. The transition due to action a_4 yields the reward associated to the departure of user 2 $R_4 = r(s_4, a_4 = 1) = f\left(\frac{l_2}{\Delta_2}\right)$.

normalized rate to each base station) as the system state, and denote it with s or s_u when it refers to the system state when user u arrives to the system.

The MDP introduced fully describes our system's dynamics, yet the state is composed of triplets $(l_v, \tau_v(t_u^i), \epsilon_v(t_u^i))$ for each connected user, plus the variables depicting the incoming user (load, normalized rate with each base station, arrival time). As discussed in past chapters, this fast growing state space quickly leads to untractable solutions, a classic problem in learning commonly addressed as the *curse of dimensionality*. In the next section, we introduce a Partially Observable Markov Decision Process (POMDP) over which we can employ our proposed reinforcement learning algorithms.

5.3 Integration with our Reinforcement Learning framework

Let us now explain how the POMDP formulation integrates with the proposed user-association algorithm. As discussed, and at the risk of sounding repetitive, we will use Reinforcement Learning which acts and observes the system's behavior in order to learn an optimal policy. As a flash reminder, we are looking for a policy $\pi(s)$ which is a probability distribution over the possible actions $b = \{0, \ldots, B\}$ given a state s. Recall that for decision instant t_u^i (corresponding to the arrival of user u) we have that the reward is

$$R_{t_u^i} = R_u = \sum_{v: t_{u-1}^i < t_v^f < t_u^i} f\left(\frac{1}{\Delta t_v} l_v\right). \tag{5.7}$$

To obtain a deterministic optimization problem (i.e., non sample-path-dependent) we will take an infinitely long observation period and the expectation of the resulting average performance function:

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \lim_{U\to\infty} \frac{1}{U} \mathbb{E}_\pi \left[\sum_{u=0}^{U} R_u\right] \tag{5.8}$$

$$= \underset{\pi}{\operatorname{argmax}} \lim_{U\to\infty} \frac{1}{U} \mathbb{E}_\pi \left[\sum_{v: t_v^f < t_U^i} f\left(\frac{l_v}{\Delta t_v}\right)\right]. \tag{5.9}$$

Reinforcement Learning algorithms typically maximize the expected discounted cumulative reward:

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}_\pi \left[\sum_{u=0}^{\infty} \gamma^u R_u\right]. \tag{5.10}$$

As mentioned, a discount factor γ sufficiently close to 1 arbitrarily approximates the solution of (5.8) [118], so we will consider both problems as equivalent.

This formulation enables our problem to be solved using Algorithm 1, updated to account for the new POMDP definitions of the state, action, and reward. Combined with the graph representation presented in Section 4.2, this approach allows us to derive user association policies for the *average a posteriori* formulation.

So in order to find our desired allocation policy, we again learn an approximation to the action-value function $\hat{Q}(s, b; \theta)$, where the optimal parameter θ^* is obtained via stochastic gradient descent so that $\hat{Q}(s, b; \theta^*) \approx Q^*(s, b)$.

Note that the number of connected users changes over time. However, the GCN requires fixed size input signals, independently of the number of users. In particular, we aggregate user information through statistics, in order to produce this vector. This also enables the proposed algorithm to scale even when facing more realistic scenarios in which we consider several base stations with dozens of physical resource blocks and hundreds of users. Motivated by classical queuing-theoretic considerations, for the metric $\epsilon_b(t_u^i)$ we consider the average, the variance and the minimum of the remaining sojourn time of users connected to base station b considering current allocation. Similarly, for l_b (remaining load) and $\tau_b(t_u^i)$ (sojourn time so far) we retain their mean and variance. These seven quantities, together with the total number of users, the incoming user's normalized rate $\overline{r_b^u}$ and load l_u, form the 10-dimensional feature vector x_b representing each node (i.e. base station). The set of B such vectors is $\hat{s} \in \mathbb{R}^{B\times 10}$, resulting in the following action-value approximation:

$$Q^*(s,b) \approx \hat{Q}(\hat{s}, b; \theta^*) \tag{5.11}$$

$$\hat{s} = [x_1, ..., x_b, ..., x_N] \tag{5.12}$$

$$x_b = [\overline{r_b^u}, l_u, var(l_b), \widehat{l_b}, var(\tau_b), \widehat{\tau_b}, var(\epsilon_b), \widehat{\epsilon_b}, \min(\epsilon_b)] \tag{5.13}$$

This compact representation substantially reduces dimensionality while retaining the essential information required for fair and stable decision making as we show in the next section.

5.4 Experiments for the average a-posteriori formulation

5.4.1 Experimental Setup

We evaluate this proposed fairness-aware user-association framework through simulations encompassing multiple network conditions and traffic models. The objective is to assess the impact of embedding fairness into the decision process on system-level efficiency, stability, and user-perceived performance.

We report several metrics, including the total achieved utility, number of rejections, number of finished jobs, per user rates and sojourn time statistics. Our code is available at gitlab.fing.edu.uy/mrandall/grows-fairness.

As previously, we will refer to our proposal as GROWS. In this case, since the approximator is not crucial and both the GNN and FCNN versions obtain similar results, only the GNN version will be compared.

To contextualize performance, we compare GROWS against four representative baselines:

- **Argmax policy:** The standard baseline we have used for comparison so far.
- **Random policy:** Users select base stations uniformly at random.
- **Processor sharing policy (PS):** The load balancing policy for process sharing proposed in [52]. The proposal is applicable under Poisson arrivals, in scenarios with more available resources than required average load, with a fixed normalized rate per base station. When meeting these conditions and considering policies unaware of the queues size, this proposal achieves optimal results. The obtained policy returns a matrix, rows being the incoming user load class and columns the base stations rates, which are assumed constant and discrete.
- **Greedy shortest remaining time policy (greedy-SRT):** Each user connects to the base station that, given the current state, minimizes its expected completion time disregarding fairness [123].

For our simulations, we present rate results normalized to the maximum achievable rate for one Resource Block (RB), which we consider $r_b^u = 1$, and time values normalized by the duration of a time slot. Depending on the 5G configuration (modulation and coding scheme, bandwidth, numerology, etc.) different time slot durations and achievable rates are possible. For the next experiments, we consider the physical layer configuration used in past chapters. We apply the throughput formula from the 3GPP 5G NR standard [124] to estimate the resulting time slot duration and rates for each user according to their position, as in previous experiments.

The following subsections present results for three representative scenarios: light-load conditions with Poisson arrivals, heavily loaded systems with bursty arrivals, and a realistic deployment using empirical topology data.

5.4.2 Poisson arrivals and unloaded scenario

We first consider a lightly loaded system where user arrivals follow a Poisson process with moderate intensity. In this regime, the network remains far from saturation, allowing us to examine how each policy allocates resources when congestion is not the dominant factor. This is important since the baselines only reject when there is no more capacity left in the base stations. The main objective with this experiment is to assess that, by embedding fairness in the reward function, our approach does not compromise efficiency.

In particular we consider a simple setting with two base stations, each with 5 resource blocks. In this experiment, users arrive centered on one of the base stations, with which the rate is $\overline{r_1^u} = 1$, while for the other base station the rate is $\overline{r_2^u} = 0.32$. This fixed rate setting is necessary in order to implement the processor sharing proposal under its hypothesis. Users' load is also normalized to the peak rate and is selected randomly between two values ($l_1 = 6$ and $l_2 = 15.2$), and users have inter-arrival times following a Poisson distribution. The arrival rates are $\mu_1 = 0.15$ users per time slot and $\mu_2 = 0.1$ users per time slot. The average load requested for the system is $\bar{l} = 2.42$ and is well below the available resources $\sum_b r_b = 6.6$.

Simulation results can be seen in Table 5.1. For each performance indicator we show the mean as well as the lower and upper quartiles (q_{25} and q_{75}). As expected, our proposal obtains larger utilities. What is more interesting is the fact that in this context the greedy-SRT policy should be optimal in terms of sojourn time, as proven in [123] and confirmed by the simulation results. Note that the PS policy also offers very competitive performance in this respect, although since it does not observe the queues' state it performs slightly worse. On the other hand, and as further illustrated in Fig. 5.2, our proposal is able to learn a quasi-optimal policy, achieving a slightly worse performance for sojourn time than both the greedy-SRT and the PS baselines. However, our proposal is able to impose fairness in the sense that the achieved sojourn times are more concentrated around the mean. This is also true for the achieved rates, where our formulation is able to maximize the achieved mean rate all while obtaining a smaller variance around it.

Table 5.1: Results for the unloaded scenario. We present the mean and the lower and upper quartiles ($q_{0.25}, q_{0.75}$) for the achieved utility, sojourn time and rates. There are no rejections for all methods, and are thus not reported.

		Random	**PS**	**Argmax**	**greedy-SRT**	**GROWS**
Utility	$\tilde{U}$	0.265	0.346	0.347	0.348	**0.353**
	$[q_{25}, q_{75}]$	[0.25, 0.28]	[0.33, 0.36]	[0.33, 0.37]	[0.33, 0.37]	**[0.34, 0.38]**
Rate	$\tilde{r}$	2.50	4.29	4.29	4.29	**4.84**
	$[q_{25}, q_{75}]$	[0.89, 5]	[2.79, 5]	[2.88, 5]	[2.95, 5]	**[3.73, 5]**
Sojourn time	$\hat{T}$	3.75	2.75	2.94	**2.68**	3.04
	$[q_{25}, q_{75}]$	[1.4, 10]	[1.2, 4.08]	[1.2, 3.91]	[1.2, 3.60]	**[1.2, 3.48]**

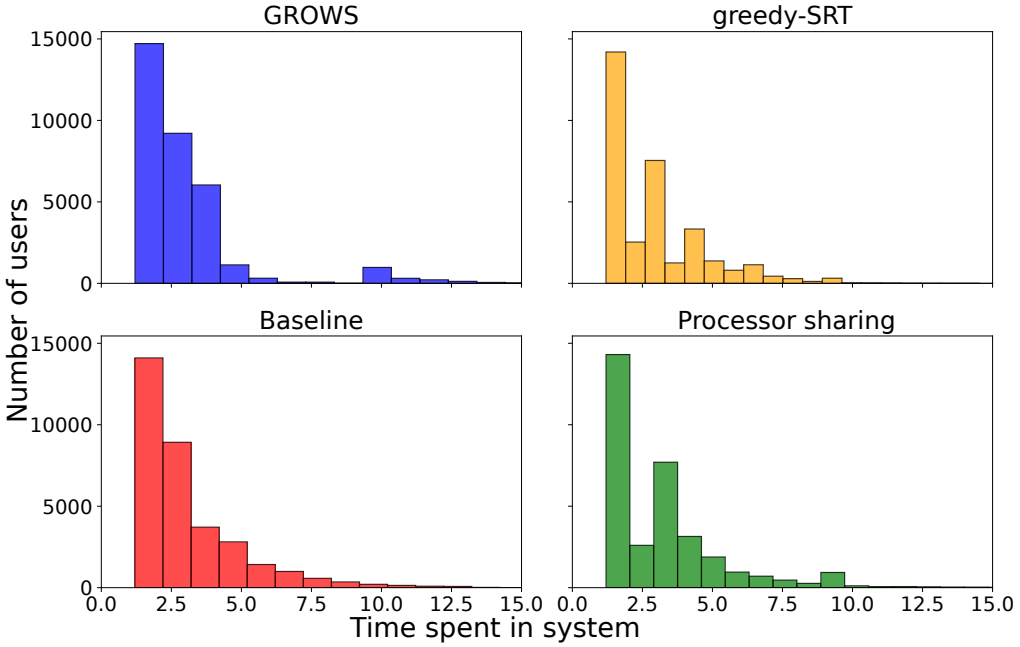


Figure 5.2: Histogram of the users' sojourn time in the unloaded scenario for all the evaluated methods. Our proposal provides a competitive mean with a smaller variance (i.e. more fair).

All in all, under light traffic, the proposed fairness-aware policy behavior is similar to efficiency-driven policies as the greedy-SRT, argmax and PS. However, as can be seen in Figure 5.2 the obtained rates and sojourn times are more concentrated, providing early evidence that fairness-embedded rewards do not bias the policy toward overly conservative decisions. This behavior establishes a solid reference for subsequent experiments under higher load, where the fairness–stability trade-off becomes more evident.

5.4.3 Heavily loaded scenario

We now analyze a heavily loaded regime in which the user-arrival rate approaches or exceeds the network's service capacity. This scenario highlights the policies' ability to balance efficiency, fairness, and stability when congestion and user re-

Table 5.2: Results for the heavily loaded scenario. All methods obtain a very similar percentage of rejected users, but our fairness-aware policy is selective onto which users should not be accepted, resulting in orders of magnitude improvements in rate and sojourn time.

		Random	**Argmax**	**greedy-SRT**	**GROWS**
Utility	$\tilde{U}$	0.212	0.191	0.199	**1.235**
	$[q_{25}, q_{75}]$	[0.21, 0.22]	[0.188, 0.193]	[0.196, 0.202]	**[1.13, 1.25]**
Rate	$\tilde{r}$	1.009	1.006	1.006	**43.52**
	$[q_{25}, q_{75}]$	[0.32, 1.01]	[0.32, 1.01]	[0.32, 1.01]	**[16, 50]**
Sojourn time	$\hat{T}$	232.6	233.3	233.3	**4.70**
	$[q_{25}, q_{75}]$	[99, 505]	[99, 507]	[99, 507]	**[2, 6.25]**
Rejected Users (%)	$\hat{RU}$	41.5	41.6	40.96	**39.2**
	$[q_{25}, q_{75}]$	[40.7, 42.1]	[40.8, 42.5]	[40.3, 42.0]	**[38.5, 39.7]**

jections become significant. In particular, in this experiment base station position is fixed with 50 resource blocks. User arrivals follow a Poisson distribution with loads $l_1 = 100, l_2 = 235, l_3 = 511$, arrival rates $\mu_1 = .25, \mu_2 = 0.2, \mu_3 = 0.04$, and possible normalized rates $r_1 = 1, r_2 = 0.32, r_3 = 0.16$. As we impose a saturated system, in this case the average load is $\bar{l} = 92.44$ and the available resources are $\sum_b r_b = 74$. Table 5.2 summarizes the results for this case. We omit the processor sharing proposal since since it is not possible to compute the corresponding policy in this overloaded scenario.

The first observation from Table 5.2 is that, by the end of the considered time window, all algorithms reject a similar number of users. However, as illustrated in Fig. 5.3, the rejection trend is clearly in favor of our method, and extending the episode duration would further accentuate the performance gap.

Furthermore, even if the total number of rejected users is comparable, GROWS learns a stable policy that selectively rejects those connections most detrimental to overall resource utilization. Consequently, it achieves a significantly larger utility, which in turn produces dramatically improved rates and sojourn times when compared to the baselines (40 times larger and 50 times smaller respectively).

These results show that, under heavy load, GROWS concentrates available resources on connections that maximize the fairness-aware utility, producing much higher per-connection rates and markedly reduced time on system for accepted users. This performance is also explained spatially. Figure 5.4 shows the number of connected users per base station for two of the heuristics baselines and GROWS. Note that under the argmax and greedy-SRT, base stations rapidly saturate (starting by the one providing the strongest SINR) and then all new arrivals are rejected until an existing transmission is completed. In contrast, GROWS converges to a policy that keeps the system in a steadier operating regime: fewer users are simultaneously active, accepted users receive higher rates, and departures occur more regularly. Interestingly, the best base-station (number 1) is maintained as a kind of "fast lane", serving only a few concurrent users, while the other two base stations are more crowded.

These results confirm that, in congested conditions, the fairness-aware learn-

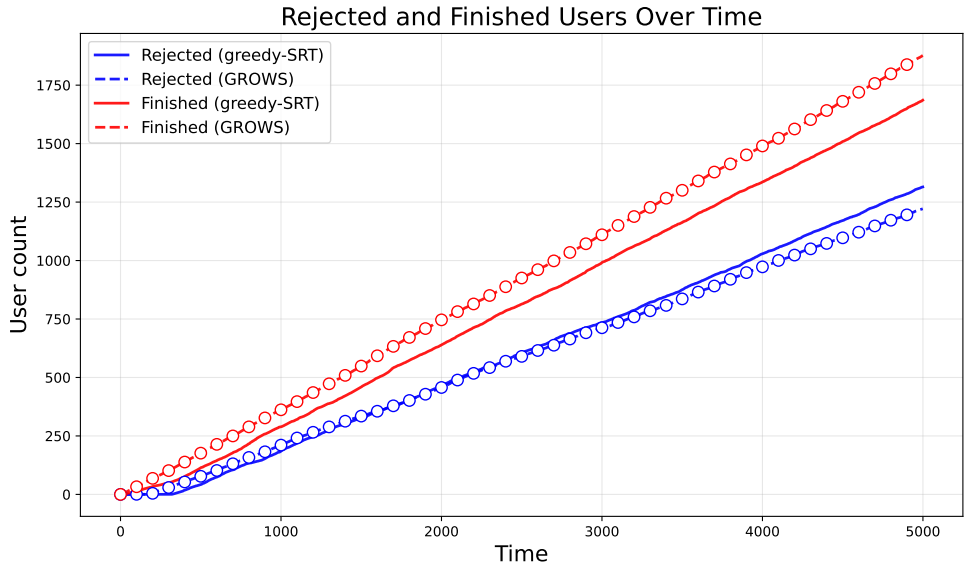
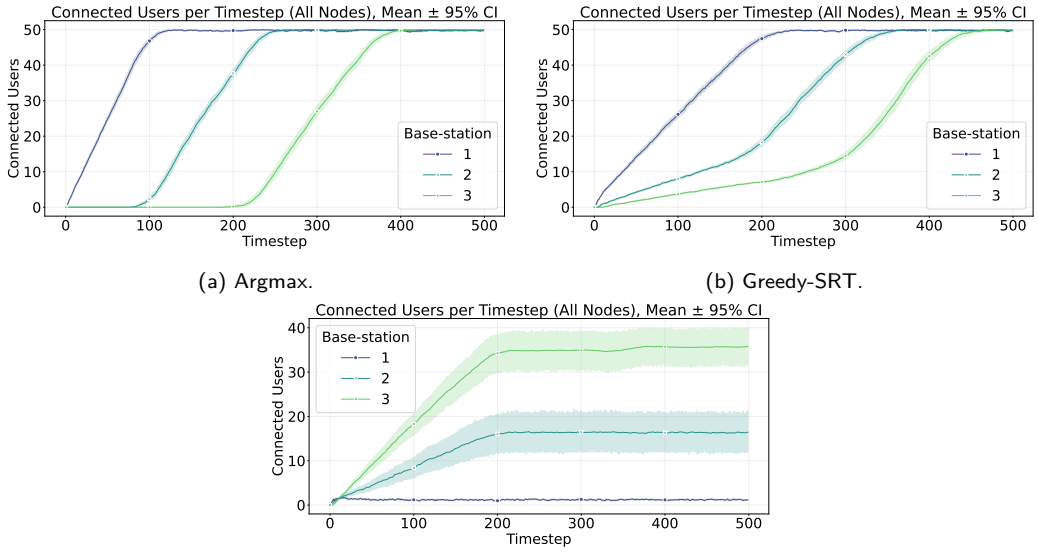


Figure 5.3: While our proposal starts rejecting some users from the beginning, the greedy-SRT first fills all three base stations and then starts rejecting incoming users. In this unstable scenario our policy is able to keep a stable policy, lowering rejections and fastening departures.



(a) Argmax.

(b) Greedy-SRT.

(c) GROWS.

Figure 5.4: Number of connected users to each base station in the heavily loaded scenario. Both the argmax and greedy-SRT policies fill the base stations, starting by the one with higher SINR. This ends up backfiring when the base stations are saturated and new users are accepted only after an active user finishes. On the contrary, our proposal keeps a "fast lane" policy: the base station with the strongest signal has very few users connected, and none of the base stations is filled. This stable policy allows for higher rates, lower time spent on the system and the same number of rejections (see Table 5.2).

Table 5.3: Results for the real deployment scenario. Performance is consistently better in our proposal, although by a smaller margin than in the overloaded scenario.

		Random	**Argmax**	**greedy-SRT**	**GROWS**
Utility	$\tilde{U}$	0.41	0.47	0.49	**0.52**
	$[q_{25}, q_{75}]$	[0.38, 0.45]	[0.42, 0.50]	[0.45, 0.54]	**[0.49, 0.55]**
Rate	$\tilde{r}$	1.6	1.97	2.10	**2.23**
	$[q_{25}, q_{75}]$	[0.9, **3.2**]	[0.9, **3.2**]	[1.1, **3.2**]	[**1.3**, **3.2**]
Sojourn time	$\hat{T}$	12.0	10.0	9.63	**8.63**
	$[q_{25}, q_{75}]$	[5.6, 22]	[**5.3**, 22]	[**5.3**, 18]	[**5.3**, **16**]
Rejected Users (%)	$\hat{RU}$	0	0	0	0
	$[q_{25}, q_{75}]$	-	-	-	**-**

ing framework enables policies that balance efficiency and stability by proactively managing load across base stations and maintaining high-quality service for active users.

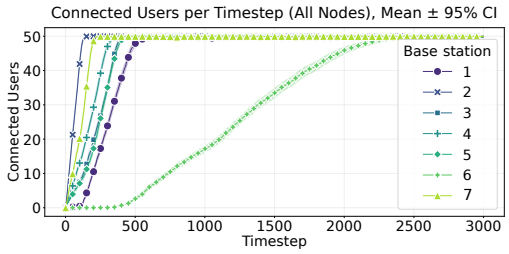
5.4.4 Performance on a real deployment based experiment: Paris

We finish the experimental evaluation by repeating the experiment over the Paris 5G deployment seen in last chapter to achieve a more realistic scenario. We mimic the base stations' positions, and center the users' arrivals on the "Île de la Cité", a very crowded area. We also change the arrival processes, using a Bernoulli distribution for the users' inter-arrival times, and using a normal distribution for their position (therefore for their normalized rates with each base station). The whole system consists of seven base stations, each with 20 resource blocks, and for our algorithm we only consider the three base stations with the highest $\overline{r_b^u}$. Users can carry two possible load demands ($l_1 = 17$, $l_2 = 43$) with arrival rates $\mu_1 = 0.5$, $\mu_2 = 0.1$. The scenario is more balanced in terms of load, having many base stations each with 50 resource blocks, although bad connection choices could lead to occasional saturation for a particular incoming user.

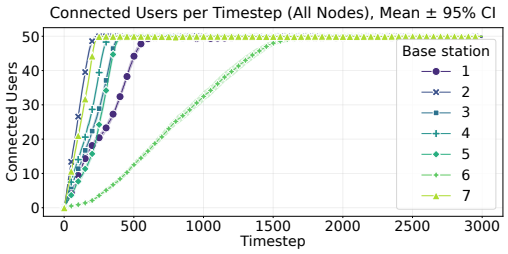
As can be seen in table 5.3, although performance gains are not as dramatic as in the overloaded scenario, results are still consistently better for GROWS (e.g. by over 10% in the case of the sojourn time). In figures 5.5 we compare the users' connection to the base stations, which are now 7 and not always with the same availability (a user in the center may see all base stations while an outlier user farther away may only see 2 or 3).

Perhaps the most noteworthy aspect of this scenario is that, as in the overloaded case, GROWS connects users to all base stations instead of filling them by proximity (as observed with the Argmax or the greedy-SRT). This behavior is clearly illustrated in Fig. 5.6 and Fig. 5.5, which compares the delivered rates per base station for GROWS and the greedy-SRT policies.

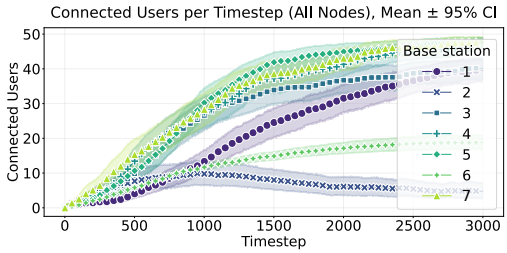
It is interesting to point that although our problem statement is user-centered (its the utility is focused on the users' perceived performance), it implicitly promotes balanced resource utilization across base stations. This emergent property



(a) Users' connections to base stations according to the baseline.



(b) Users' connections to base stations according to the greedy-SRT.



(c) Users' connections to base stations according to our proposal.

Figure 5.5: As shown in the previous experiment, both the baseline and *greedy*-SRT policies end up filling the base stations with higher SINR. In contrast, our proposal achieves a more stable scenario, using all available resources to avoid bottlenecks in saturated base stations.

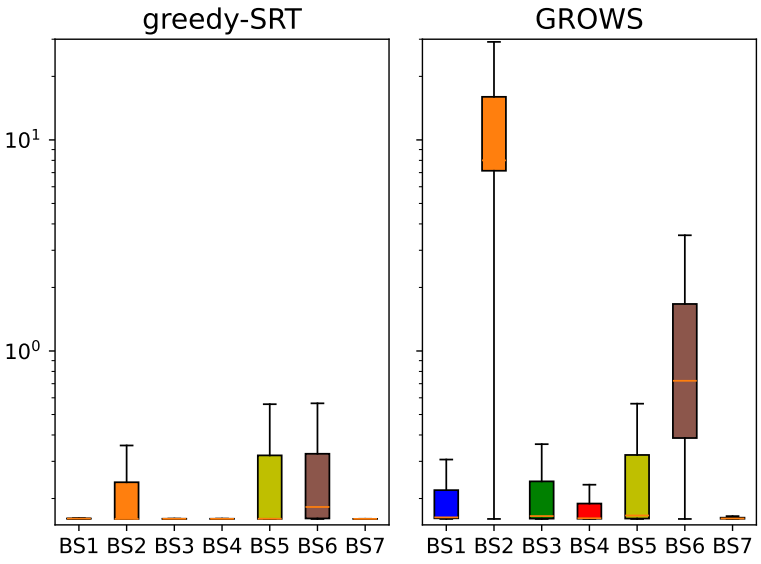


Figure 5.6: Mean rate per base-station for GROWS and the greedy-SRT (the baseline has a similar behavior and is thus omitted). Although our proposed framework is user-centric, the resource usage is also significantly improved.

reflects the advantage of a state-aware, fairness-driven policy: by pursuing equitable user outcomes, the system also attains higher overall efficiency.

5.5 Summary of our approaches to UA

We now share some considerations on our work over user association. As new communication technologies multiply the possible services and devices, an ever more complex and entangled system grows, leading to an increasing need for resource optimization. We address a very simple problem, as old as wireless communications themselves, but from which there is still much to gain: user association, or 'to which connectivity provider a user has to connect?'. We chose to tackle user association as it embodies the first resource allocation decision.

As many resource allocation problems, user association can be seen as discrete combinatorial optimization. Yet, as the possible connections grow in number (users, base stations), this problem becomes untractable. We profit from the recent wave of machine learning techniques to propose the combination of reinforcement learning and graph neural networks to find user association policies. In the following paragraphs, we summarize and compare our proposals and we discuss on our implementation choices.

Along this first part, we have introduced several formulations, starting from a simpler model-free approach to apply RL to a fair comprehensive proposal. For all three implementations, our system's state is constructed by combining the arriving user's characteristics and the current state of the system, given by the base stations descriptors. As we want our system to be completely defined by the node signals over a graph in which the base stations are the nodes, we combine the base stations' state and the incoming user's load and rate with each base station to form the node signal. Our first graph representation is built by establishing the base stations as nodes, and an edge will connect two base stations if they are able to communicate directly. For the 5G scenario, as we consider the backhaul to provide cost free communications, it will result in a fully connected graph (in other scenarios this could not be the case).

For our second approach, we propose a decentralized framework in which each base station shares its belief state, and the user estimates the Q-function values and selects the action. This implies only considering as decision instants those time slots in which a user arrives. As time elapses between two actions, we now keep track on the remaining sojourn time for connected users (or equivalently, their remaining load). As rejection is included, we now build the graph as a star-graph with a 'rejection node' as the center node and the available base stations as the other nodes.

Finally, we propose a fairness oriented formulation, that by considering only the reward from ending users, enables a more comprehensive definition of fairness. On this last framework, we still consider decision instants only for those time slots in which a user arrives, and the action space still consider rejection as a valid action. As we now consider for our reward the average rate achieved by the users, our state needs to keep track of the users' history (initial load, arrival time, normalized

rate) and their remaining load (or sojourn time). We use expressive statistics to aggregate this per-user metrics and achieved a fixed size state representation.

Each proposal has its advantages and drawbacks. Our first approach has many 'fake' decision instants, which makes difficult training and convergence for our RL algorithms, and our state representation, although intentionally simple, hides what may be useful information on the system's state, as the evolution of connected users.

Our second proposal considers only decision instants when an action can be taken, which in turn forces the incorporation on the state of connected users' descriptors. We summarize these users' information through statistics, so that our state has a fixed size, independent on the number of connected users. The possibility of rejection enhances the action space, and entrails a new graph representation. Yet, the reward is slot-time based, which may not be fair on the long run.

Finally, we focus our reward definition looking for a more comprehensive fair formulation. To do so, we only consider for the reward collection those users ending between decision instants. This forces to keep track of the users' past information, resulting in a larger state definition, which we summarize through statistics to a fixed number of features per base station (10 in this case).

More interestingly, our considerations on the reward collection process in order to ensure fairness are applicable to other (PO)MDP formulations, up to the whole RL/MDP framework. We are emphatic in pointing out that without careful thought over our maximization target, resource optimization can only go so far: our learning algorithms are only as fair as we design them.

Discussion on the Selected Algorithms and Deep Learning Models

The rapid growth of machine learning, driven by its astonishing results on several areas, has led to an intense focus on identifying the 'best' algorithms, often framed as a race towards marginal performance gains. In this context, it is important to clarify the scope and intent of our work and to justify our methodological choices accordingly.

We begin by distinguishing between algorithm design and problem formulation. This thesis primarily focuses on the latter: the construction of a POMDP formulation that supports the application of reinforcement learning to the user association problem. While a large body of literature is devoted to proposing novel learning algorithms, our objective is instead to define a state and action representation that is sufficiently expressive to capture the relevant system dynamics, while remaining simple enough to scale to realistic network sizes.

From this perspective, the problem could alternatively be cast within a multi-agent reinforcement learning framework or even under a game-theoretic paradigm, as explored in several related works. Similarly, one may question the choice of a Double DQN based algorithm, given its relative maturity, or the use of a classical graph convolutional network (GCN) instead of more recent attention-based architectures. These alternatives are indeed viable, but they are deliberately not pursued here.

The rationale behind these choices is tied to the intended contribution of this thesis. Our goal is not to demonstrate state-of-the-art performance under a carefully tuned, problem-specific learning architecture, but rather to advance the discussion on how learning frameworks interact with problem formulation. Specifically, we aim to show that a well-defined POMDP, together with a principled utility function and a graph-based representation, is sufficient to enable effective policy learning for UA, independently of the particular DRL or GNN variant employed.

In fact, once the (s, a, r, s') tuple and the graph structure are established, extending our framework to other DRL–GNN combinations can be done with minimal effort. However, achieving, for instance, a 20% reduction in rejection rate or a 50% increase in throughput through algorithmic fine-tuning would offer limited scientific insight in this context. Such improvements are often driven by extensive hyperparameter searches, longer training times, or highly specialized architectures, and do not necessarily generalize beyond the specific setting considered.

Instead, our results demonstrate that incorporating learning-based decentralized decision-making into dynamic resource optimization problems yields substantial benefits, even when using relatively simple and well-known models. The proposed POMDP formulation is generic enough to support a wide range of DRL algorithms, and the graph representation is sufficiently robust to accommodate different GNN architectures. The experimental results confirm that the resulting policies align with the intended optimization objectives, adapt to varying system conditions, and naturally extend to different fairness criteria. Moreover, the framework is general enough to be integrated into other resource allocation problems where fairness considerations are essential.

This page has been intentionally left blank.

Part II

Intra-cell resource allocation for different user requirements over different time scales

Chapter 6

Introduction and Problem Statement

We now turn our attention to the intra-cell resource allocation policy. In the first part of this thesis, we examined a proportionally shared spectrum policy among user equipments, where we assumed a single class of users with different loads and rates. One of the biggest innovations on 5G and beyond is the support of three different services with particular delay and bandwidth requirements, such as Massive Machine Type Communications (MMTC), enhanced Mobile Broad Band (eMBB) and Ultra-Reliable Low Latency Communications (URLLC). The base station's intra-cell resource allocation policy has to take into account their different needs, which is the subject of the next sections.

In order to achieve these multiple service requirements, all users have to share resources over the 5G Orthogonal Frequency-Division Multiple Access (OFDMA) frame. One of the strategies proposed by the 5G standard is puncturing, which allows the scheduler to assign eMBB services, and on a shorter timescale to overwrite part of these assignments when a URLLC user arrives. The optimization of puncturing poses a challenging problem: the optimal allocation depends on traffic arriving over different timescales, which forces the scheduler to make allocation decisions without knowledge of future users' demands, all while having to satisfy several strong constraints. This kind of multiple timescales optimization with restrictions is also to be found in many interesting problems. We propose a learning mechanism where the system learns offline the optimal allocation according to the network state. This learned estimation is then used online to determine the optimal allocation. Through simulations, we verify that the proposed learning strategy provides results close to the optimal policy, improving state of the art proposals for puncturing schemes.

6.1 Introduction

Next generation wireless networks introduce many interesting challenges, as new services and users multiply. In order to deal with these stringent and dynamic scenarios, and as expressed in the previous part, recent standards as 5G and future 6G enable and promote the use of machine learning in order to optimize resources

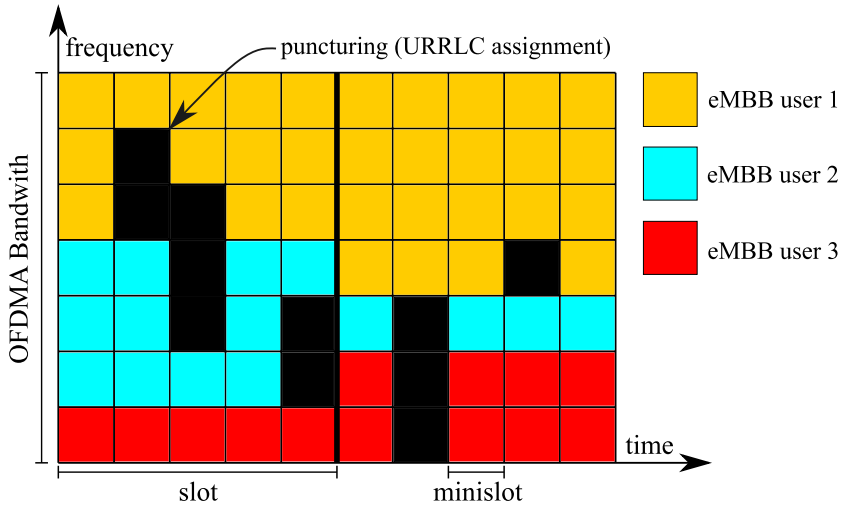


Figure 6.1: Puncturing scheme. The OFDMA grid is split in time (slots and minislots) and frequency. While resources to eMBB users are assigned at the slot timescale, URRLC users' resources are allocated at the minislot timescale, overwriting eMBB data and possibly resulting in errors for the corresponding eMBB receiver.

and services [125, 126]. From beam selection in massive Multiple Input Multiple Output schemes [127, 128] to Non-Orthogonal Multiple Access (NOMA) [129], from cyber security [130] to user centered QoE [131, 132], the machine learning revolution has set foot on open problems arising from new generation wireless communications.

We focus here in the coexistence and resource sharing of two different types of users in 5G and beyond: the classical high bandwidth user (eMBB) and the innovative reliable and low latency new users (URLLC). Each of these services has very different characteristics and requirements. While eMBB users demand high data rates and thus an important set of resources of the OFDMA (Orthogonal Frequency Division Modulation Access) grid, URLLC services require high reliability and very low latency [133]. As far as the scheduler is concerned, the URLLC service requires its queuing delay to be virtually null.

In order to satisfy the aforementioned requirements, the 5G New Radio (NR) standard proposes different mechanisms [134], among them the so-called puncturing scheduling. In this case both services are assigned resources on the OFDMA grid at different timescales: slots for eMBB and (shorter) minislots for URLLC (see Figure 6.1). The main idea behind puncturing is to assign with preemptive priority URLLC traffic over eMBB resources. In this case, the scheduler assigns resources to each eMBB user every time slot, but URLLC traffic are assigned on a minislot timescale.

The resulting resource allocation problem is inherently complex given the coupled nature of the two timescales involved (URLLC and eMBB scheduling), and the uncertainty of future URLLC demand: the scheduler needs to assign resources to eMBB users so as to maximize a chosen metric (e.g. throughput for eMBB users),

resources that may be later overwritten in order to satisfy URLLC requirements, unknown at the moment eMBB resource are scheduled.

Naturally, assigning an OFDMA resource during a minislot to a URLLC user on a resource already assigned to an eMBB user implies that the latter will receive, at least, part of the corresponding frame with errors. Let us review more carefully the steps our double scheduler has to take.

First, the scheduler would need to allocate OFDMA resources regarding some optimal criterion to eMBB users at the beginning of each time slot. An essential variable to optimize this allocation is the effective rate each eMBB user would obtain according to their wireless channel's state (i.e. SINR). Nevertheless, that effective rate depends on the number of minislots that would be eventually overwritten by URLLC users. This dependence means that such information will only be available in the future and not at the beginning of the time slot when the scheduler assigns the resource blocks to eMBB users. Additionally, it has to decide which eMBB users to overwrite when assigning URLLC traffic within some optimal criterion per minislot, and this typically depends on the resources assigned to, and the SNR of, each eMBB user.

Thus the scheduler has to solve two optimization problems at different timescales but highly coupled. Adding further difficulty, the scheduling decisions have to be made in a very narrow time lapse.

6.2 Related works

This resource allocation problem arising from puncturing in 5G NR has received significant research attention in recent years [135]. In particular, the performance of specific assignment and scheduling policies has been extensively analyzed under a variety of traffic and deployment scenarios [136–139].

Several works explicitly formulate the puncturing optimization problem and propose heuristic solutions that perform well in selected scenarios [140,141]. Other authors propose a Q-learning or deep-learning approach to derive adaptive puncturing policies [141, 142]. The use of reconfigurable intelligent surfaces (RIS) has been proposed to improve the coexistence between eMBB and URLLC services [143]. These works help us to visualize how puncturing behaves in certain scenarios, and provide insight into the problem modeling.

We particularly remark the work of [119], where the puncturing optimization problem is rigorously formalized and several practical relevant cases are analyzed. An interesting observation in [119] is that when the problem's functions (loss, constraints) are non-convex (e.g. threshold loss), the proposed methods increasingly deviate from the optimal solution as the URLLC traffic load grows. This behavior is especially relevant in light of the fact that, as discussed for instance in [144], the bit error rate (BER) in OFDM systems exhibits a threshold-like dependence on noise and interference. As puncturing increases, such behavior is expected, highlighting the relevance of finding policies that are well-adjusted to threshold losses. We will further develop on the subject over the chapter.

Highly coupled optimization problems with multiple time scales and hard constraints are not exclusive to wireless communications. Similar structures arise in a variety of application domains. For instance, a two timescales problem arising from long term storage management is studied in [145]. Similar problems are studied in the more general field of resource allocation [146], as in [147] where a two timescales coupled optimization problem emerges from management of IoT devices, or [148] where authors use supervised learning in order to solve wireless IoT networks' resource allocation problems. The authors of [149] study a two timescales optimization problem arising from micro grid scheduling, and [150] analyzes the optimization problem given by the need to make predictions over energy consumption for electrical vehicles charging on two timescales.

6.2.1 Contributions

In order to decide now the best resource allocation without knowing future puncturing needs, we propose a supervised learning framework that learns to approximate both scheduling policies. The key observation is that, although the optimal solution for our coupled optimization problem is unavailable at the scheduling intervals, it can be found afterwards, once events have happened and we can consider all necessary data from a certain time slot (and its corresponding minislots). If the loss due to puncturing penalizes the eMBB user's utility through a convex loss function, the problem turns out to be a convex optimization problem, which has led to numerous and well known solutions [119]. But if the loss penalty is not convex (for example, a threshold penalty, a much more suitable model for the actual wireless communication link [144]), as we mentioned before proposed solutions can grow afar from optimal. In such cases, the optimal policy would have to be found by trying all possible scheduling scenarios, making it impossible to find online.

Following the ideas of the 'learning to optimize' paradigm (as in [151] and [152]), we propose a statistical learning method that enables the exploitation of signal's correlations in order to approximate optimal solutions. The proposed learning framework separates the resource allocation problem over each timescale, although incorporating significant statistical information. This is achieved by training two learning machines (agents) that will be decision makers for each timescale resource allocation. As we show in the simulations section, this method has shown to be well suited for learning when faced with non-convex loss functions, allowing the proposed learning formulation to surpass current state of the art policies for the eMBB and URLLC coexistence dilemma. Crucially, since actual resources are assigned in the inference phase of the learning algorithms (i.e. costly training is peformed offline) the computational cost of the proposed method's operation is extremely lightweight.

All in all, the main contribution of this work is the general formulation we propose to deal with a two timescales coupled optimization problem, following the learning to optimize paradigm. We also provide a concrete application to the 5G NR problem of scheduling eMBB and URLLC users through puncturing. Our adaptive learning framework allows us to find a close to optimal policy given any

Table 6.1: Comparative analysis of State-of-the-Art proposals.

Proposal	**Minislot scheduling**	**Puncturing lossess**			
		Linear	Quadratic	Threshold	Other
Round Robin heuristic [137–139]	✗	✓	✗	✗	✗
House Allocation Policy heuristic [140]	✓	✓	✗	✗	✗
Mixed integer linear programming-deep reinforcement learning [141]	✓	✓	✗	✗	✗
Q-learning [142]	✗	✓	✗	✗	✗
Alternating optimization based Heuristic [143]a	✗	✓	✓	✓	✗
Heuristic [143]b	✗	✓	✓	✓	✗
Gradient scheduler based optimization [119]	✓	✓	✓[2]	✓[2]	✗
Ours: Learning to Optimize paradigm	✓	✓	✓	✓	✓

[2] For certain scenarios and specific URLLC traffic distributions they approximate the optimal policies.

loss function. This distinguishes our practical approach from similar state-of-the-art proposals, enabling more realistic scenarios. As succinctly presented in Table 6.1, when compared to other methods available in the literature, our proposal is able to closely approximate optimal policies all while being computationally amenable to an online operation.

In Section 6.3 we define a mathematical formulation for the puncturing problem and detail the proposed learning framework. This general problem, which as we mentioned before appears in several other domains, is instantiated in the URLLC and eMBB puncturing case in chapter 7, presenting it as a use case example and proposing different system reward functions, formulating the offline optimization problems with which we get the optimal solutions. As we present in chapter 8 our proposal yields state of the art results for convex loss functions, while allowing for a much closer approximation to the optimal solution than most widely used algorithms in the non-convex case. Furthermore, we compare the results of using several supervised learning algorithms, illustrating that the mathematical formulation of the learning problem is robust and generalizes well with different techniques. This allows for a combination of total or partial handcrafted algorithms (which are widely used in practical applications) with state of the art learning techniques (which are widely used in scientific research).

6.3 A two timescales optimization problem

Although this thesis focuses on AI for wireless systems, many optimization or decision making problems of different fields share the same structure or face similar difficulties, leading to adapting and migrating solutions across disciplines. In order to facilitate this healthy exchange, we now define the coupled allocation problem in a more general form, and defer to the next chapter the 5G use case.

Let us consider the following optimization problem. A finite and fixed amount of a certain resource has to be distributed over time to different users, who will in turn profit from the allocation by obtaining an utility. Consider two groups of users: E primary users, and U secondary users that will be puncturing primary users. In a more general formulation, instead of users there could be simply two sets of resource consumers. The second group of resource recipients has strong requirements with regard to the resource utility to be satisfied (a certain demand to fulfill), which implies the utilization of resources already assigned to the first group. A decision maker has to allocate resources to both groups, satisfying the second group's demands all the while trying not to harm the first group's utility.

We will consider that time is slotted on two scales: a generic time slot t of length T will contain M minislots indexed by τ. At each time slot t there is a resource assignment decision to make, represented by the vector $\mathbf{x}[t] \in \mathbb{R}^E$ assigning resources to each of the E primary users during time slot t. The vector $\mathbf{x}[t]$ can be constructed, for example, by indicating in its e-th entry the amount of resources assigned to user e. Similarly, for every minislot τ of time slot t, there is a resource assignment matrix $\mathbf{y}[t,\tau] \in \mathbb{R}^{U\times E}$. In the same way as with $\mathbf{x}[t]$, this second resource assignment variable indicates on its u, e entry the amount of resources of each primary user e reassigned to the secondary user u. By summing over row u of matrix $\mathbf{y}[t,\tau]$ we get the total number of resources assigned to user u, while by summing over column e we get the total number of resources previously assigned to e and now reassigned to users of the U group. We will denote as $\mathbf{y}[t] = [\mathbf{y}[t,0], \mathbf{y}[t,1] \dots \mathbf{y}[t,M-1]] \in \mathbb{R}^{E\times U\times M}$ the minislot based assignments during time slot t.

We will consider two kinds of random variables. On the one hand a random vector $\mathbf{R}[t] \in \mathbb{R}^{E+U}$, that will be drawn at the beginning of each time slot. This vector represents the user's ability of exploiting the resources; for instance, maximum attainable rate over a wireless channel for all users, or an electrical generation capacity. On the other hand, random vector $\mathbf{D}[t] = [\mathbf{D}[t,0], \mathbf{D}[t,1] \dots \mathbf{D}[t,M-1]] \in \mathbb{R}^{M\times U}$, with realizations for every minislot τ in time slot t. This vector represents a certain requirement for secondary users; for example, a data rate demand to be satisfied. As we mentioned, these requirements only correspond to secondary users, and we consider primary users to be greedy towards resources (resources are shared among them in order to maximize some utility). It is important to note that random vector $\mathbf{D}$ has realizations on a minislot basis, thus not being known when resource assignment to primary users happens. Suppose all random vectors ($\mathbf{D}$ and $\mathbf{R}$) and policy assignments ($\mathbf{x}$ and $\mathbf{y}$) are coupled through a set of H restrictions. Choosing both assignment policies in order to maximize a cer-

tain function of the variables leads to a very general formulation as a constrained stochastic optimization problem:

$$\max_{\mathbf{x}[t],\mathbf{y}[t]} f(\mathbf{x}[t],\mathbf{y}[t],\mathbf{R}[t],\mathbf{D}[t]) \tag{6.1}$$

s.t.:

$$g_i(\mathbf{x}[t],\mathbf{y}[t],\mathbf{R}[t],\mathbf{D}[t]) \leq 0, i = 0,\ldots,H-1. \tag{6.2}$$

These optimization problems depending on random variables with realizations observed after the decision instant are very hard to solve. Note the difficulty of making resource assignment decisions at the beginning of a time slot when resource demands and resource assignment decisions for the same time period are unknown and bound to affect all utilities.

Our proposed supervised learning framework on its most general form will try to approximate the optimal solution $(\mathbf{x}^*,\mathbf{y}^*)$ in the following manner. After a time slot has elapsed, all random variables realizations are known and the optimal solution can be found offline. Therefore the approximate solution will be learned from the combined data set of the random variables' realizations and the optimal allocation policy. The set of constrains has to be ensured after the supervised agents make their assignments, which can be achieved through some projection over the feasible space. The algorithm consists of the following steps during training:

1. Given K previous realizations of the random variables $\mathbf{R}$ and $\mathbf{D}$, we solve the optimization problem and find optimal assignments $(\mathbf{x}^*[t],\mathbf{y}^*[t])$ for every $t \in [0,K]$.

2. We train supervised learning agent 1 with available random variables realizations as features and the optimal assignments as targets for every t. The features may include the last K_1 realizations. As an example, for time slot q, realizations $\mathbf{R}[t]$ with $t \in [q-K_1 \ldots q]$ and $\mathbf{D}[t]$ with $t \in [q-K_1 \ldots q-1]$ will be considered as features and $\mathbf{x}^*[q]$ will be choosen as target.

3. We train a second supervised learning agent on a minislot basis, considering the last $K_2 \times M + p$ minislots' realizations of random vectors $\mathbf{D}$ and $\mathbf{R}$ in order to predict an approximation to the optimal policy for minislot p in time slot q. In this case, features will be $\mathbf{R}[t]$ with $t \in [q-K_2 \ldots q]$, $\mathbf{D}[t]$ with $t \in [q-K_2 \ldots q-1]$, and $\mathbf{D}[q,\tau]$ with $\tau \in [0 \ldots p]$. Finally, another feature that will be used is the inferred value of $\mathbf{x}^*[q]$. Our target will be the minislot assignment policy $\mathbf{y}^*[q,p]$.

Together, our learned agents will be able to forecast online resource assignments $(\hat{\mathbf{x}^*}[t],\hat{\mathbf{y}^*}[t,\tau])$. Note that the training phase is done offline, with no time constrains, which enables step 1 to eventually solve non-convex optimization problems. Also observe that when using a time slot basis, mini slot features have to be somehow summarized into a time slot basis (for example, as the aggregated demand over the time slot); similarly, time slot based data has to be transformed

into mini slot based realizations. Both learning agents offer the freedom to choose architectures over the vast world of supervised learning, according to the problem's nature and/or domain knowledge over the problem and random variables $\mathbf{R}$ and $\mathbf{D}$.

When working online, the procedure is very simple:

1. At time slot t, features are available for supervised agent 1 to predict assignment $\hat{\mathbf{x}}[t]$.
2. Projection of $\hat{\mathbf{x}}[t]$ in order to satisfy restrictions over $\mathbf{x}$.
3. For every minislot τ, features are available for the second agent to predict assignment $\hat{\mathbf{y}}[t, \tau]$.
4. Projection of $\hat{\mathbf{y}}[t, \tau]$ in order to satisfy restrictions over $\mathbf{y}$.

It is important to remark that the projection is needed in order to satisfy restrictions and propose policies belonging to the feasible space of solutions. There are different ways of projecting that can be used in order to ensure that restrictions are complied, which we will further discuss in subsection 7.3. Observe that another approach could be to try to learn the second group's demand, as in learning to predict $\mathbf{D}$ for the next time slot.

In the next chapter we introduce the coupled problem that arises from URLLC and eMBB coexistence and employ our proposed learning framework in order to find optimal allocation policies.

Chapter 7

URLLC and eMBB coexistence in 5G NR

As we have introduced, 5G NR establishes three types of users. Two of these users, URLLC and eMBB, share resources of the OFDMA grid. But their coexistence comes at a price: puncturing over eMBB assigned resource is done in order to satisfy URLLC demand. This degrades performance for eMBB users, which poses a challenge: to minimize puncturing effects on eMBB users, all the while satisfying URLLC demand. We now develop on the utilization of the proposed learning framework from the last chapter in order to find optimal resource assignment policies for URLLC and eMBB coexistence.

7.1 Definitions and optimization problem

In the frequency domain, the OFDMA grid is divided into sub-channels, whereas in the time domain it is split into slots and minislots (a set of OFDM symbol times). The 5G NR standard defines the minimum set of OFDMA resources that can be assigned to an eMBB user as a physical Resource Block, which as presented in the previous part corresponds to what we call a time slot and a set of sub-channels. Considering the complete OFDMA grid, the system disposes of a total of N_{RB} resource blocks.

At each slot t a first scheduler distributes among active eMBB users the set of resource blocks. The system has a set of eMBB users in each time slot t: $e \in \{1, .., E\}$. We will assume that eMBB users' traffic follows a full buffer model (i.e. eMBB users always have data to transmit). Differently from the first part, we now consider that there is a fixed number of eMBB users, due to the very short time scale of the puncturing mechanism. Figure 7.1 pictures actions from the first scheduler with respect to eMBB users.

As seen in the previous section, and using the same notation for ease of exposition, we define a time slot t as divided into time minislots indexed by $\tau \in \{1, ..., M\}$. The system has a set of U URLLC users, $u \in \{1, .., U\}$. Each URLLC user u at (t, τ) has a byte demand $d_u(t, \tau)$, which is the realization of a random variable D_u,

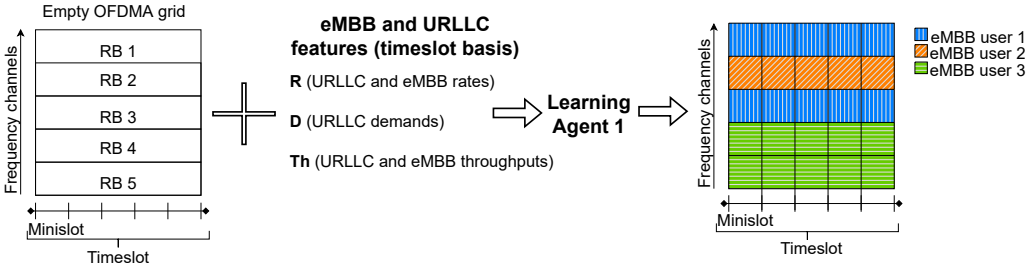


Figure 7.1: First resource allocation scheduling, corresponding to eMBB users on a time slot basis. Scheduling of eMBB users is prior to scheduling of URLLC users. Features used by the scheduling agent are composed of eMBB and URLLC rates, throughputs and demands of past time slots.

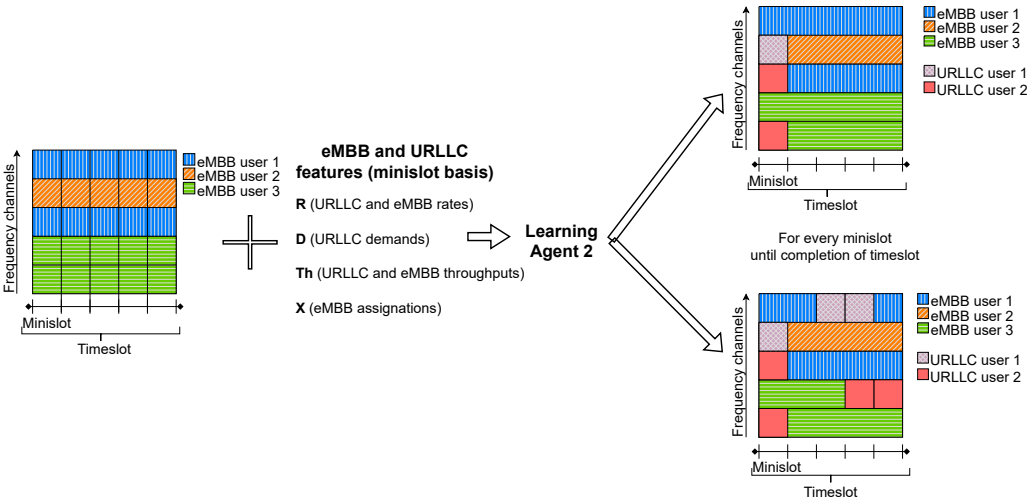


Figure 7.2: The second agent schedules resources to URLLC users on a minislot basis. These resources were already assigned to eMBB users, and may thus inflict losses to eMBB communications.

known at the beginning of each minislot. In order to fulfill URLLC traffic demands, a second scheduler assigns at each minislot resource blocks that share the same set of sub-channels of an RB assigned to an eMBB user, but the time duration of a minislot. Figure 7.2 allows readers to visualize the minislot based puncturing action for URLLC users, overwriting assigned resources to eMBB users.

All users have a wireless channel with the base station that varies between time slots depending on the channel's fading and noise. We assume the base station knows the SNR of each user at all times, which is constant over a time slot. Using the 5G NR standard, as in previous chapters the system can find the maximum reachable rate for any user by using the measured SNR and the corresponding coding and modulation. This SNR evolution and its impact on the maximum attainable rate introduces a time slot and user dependent random variable R_e for eMBB users and R_u for URLLC users, with realizations $r_e(t)$ and $r_u(t)$ known at the beginning of each time slot. We consider all users' peak rates at time slot t

on vector $\mathbf{R}[t] = [r_e(t), r_u(t)]^{e\in\{0...E\},u\in\{0...U\}}$ and demands for URLLC users on vector $\mathbf{D}[t] = [d_u(t)]^{u\in\{0...U\}}$.

Typically these random vectors have correlations between them and temporal correlations during consecutive time slots and minislots. These dependencies are to be exploited in order to learn accurate predictions, as well as the impact of the scheduling policies on the overall utility.

We consider the vector $\mathbf{x}[t] = \{x_1(t), ..., x_E(t)\}$, where $x_e(t)$ represents the total number of RBs assigned in slot t to eMBB user e (e.g. $\mathbf{x}[t] = [2, 1, 2]$ in figure 7.1). We define as well the RB assignment to URLLC users as the vector $\mathbf{y}[t] = \{y_1(t), ..., y_E(t)\}$, where $y_e(t)$ is the sum over all minislots on slot t of the number of punctured RBs being assigned to eMBB user e (e.g. $\mathbf{y}[t] = [0.6, 0.2, 0.6]$ in figure 7.2). In a similar way, we represent by $y_{e,u}(t, \tau)$ the number of RBs reassigned from eMBB user e to URLLC user u at time slot t and minislot τ (e.g. $\mathbf{y}_{2,1}(t, 0) = 0.2$ in figure 7.2), and thus $y_u(t, \tau) = \sum_e y_{e,u}(t, \tau)$ is the total resource block assignment for URLLC user u.

Please note that for a given URLLC user u, its byte demand $d_u(t)$ has to be satisfied over a minislot, meaning that assignment $y_u(t, \tau)$ multiplied by the time slot peak rate $r_u(t)$ has to equal the users demand. As puncturing of a certain eMBB user cannot exceed its original total amount of resources, $y_e(t) \leq Mx_e(t)$, $\forall e, t$. Combining the aforementioned analysis, puncturing has to verify the following equations:

$$y_e(t) = \sum_{\tau=1}^{M} \sum_{u=1}^{U} y_{e,u}(t, \tau) \tag{7.1}$$

$$d_u(t, \tau) = \sum_{e=1}^{E} y_{e,u}(t, \tau) \times r_u(t) \tag{7.2}$$

As we mentioned before, we want the resource allocation to eMBB and URLLC users to be performed so that a certain utility function f is maximized. We will consider this reward to be a function of random variables realizations: $\mathbf{r}[t]$, $\mathbf{d}[t]$ and of course of assignments $\mathbf{x}[t]$, $\mathbf{y}[t]$. In particular, the decision on $\mathbf{y}[t]$ will impact on the performance as perceived by the eMBB users, as part of their transmission is overwritten.

We assume that by knowing a time slot's realizations (and past if needed), the scheduler can find the optimal assignment in the present slot t:

$$\mathbf{x}^*[t], \mathbf{y}^*[t] = \arg \max_{\mathbf{x}[t],\mathbf{y}[t]} f(\mathbf{x}[t], \mathbf{y}[t], \mathbf{r}[t], \mathbf{d}[t]) \tag{7.3}$$

The total utility of our system f will be defined as the sum over eMBB users' utility, which is in turn defined as a per user utility function $\mathcal{U}$ constructed as some combination over eMBB users' throughput. As an example, we could use as global and per user utilities $f = \sum_e \mathcal{U}(th_e(t)) = \sum_e \log(1 + th_e(t))$. Throughput $th_e(t)$ for user e at slot t is defined as a running mean with averaging ratio α over the effective rate (see eq. (7.5)), which is defined as a function of the assigned resources $x_e(t)$, maximum attainable rate $r_e(t)$ and, as we mentioned before, the

rate loss due to puncturing of user's e transmission (a function we will denote as $L(x_e(t), y_e(t))$).

We consider throughput for all eMBB users at time t in vector $\mathbf{th}[t] \in \mathbb{R}^E$.

$$f(\mathbf{x}[t], \mathbf{y}[t], \mathbf{r}[t], \mathbf{d}[t]) = \sum_e \mathcal{U}(th_e(t)) \tag{7.4}$$

$$th_e(t) = \alpha th_e(t-1) + (1-\alpha)c_e(t) \tag{7.5}$$

$$c_e(t) = x_e(t) r_e(t) (1 - L(x_e(t), y_e(t))) \tag{7.6}$$

Remark 6 (On the inclusion of fairness) Observe that by applying the utility function on the throughput, in turn a running mean over the rate, we are implicitly taking into account a long-term fairness criteria. Given that eMBB users' timescale is much larger than the URLLC's lifespan, we consider eMBB as full buffer (i.e. they never leave the system). This results in a very different scenario from the dynamic setting analyzed in the first part (i.e. considering the proposed average a-posteriori makes no sense in this static problem).

All in all, the resulting optimization problem is as follows:

$$\max_{\mathbf{x}[t], \mathbf{y}[t]} \sum_{e=1}^{E} \mathcal{U}(th_e(t)) \tag{7.7}$$

s.t.:

$$th_e(t) = \alpha th_e(t-1) + (1-\alpha)c_e(t) \tag{7.8}$$

$$c_e(t) = r_e(t)(1 - L\left(x_e(t), y_e(t)\right)) \tag{7.9}$$

$$\sum_e x_e(t) = N_{RB} \tag{7.10}$$

$$\sum_u y_{e,u}(t,\tau) \leq x_e(t) \ \forall \ t, e, \tau \tag{7.11}$$

$$y_e(t) = \sum_\tau \sum_u y_{e,u}(t,\tau) \ \forall \ t, e \tag{7.12}$$

$$\sum_e y_{e,u}(t,\tau) = \frac{d_u(t,\tau)}{r_u(t)} \ \forall t, \tau, u \tag{7.13}$$

$$y_{e,u}(t,\tau) \geq 0 \ \forall \ e, u, t, \tau \tag{7.14}$$

$$x_e(t) \geq 0 \ \forall \ t, e \tag{7.15}$$

Restrictions ensure that definitions hold to the puncturing model. For instance, any resource allocation has to be positive, and there cannot be more resources punctured to an eMBB user than the ones originally assigned. Constraints will be further analyzed in subsection 7.3.

Let us further discuss the loss function L. As puncturing of eMBB and URLLC in 5G is a hard to solve optimization problem, different scenarios have been proposed in the literature, mainly using heuristics yielding good results for particular

settings. Other approaches, as [119], formulate optimization problems very similar to our own, and consider convex functions for $\mathcal{U}$ and L. They prove that some policies are optimal under certain conditions, notably a proportional fair assignment for eMBB resources and a random assignment for URLLC resources. However, digital communications' performance exhibit a non-convex response to interference, that can be viewed as a threshold penalty: if interference is larger than a certain value, then the message is completely lost [144].

We will thus test our framework using two kinds of loss functions: convex (quadratic) and the more realistic non-convex threshold. This allows a comparison with regard to optimal solutions with theoretical guarantees (as in the convex scenarios), as well as with state of the art proposed solutions on the threshold case, which is a much more realistic approach, and closer to application. Our proposed learning method proves to be able to adapt to different scenarios all while maintaining close to optimal mean utility for eMBB users.

The offline optimization phase may be performed using any chosen solver when the loss and utility functions are convex. For the threshold scenario, even if this is not a convex optimization problem, it can be transformed and solved by optimizing 2^E convex optimization problems. Even though challenging, the computational complexity of solving the threshold scenario only depends then on the number E of EMBB users, and neither on the number of active URLLC users nor the number of resource blocks the system disposes.

As stated, in a general non-convex scenario, the optimal policy has to be found by trying all possible policies, which is highly time and resources consuming. Still, in our framework this is feasible since solving the optimization problem and the training phase are both done offline.

7.2 Learning formulations

An important consideration to be made over resource block assignments and puncturing is the strong time limitation a 5G scheduler meets. As an example to fix this idea, a time minislot in 5G is on the order of the hundreds of nanoseconds. This is why supervised learning is an appropriate choice in order to exploit statistical information: once trained, execution is very quick.

Different supervised learning methods could be exploited. The chosen learning machine will typically depend on domain knowledge, mainly on the nature of the random variables, depending if and how much we know about them. The range of possible choices goes from well known methods as Random Forest or Support Vector Machines up to the newest deep learning architectures, including custom made algorithms. In section 8 we show results with different learning agents.

Whichever learning method is chosen, with our proposed statistical learning framework they will all share features and target. Our first scheduler makes the prediction of $\mathbf{x}^*[t]$, for which we use the following features:

$$feats_{\mathbf{x}^*[t]} = [\mathbf{r}[t] \dots \mathbf{r}[t-K_R], \mathbf{d}[t-1] \dots \mathbf{d}[t-K_D], \mathbf{th}[t-1] \dots \mathbf{th}[t-K_{th}]] \quad (7.16)$$

We include rate for all users (of both kinds) for this time slot and past K_R time slots. We also consider eMBB users' throughput and URLLC demand for the past K_D and K_{th} time slots respectively; observe that neither eMBB's throughputs nor URLLC's demands are available at the present time slot t. Demand from URLLC users has a different timescale, so including the minislot demand has little significance. Instead, we consider the aggregated sum of URLLC users' demands over each time slot. With regard to the more general formulation described in section 6.3, note that we have introduced as a feature the throughput instead of the assignment x, both being directly related: with the assignments and rates the throughput can be easily obtained. We consider the throughput to be a more direct feature than to introduce the past assignments for the present learning poblem.

Our second scheduler will be working on a minislot time basis. In order to match dimensions over a minislot basis, timeslot features are extended to the M minislots (e.g. rates). Besides past and present rates for all users and throughput for eMBB users, we use URLLC users demands per minislot (past and present), and resource assignment for eMBB users $\mathbf{x}^*[t]$. This results in:

$$feats_{\mathbf{y}^*[t,\tau]} = [\mathbf{r}[t], \mathbf{d}[t,\tau] \dots \mathbf{d}[t,\tau-K_D], \mathbf{th}[t-1], \mathbf{x}^*[t]] \quad (7.17)$$

7.3 Optimization constraints

The regression problem formulated in our approach must take into account the constraints of the system. In the estimation of the eMBB assignments, we must ensure the following constraints:

$$\sum_e x_e(t) = N_{RB} \quad (7.18)$$

$$x_e(t) \geq 0 \ \forall \ e \quad (7.19)$$

Regarding the estimation of the URLLC allocation we must ensure the following constraints:

$$\sum_u y_{e,u}(t,\tau) \leq x_e(t) \ \forall \ e,t,\tau \quad (7.20)$$

$$\sum_e y_{e,u}(t,\tau) = \frac{d_u(t,\tau)}{r_u(t,\tau)} \ \forall \tau,u,t \quad (7.21)$$

$$y_{e,u}(t,\tau) \geq 0 \ \forall \ e,u,t,\tau \quad (7.22)$$

These constraints must be imposed on the learning regression system. There are two types of constraints in the previous equations. The first type guarantees that the output vector of the regression ($\mathbf{x}[t]$ or $\mathbf{y}[t,\tau]$) has all terms greater or equal than zero and the sum of its components must equal some fixed value: the total amount of resources for $\mathbf{x}$, and the demand satisfaction for $\mathbf{y}$. This type of constraints can be imposed using a well known method consisting of a logarithmic transformation over the normalized target vector, as described in [153]. This transformation ensures that the regression output satisfies fixed sum constraints.

The other type of constraint is:

$$\sum_{u} y_{e,u}(t,\tau) \leq x_e(t) \ \forall \ e,\tau \tag{7.23}$$

This constraint must be imposed in the regression of the $\mathbf{y}$ values where the x_e values were calculated in the first regression. This type of constraint is more difficult to impose together with the first type of constraints. In this case after the regression with the log transformation, we assure that all the constraints for $\mathbf{y}$ are verified by projecting the output values obtained $\mathbf{y}^*[t,\tau]$ to the constraint space.

$$\min_{\mathbf{y}} \sum_{e} (y_{e,u}(t,\tau) - \hat{y}_{e,u}(t,\tau))^2 \tag{7.24}$$

s.t.:

$$\sum_{e} y_{e,u}(t,\tau) = \frac{d_u(t,\tau)}{\hat{r}_u(t)} \ \forall u \tag{7.25}$$

$$\sum_{u} y_{e,u}(t,\tau) \leq x_e(t) \ \forall e \tag{7.26}$$

$$y_{e,u}(t,\tau) \geq 0 \ \forall e,u,t,\tau. \tag{7.27}$$

The projection is realized for each minislot τ.

This page has been intentionally left blank.

Chapter 8

Simulations and Results

In order to validate our system model and learning procedure, we compare our results to the proposed solution of [119], which we will refer to as 'baseline' in experiments and tables. In a nutshell, this heuristic consists of a proportional fair assignment for eMBB users and a random assignment for URLLC puncturing. This comparison might be unfair, given that the solution is only optimal in certain specific scenarios (for example, for convex loss functions), but it is a state of the art algorithm for puncturing optimization, and as such serves well as a baseline in order to compare our proposed learning procedure behavior. We use well known supervised learning methods such as the classic Support Vector Machines (SVM) and the Recurrent Neural Network (RNN), and implement the learning algorithm over different synthetic experiments.[1]

We selected three particularly interesting cases. On the first experiment we apply our proposed framework to a convex problem (the loss function is convex and the utility is concave). This type of 'convex' scenarios have well known solutions (as shown in [119]), which enable us to test a simple setting in order to verify the correct behavior of our learning agents. On the second and third scenarios we use a threshold loss: if more than a certain amount of assigned resources are punctured for an eMBB user, all communication will be considered lost (meaning the rate for that user will be 0 for the whole time slot). On the other hand, if that threshold is not surpassed, communications (and thus eMBB user rates) are unaltered. This non-convex scenarios are studied by [119], and while their proposed heuristic works reasonably well when URLLC demand is low, as traffic increases the proposed solution grows afar from optimal policies. That is why we finally introduce a synthetic scenario with larger URLLC traffic loads, maintaining the threshold penalty as the loss function. As mentioned before, the system's reward is defined as $f(t) = \sum_e \mathcal{U}(th_e(t)) = \sum_e \log(1 + th_e(t))$.

Besides our proposal and the solution described in [119], we compare our results with a 'random agent'. This agent will assign a fixed number of resource blocks to eMBB users (total number of resource blocks over number of eMBB users), and a random number of resource blocks to URLLC users. The idea with this

[1] All code can be found at `https://gitlab.fing.edu.uy/ai45g/puncturing-ai45g`.

extremely simple agent is to obtain a minimum performance so as to quantify the gain obtained by the other two methods. For all heuristics we ensure restrictions are met by projecting the found resource assignments over the feasible space of solutions.

Finally, we also compare against the optimal allocation, which can be easily obtained when the problem is convex. When considering a threshold loss the optimal assignment can also be found if we consider it as 2^E convex optimization problems, as described in the past chapter. For the presented experiments we derived the optimal solution for comparison purposes, but at the price of costly computation, unfeasible in real-time applications.

Simulation parameters can be found in table 8.1. The total number of resource blocks is $N_{RB} = 270$ for all simulations. Channel evolution for all users follows a finite Markov process as proposed in [154]. The URLLC users' traffic demand follows a two state finite Markov process, being URLLC's demand turned on and off. All convex optimization problems were solved using **cvxpy** [155].

When using SVM, the regression for learning both policies is obtained by using Scikit-learn Support Vector Regression (SVR) software [156]. Parameters C and gamma of the Radial Basis Function (RBF) kernel are obtained via grid search.

Recurrent Neural Networks are well known deep learning architectures particularly suited to learn and capture sequential dependencies (e.g. temporal sequences), making them a natural choice for testing our proposed statistical learning method. We used the keras [157] implementation of recurrent neural networks, experimenting with LSTM, GRU and vanilla RNN. We only present experiments concerning the implementation achieving highest results, and we used a very simple architecture: an LSTM (or GRU) layer followed by a fully connected layer for regression purposes. In all cases, the learning rate follows an exponential decay and the loss function is computed by using mean squared error. The parameters of the RNN (number of neurons, learning rate, normalization, sequence length) were also chosen using grid search, and details for each experiment can be found in table 8.1. When using neural networks, normalization was used prior to learning in order to accelerate convergence and obtain better scores.

We first tried the proposed learning method using a quadratic loss function, defined as $L(x_e, y_e) = \left(\frac{y_e}{x_e \times M}\right)^2$. The predicted policy obtained is very close to optimal as can be seen in figure 8.1, and the average utility is over 99% of the optimal utility (see tables 8.2, 8.3). Observe that in this case all policies behave really well; in a convex scenario the puncturing policy is not that important in order to achieve good results, and an average distribution of resources over eMBB users will be optimal in expectation. Results are good but also expected, because by using a simple convex loss function the optimal policy should be easily learned. This is caused because losses by puncturing are not so dependent on the punctured eMBB user, as all eMBB users fare similarly and contribute proportionally to the global utility. Also, on average a mean distribution of resources for eMBB users will be optimal, meaning the random heuristic proposed should (and effectively does) yield good results. Yet, results are encouraging, as our proposed learning method fares as well as heuristics that are optimal on mean (as the Baseline).

Table 8.1: Table of Parameters

	Scenario 1	**Scenario 2**	**Scenario 3**
Loss model	**Quadratic**	**Threshold**	**Threshold**
eMBB users	20	10	10
URLLC users	5	3	3
Minislots (M)	3	3	3
α	0.8	0.8	0.8
Threshold	-	0.11	0.11
T_{train}	7700	7700	19700
T_{test}	300	300	300
SVM-C (x)	50×10^3	1000	200
SVM-G (x)	1×10^{-4}	1×10^{-6}	5×10^{-6}
SVM-C (y)	100	5×10^3	20×10^3
SVM-G (y)	1×10^{-7}	5×10^{-6}	5×10^{-7}
Max Iterations	1×10^4	1×10^5	1×10^5
RNN Cell	GRU	LSTM	LSTM
Hidden Layer (x)	24	24	24
Hidden Layer (y)	128	32	32
Normalization	No Scaler	MinMax	MinMax
Learning rate (x)	1×10^{-3}	1×10^{-4}	1×10^{-4}
Learning rate (y)	5×10^{-4}	5×10^{-4}	5×10^{-4}
Sequence length (x)	2	4	4
Sequence length (y)	6	12	12

We then applied the framework in a more complex scenario, using the threshold loss function formulation (see figure 8.2). Results are even more encouraging, even if they are not as near optimal as in the quadratic case, the utility achieved is almost 90% of the utility the optimal policy would obtain. The difficulty when using the threshold loss function is that when an eMBB user is punctured it may result in a null rate ($r_e(t) = 0$), which means the discontinuities observable in the figures and an evident larger impact on the utilities for different policies. For an agent to learn these highly discontinuous patterns is not a trivial task, and yet our proposed method is able to grasp statistical correlations and features' relationships in order to choose close to optimal policies.

In the work of [119], authors find greater differences between their proposed heuristics and optimal solutions as the traffic load increases for URLLC users, when using a non-convex loss function (for example a threshold loss). Even if the discontinuities may appear more often (eMBB users with $r_e = 0$), this may actually be beneficial for the learning agent. For instance, it will learn to puncture those eMBB users with $r_e = 0$, since their limit threshold has already been surpassed.

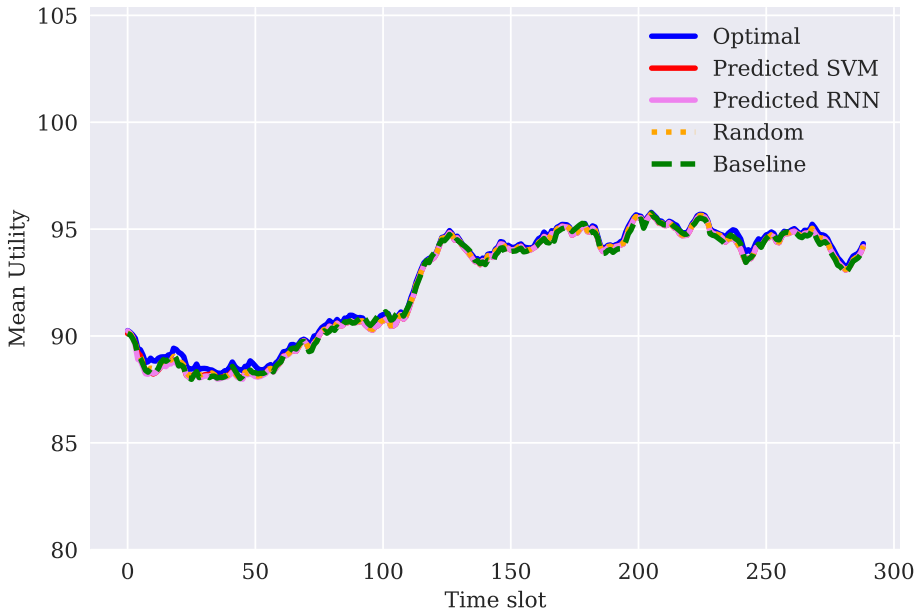


Figure 8.1: Reward during online test for Scenario 1. All heuristics fare almost as well as the optimal solution, given the convex setting for the trial: both loss function (quadratic) and utility are convex. In this case the puncturing policy has a lesser effect on global utility, and a mean distribution of resources for eMBB will have a good performance on expectation.

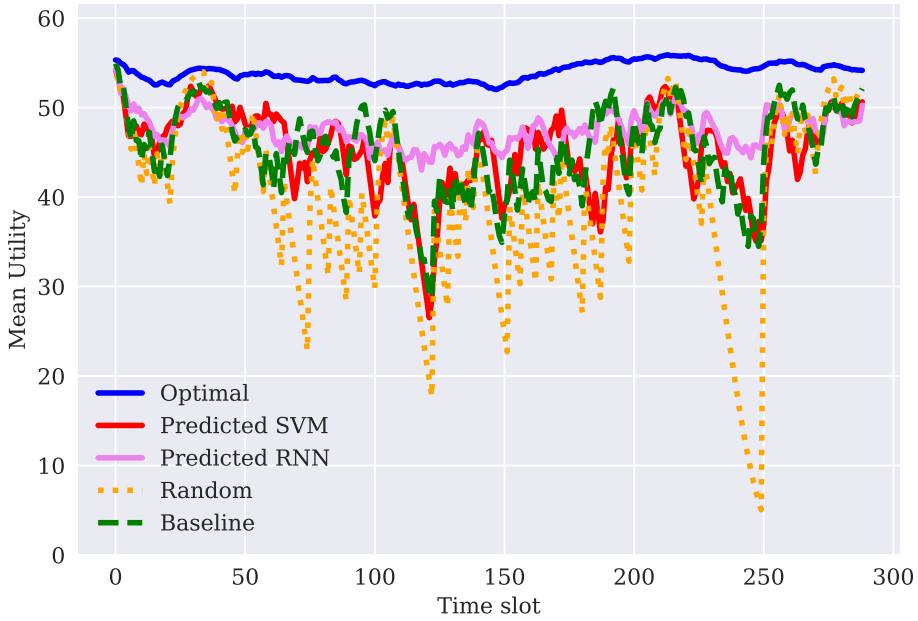


Figure 8.2: Reward during online test for Scenario 2 (Threshold Loss). During test, URLLC traffic demand is low, allowing agents to either allocate all puncturing without losses or to puncture one eMBB user over the threshold limit. This scenario is harder to learn, having two very distinct scenarios (either send all puncturing to a user of distribute it evenly among all users).

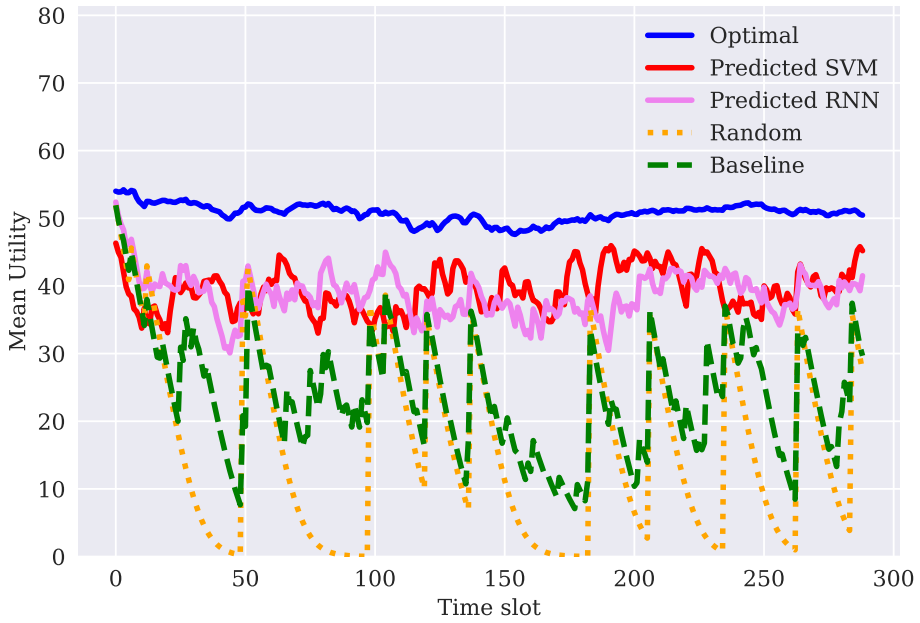


Figure 8.3: Reward during online test for Scenario 3 (Threshold Loss). In this case during test URLLC traffic demand is higher, which induces more losses because of the threshold penalty. Even if global utility is lower because of heavier puncturing, the learning problem is easier to approximate.

Table 8.2: Mean reward (overall utility) for the compared algorithms over different scenarios. Reward is calculated by applying a fairness utility function to the eMBB users' throughput as in eq. 7.4.

	Mean Reward				
Loss model	**Baseline**	**Random**	**SVM**	**RNN**	**Optimal**
Quadratic	92.43	92.47	92.43	**92.48**	**92.63**
Threshold (low demand)	45.30	40.13	45.32	**47.62**	**53.94**
Threshold (high demand)	24.21	16.68	**39.41**	38.79	**50.78**

This is evidenced in the third scenario, where URLLC's traffic load is larger (see figure 8.3).

As expected, utilities are farther apart from optimal than in previous experiments, although our framework fares much better than the rest of the proposed solutions: while the other heuristics attain less than 50% of optimal utility (and almost vanishing at certain time-slots), our proposed learning method achieves over 75% of the optimal utility in all cases. Results are summarized in tables 8.2 and 8.3. Table 8.2 compares achieved utilities for the optimal policy, our proposed algorithm, the baseline and the random policy. In table 8.3, the obtained rewards are normalized with respect to the optimal value to derive a performance metric.

Table 8.3: Performance with respect to the optimal policy's utility for proposed and baseline heuristics.

	Performance (%)			
Loss model	**Baseline**	**Random**	**SVM**	**RNN**
Quadratic	99.78	99.83	99.78	**99.84**
Threshold (low demand)	83.98	74.40	84.02	**88.28**
Threshold (high demand)	47.68	32.85	**77.61**	76.39

Other classic supervised learning tools (Random Forest, Fully Connected Neural Networks) can be easily used as a proof of concept of the versatility of the proposed framework. Results are not included, but implementations can be found in the project's repository.

8.1 Summary

We have presented a framework for solving online two constrained optimization problems coupled over two different timescales. Our approach uses the system's state and statistical correlations in order to train two supervised agents, one for each timescale policy. Our framework consists of an offline learning phase, in which we train the agents with the system state and the optimal assignments obtained offline. The supervised agents then apply the estimated optimal assignments online.

The proposed framework is instantiated on resource allocation with puncturing in 5G networks. The optimization of 5G resources with puncturing is in challenging to solve, if possible at all. URLLC and eMBB coexistence implies a downgrading of eMBB communications by reassigning eMBB users' resources to URLLC users in order to satisfy URLLC's demand. We solve the optimization problem arising from the aforementioned coexistence, focusing on maximizing a throughput based utility for eMBB users all while satisfying URLLC demands.

The effectiveness of the proposed approach was demonstrated through a variety of simulations. We used different scenarios and established realistic settings to prove the framework's performance, achieving results above current state of the art solutions. We were able to approximate online optimal solutions, even in non-convex scenarios in which state of the art techniques do not approximate well the optimal solution. On all scenarios our agents perform better than compared heuristics, achieving up to a 50% increase on eMBB users' utility with regards to well known state of the art proposals.

In future works it would be interesting to try out other learning algorithms, modifying restrictions and developing feature engineering, in order to find even better policies approximations. It would also be interesting to explore on imple-

menting a combined training of both learning agents, by introducing in the loss function a utility divergence penalty. This may allow to exploit domain knowledge in order to achieve better results. Finally, application of this statistical learning method over a broader series of problems over several timescales involving constrains, as energy management, would be interesting in order to further validate the proposal.

This page has been intentionally left blank.

Chapter 9

Conclusions and Future Work

This thesis presented our research on the integration of artificial intelligence into next-generation communication networks. Although AI has achieved rapid expansion across many disciplines, incorporating intelligent control into communication systems remains a demanding challenge. The robustness of current network architectures and the need for reliability constrain the adoption of adaptive, AI-driven algorithms for optimizing resource utilization in data and wireless communications.

However, as communication scenarios multiply and network complexity increases, emerging technologies such as 5G and upcoming 6G standards are designed to embrace AI-driven management. Their objectives include optimizing the use of available resources and providing enhanced, context-aware services.

9.1 Main Contributions

This work addressed two central problems in resource allocation for modern communication systems, focusing on the time and frequency dimensions fundamental to wireless access.

9.1.1 User Association with Fairness-Aware Reinforcement Learning

Among the many challenges in resource optimization for wireless communications, this thesis focuses on the user association problem, namely the decision of which base station should serve an incoming user. For this purpose, we introduced a reinforcement learning framework for dynamic user association in wireless networks, referred to as GROWS. By leveraging the ability of reinforcement learning to address sequential decision-making problems, together with the capacity of graph neural networks to represent relational structure, GROWS efficiently tackles the UA problem and learns stable association policies.

Within this framework, we develop three problem formulations. The first is a naive reinforcement learning approach, in which we introduce an RL+GNN algorithm based on a simplified state representation. Simulation results show

that, despite its simplicity, this formulation already leads to improved resource utilization, achieving higher data rates and lower rejection probabilities.

As a second formulation under the GROWS umbrella, we consider a decentralized computation scheme that incorporates slot-level fairness and explicitly models user rejection as a valid action. This formulation accounts for connected users' dynamics, including departures, which requires maintaining information about active connections. To preserve scalability, the proposed POMDP state summarizes user-level information through relevant statistics, such as remaining sojourn time.

Finally, we formalize the complete MDP underlying the UA problem and derive a POMDP that incorporates fairness by accounting for the achieved reward of completed user sessions, referred to as *average a-posteriori* fairness. Unlike classical approaches that rely on static or purely efficiency-driven objectives, this formulation embeds fairness directly into the reward function. As a result, efficiency, fairness, and stability can be addressed jointly by any reinforcement learning algorithm, without ad hoc reward shaping or algorithm-specific adjustments.

Experiments across multiple scenarios demonstrate that the learned policies outperform standard heuristics and existing fairness-based methods. In lightly loaded conditions, GROWS performs on par with efficiency-oriented heuristics while maintaining balanced resource usage. Under heavy congestion, it selectively limits new admissions to prevent overload, achieving significantly higher per-user rates and shorter service times. Finally, in the real deployment, the learned policy produces both spatially balanced associations and improved resource usage. These results confirm that fairness-oriented learning not only improves user-perceived performance but also enhances overall resource utilization.

9.1.2 Intra-Cell Resource Allocation for Heterogeneous 5G Services

We analyzed the coexistence of the three service types defined in 5G: enhanced Mobile Broadband (eMBB), massive Machine-Type Communications (mMTC), and Ultra-Reliable Low-Latency Communications (URLLC). These different users share common time-frequency resources but operate on distinct temporal scales and with different service requirements.

Our focus was on quantifying and mitigating the disruptive effect of URLLC traffic on ongoing eMBB sessions. We modeled the interaction across time scales and proposed a constrained optimization and supervised learning approach. Simulations validated the model, showing that our method effectively limits the impact of URLLC arrivals while preserving eMBB quality of service.

Together, these contributions illustrate that integrating AI-driven, fairness-oriented decision-making into wireless networks can substantially enhance resource efficiency and user experience. The proposed frameworks also support multi-objective extensions, allowing the inclusion of energy consumption, interference, or handover cost minimization as additional criteria.

9.2 Considerations and Future Work

Our most recent efforts focused on the concept of fairness and its enforcement within reinforcement learning frameworks. This direction holds strong potential: in a world increasingly shaped by algorithmic decision-making, reflecting on *what* (i.e. *for whom*) we optimize is crucial. The proposed MDP reward collection modification allows fairness to be enforced in resource allocation with minimal computational overhead, making it compatible with existing models and algorithms.

Future work can extend this research along several directions:

9.2.1 Learning the hidden MDP

In our last formulation we assume that through statistics the system will be able to model the hidden dynamics that relate the three features per users needed to have a complete MDP representation. An interesting enhancement would be to learn embeddings for these per-user data, and not simply assume a statistical representation that may not be sufficient in higher complexity settings.

9.2.2 Concept drift and continual learning

Our proposed solution has to be trained beforehand to be used online: although it can be trained with a variety of trajectories and scenarios, it may represent an interesting plus to have a continual learning set up in which our model is able to keep learning from the data even when facing concept drift: changes in the underlying statistical relationships (namely user arrivals).

9.2.3 Fairness in Real-World Implementations

Theoretical fairness metrics such as α-fairness or proportional fairness are much too often adopted mechanically in optimization literature. Further studies should validate whether the desired fairness outcomes are effectively achieved in real deployments, and explore more adaptive, context-aware definitions of fairness. In future work, we plan to deepen both the theoretical and experimental analysis of the utility functions employed in the fairness-aware formulations. In particular, exploring how different concave utilities influence convergence, stability, and long-term fairness will help clarify the relationship between short-term reward shaping and global system behavior. For instance, one can expect that using a very steep positive sigmoid utility (e.g. $f(x) = \mathrm{ReLU}((1 - e^{-\alpha x})/(1 + e^{-\alpha x}))$ with large α) would yield a policy that prioritizes user acceptance independently of the achieved rate. Establishing properties of the resulting policy for a family of such functions is the subject of our current research.

9.2.4 Cross-Domain Applications

Beyond telecommunications, the developed framework can be applied to other domains involving dynamic resource allocation and fairness—such as healthcare.

We initiated discussions with colleagues at the Hospital de Clínicas to explore the optimization of emergency department operations, aiming to reduce waiting times. The same modeling principles could guide infrastructure design, for example by determining the optimal number and placement of triage points, analogous to base stations in wireless networks.

9.2.5 Integration of Both Algorithms

A natural continuation involves combining the user association and intra-cell allocation mechanisms into an end-to-end optimization pipeline. The internal allocation policy at each base station would operate locally, while its effects would be indirectly captured by the global user association through the reward structure. Analyzing how local fairness constraints influence global network behavior represents an interesting and open research question.

9.2.6 Multi-Objective Optimization

Extending the proposed formulations to include multiple objectives as minimizing energy consumption, handover costs, carbon footprint and interference would provide a more comprehensive optimization framework. These goals can be expressed either as additional constraints or as terms in the overall objective function.

9.3 Final Remarks

Overall, this thesis contributes to a principled and scalable integration of AI into wireless network design. By embedding fairness into reinforcement learning and graph-based models, we provide a pathway towards intelligent decision-making that can improve both network efficiency and user equity. This work contributes to build future generations of communication systems that are not only adaptive and resource-efficient but also fair and socially responsible.

References

[1] International Telecommunication Union. The impact of covid-19 on the internet ecosystem. 2021.

[2] Manuel Castells. *The Rise of the Network Society*. Wiley-Blackwell, 2010.

[3] Nancy Baym. *Personal Connections in the Digital Age*. Polity Press, 2015.

[4] Jean Twenge. *iGen: Why Today's Super-Connected Kids Are Growing Up Less Rebellious, More Tolerant, Less Happy—and Completely Unprepared for Adulthood*. Atria Books, 2017.

[5] Heike Appel et al. Social media and mental health: A review. *Current Opinion in Psychology*, 36:138–143, 2020.

[6] Jens Malmodin and Dag Lundén. The energy and carbon footprint of the global ict and e&m sectors 2010–2015. *Journal of Industrial Ecology*, 24(4):770–784, 2020.

[7] Lotfi Belkhir and Ahmed Elmeligi. Assessing ict global emissions footprint: Trends to 2040. *Journal of Cleaner Production*, 177:448–463, 2018.

[8] International Energy Agency. Data centres and data transmission networks, 2023.

[9] Jeffrey G. Andrews et al. What will 5g be? *IEEE Journal on Selected Areas in Communications*, 32(6):1065–1082, 2014.

[10] Cisco Systems. Cisco annual internet report (2018–2023). Cisco White Paper, 2023.

[11] Amazon Web Services. Summary of the amazon web services outage (december 2020). AWS Incident Report, 2020.

[12] Ben Tarnoff. *Internet for the People: The Fight for Our Digital Future*. Verso, 2022.

[13] Barry M. Leiner et al. A brief history of the internet. *ACM SIGCOMM Computer Communication Review*, 39(5):22–31, 2009.

[14] Emily M. Bender, Timnit Gebru, et al. On the dangers of stochastic parrots: Can language models be too big? *Proceedings of FAccT*, pages 610–623, 2021.

[15] Wenting Wei, Huaxi Gu, and Baochun Li. Congestion control: A renaissance with machine learning. *IEEE Network*, 35(4):262–269, 2021.

[16] Aleksandr Algazinov, Joydeep Chandra, and Matt Laing. Insight: A survey of in-network systems for intelligent, high-efficiency ai and topology optimization. arXiv preprint arXiv:2505.24269, 2025.

[17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[18] Kate Crawford. *Atlas of AI: Power, Politics, and the Planetary Costs of Artificial Intelligence*. Yale University Press, 2021.

[19] Eric Topol. *Deep Medicine: How Artificial Intelligence Can Make Healthcare Human Again*. Basic Books, 2019.

[20] Hongzi Mao et al. Resource management with deep reinforcement learning. *HotNets*, pages 50–56, 2017.

[21] Khaled B. Letaief, Wei Chen, Yuan Shen, and Zhiguo Ding. The road to 6g: Ai-empowered wireless networks. *IEEE Communications Magazine*, 57(8):84–90, 2019.

[22] Amitabha Ghosh et al. 5g evolution: A view on 5g cellular technology beyond 3gpp release 15. *IEEE Access*, 7:127639–127651, 2019.

[23] S. Singh and J. G. Andrews. Joint resource partitioning and offloading in heterogeneous cellular networks. *IEEE Transactions on Wireless Communications*, 13(2):888–901, 2014.

[24] Apoorv Gupta et al. Energy efficiency metrics for green wireless communications. *IEEE Wireless Communications*, 22(2):12–19, 2015.

[25] Jie Liu et al. Multi-objective optimization for energy and fairness in wireless networks. *IEEE Transactions on Communications*, 67(8):5738–5752, 2019.

[26] Juncheng Jiang et al. Deep reinforcement learning for wireless networks: A review. *IEEE Communications Surveys & Tutorials*, 23(3):1659–1695, 2021.

[27] 3GPP. Technical specification group radio access network; nr; overall description; stage 2 (release 17). 3GPP TS 38.300 V17.4.0, 2023.

[28] Alireza Azari, Petar Popovski, et al. Ultra-reliable low-latency communication: Principles and building blocks. *IEEE Communications Surveys & Tutorials*, 21(3):1569–1593, 2019.

[29] Shahid Raza, Linus Wallgren, and Thiemo Voigt. A survey of low power wide area network technologies for iot. *IEEE Communications Surveys & Tutorials*, 20(3):1821–1844, 2018.

[30] Hamid Shariatmadari et al. 5g use cases and requirements. *Proceedings of the IEEE*, 104(3):593–608, 2016.

[31] Mehdi Bennis, Mérouane Debbah, and H. Vincent Poor. Ultra-reliable and low-latency wireless communication: Tail, risk, and scale. *Proceedings of the IEEE*, 106(10):1834–1853, 2018.

[32] Petar Popovski, Kasper F. Trillingsgaard, Osvaldo Simeone, and Giuseppe Durisi. 5g wireless network slicing for urllc and embb: A communication-theoretic view. *IEEE Access*, 6:55765–55779, 2018.

[33] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[34] Martín Randall, Pablo Belzarena, Federico Larroca, and Pedro Casas. Grows: improving decentralized resource allocation in wireless networks through graph neural networks. In *Proceedings of the 1st International Workshop on Graph Neural Networking*. Association for Computing Machinery, 2022.

[35] Martín Randall, Pablo Belzarena, Federico Larroca, and Pedro Casas. Deep reinforcement learning and graph neural networks for efficient resource allocation in 5g networks. In *2022 IEEE Latin-American Conference on Communications (LATINCOM)*, pages 1–6, 2022.

[36] Martín Randall, Santiago Paternain, Pedro Casas, Federico Larroca, and Pablo Belzarena. User association in wireless networks with distributed gnn-based reinforcement learning. In *2025 12th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pages 352–360, 2025.

[37] Martin Randall, Gonzalo Belcredi, Pablo Belzarena, and Federico Larroca. Learning to solve decision problems over two timescales: An application to 5g puncturing. *Wireless Personal Communications*, 132(4):2603–2623, 2023.

[38] Akhil Gupta and Rakesh Kumar Jha. A survey of 5g network: Architecture and emerging technologies. *IEEE access*, 3:1206–1232, 2015.

[39] Xiaohu Ge, Hui Cheng, Mohsen Guizani, and Tao Han. 5g wireless backhaul networks: challenges and research advances. *IEEE Network*, 2014.

[40] Manuel Eugenio Morocho Cayamcela and Wansu Lim. Artificial intelligence in 5g technology: A survey. In *2018 International Conference on Information and Communication Technology Convergence*, pages 860–865, 2018.

[41] N. Panwar, S. Sharma, and A. Kumar Singh. A survey on 5g: The next generation of mobile communication. *Physical Communication*, 2016.

[42] D. Liu, L. Wang, Y. Chen, M. Elkashlan, K. Wong, R. Schober, and L. Hanzo. User association in 5g networks: A survey and an outlook. *IEEE Communications Surveys 'I&' Tutorials*, 2016.

[43] H. Ramazanali, A. Mesodiakaki, A. Vinel, and C. Verikoukis. Survey of user association in 5g hetnets. In *2016 8th IEEE Latin-American Conference on Communications (LATINCOM)*, 2016.

[44] Qiaoyang Ye, Beiyu Rong, Yudong Chen, Mazin Al-Shalash, Constantine Caramanis, and Jeffrey G. Andrews. User association for load balancing in heterogeneous cellular networks. *IEEE Transactions on Wireless Communications*, 12(6):2706–2716, 2013.

[45] M. Alam, R. Chugh, S. Azad, and R. Hossain. Optimizing cell association in 5g and beyond networks: a modified load-aware biased technique. *Telecommunication Systems*, 2024.

[46] Naor Zohar. To associate or not to associate? a user-based threshold scheme for 5g and beyond networks. *ICT Express*, 2025.

[47] Dantong Liu, Lifeng Wang, Yue Chen, Tiankui Zhang, Kok Keong Chai, and Maged Elkashlan. Distributed energy efficient fair user association in massive mimo enabled hetnets. *IEEE Communications Letters*, 2015.

[48] Phu Lai, Qiang He, Guangming Cui, Feifei Chen, John Grundy, M. Abdelrazek, J. Hosking, and Yun Yang. Cost-effective user allocation in 5g noma-based mobile edge computing systems. *IEEE Transactions on Mobile Computing*, 2021.

[49] Tong Wang and Chuanchuan You. Distributed user association and computation offloading in uav-assisted mobile edge computing systems. *IEEE Access*, 2024.

[50] Ehsan Sadeghi, Hamid Behroozi, and Stefano Rini. Fairness-oriented user association in hetnets using bargaining game theory. *arXiv preprint arXiv:2011.04801*, 2020.

[51] Hesham ElSawy, Ekram Hossain, and Martin Haenggi. Stochastic geometry for modeling, analysis, and design of multi-tier and cognitive cellular wireless networks: A survey. *IEEE Communications surveys 'I&' tutorials*, 2013.

[52] Eitan Altman, Urtzi Ayesta, and Balakrishna J Prabhu. Load balancing in processor sharing systems. *Telecommunication Systems*, 47:35–48, 2011.

[53] Zihao Han, Ting Zhou, Tianheng Xu, and Honglin Hu. Joint user association and deployment optimization for delay-minimized uav-aided mec networks. *IEEE Wireless Communications Letters*, 2023.

[54] Qiaoyang Ye, Beiyu Rong, Yudong Chen, Mazin Al-Shalash, Constantine Caramanis, and Jeffrey G Andrews. User association for load balancing in heterogeneous cellular networks. *IEEE Transactions on Wireless Communications*, 2013.

[55] Daosen Zhai, Huan Li, Xiao Tang, Ruonan Zhang, and Haotong Cao. Joint position optimization, user association, and resource allocation for load balancing in uav-assisted wireless networks. *Digital Communications and Networks*, 10(1):25–37, 2024.

[56] M. Chinipardaz, S. Amraee, and A. Sarlak. Joint downlink user association and interference avoidance with a load balancing approach in backhaul-constrained hetnets. *Plos one*, 2024.

[57] Jonggyu Jang and Hyun Jong Yang. α-fairness-maximizing user association in energy-constrained small cell networks. *IEEE Transactions on Wireless Communications*, 21(9):7443–7459, 2022.

[58] Yeongjun Kim, Jonggyu Jang, and Hyun Jong Yang. Distributed resource allocation and user association for max-min fairness in hetnets. *IEEE Transactions on Vehicular Technology*, 73(2):2983–2988, 2023.

[59] Shaofeng Dong, Jinsong Zhan, Wei Hu, Amin Mohajer, Maryam Bavaghar, and Abbas Mirzaei. Energy-efficient hierarchical resource allocation in uplink–downlink decoupled noma hetnets. *IEEE Transactions on Network and Service Management*, 20(3):3380–3395, 2023.

[60] Yan Lin, Yi Wang, Chunguo Li, Yongming Huang, and Luxi Yang. Joint design of user association and power allocation with proportional fairness in massive mimo hetnets. *IEEE Access*, 5:6560–6569, 2017.

[61] Jihoon Moon, Seungnyun Kim, Hyungyu Ju, and Byonghyo Shim. Energy-efficient user association in mmwave/thz ultra-dense network via multi-agent deep reinforcement learning. *IEEE Transactions on Green Communications and Networking*, 7(2):692–706, 2023.

[62] Zizhen Zhou, Jungang Ge, and Ying-Chang Liang. User association and coordinated beamforming in cognitive aerial-terrestrial networks: A safe reinforcement learning approach. *arXiv preprint arXiv:2502.13663*, 2025.

[63] Nabila Sehito, Yang Shouyi, Haya Mesfer Alshahrani, Mohammad Alamgeer, Ashit Kumar Dutta, Shtwai Alsubai, Lewis Nkenyereye, and Rajesh Kumar Dhanaraj. Optimizing user association, power control and beamforming for 6g multi-irs multi-uav noma communications in smart cities. *IEEE Transactions on Consumer Electronics*, 2024.

[64] Asad Mahmood, Thang Xuan Vu, Symeon Chatzinotas, and Björn Ottersten. Joint optimization of 3d placement and radio resource allocation for per-uav sum rate maximization. *IEEE Transactions on Vehicular Technology*, 72(10):13094–13105, 2023.

[65] Guangming Cui, Qiang He, Xiaoyu Xia, Feifei Chen, Fang Dong, Hai Jin, and Yun Yang. Ol-eua: Online user allocation for noma-based mobile edge computing. *IEEE Transactions on Mobile Computing*, 22(4):2295–2306, 2023.

[66] Ali Nauman, Haya Mesfer Alshahrani, Nadhem Nemri, Kamal M Othman, Nojood O Aljehane, Mashael Maashi, Ashit Kumar Dutta, Mohammed Assiri, and Wali Ullah Khan. Dynamic resource management in integrated noma terrestrial–satellite networks using multi-agent reinforcement learning. *Journal of Network and Computer Applications*, 221:103770, 2024.

[67] Tareq Si Salem, Georgios Iosifidis, and Giovanni Neglia. Enabling long-term fairness in dynamic resource allocation. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 6(3):1–36, 2022.

[68] Pengcheng He, Yijia Tang, Fan Xu, and Qingjiang Shi. Optimization-inspired graph neural network for cellular network optimization. *IEEE Transactions on Mobile Computing*, 2025.

[69] John Rawls. Justice as fairness. *The philosophical review*, 1958.

[70] John Rawls. A theory of justice. In *Applied ethics*. 2017.

[71] R. Mazumdar, L.G. Mason, and C. Douligieris. Fairness in network optimal flow control. In *SBT/IEEE International Symposium on Telecommunications*, pages 590–596, 1990.

[72] Frank Kelly. Charging and rate control for elastic traffic. *European transactions on Telecommunications*, 8(1):33–37, 1997.

[73] Tian Lan and Mung Chiang. An axiomatic theory of fairness in resource allocation. *George Washington University*, 2011.

[74] Jonggyu Jang and Hyun Jong Yang. α-fairness-maximizing user association in energy-constrained small cell networks. *IEEE Transactions on Wireless Communications*, 21(9):7443–7459, 2022.

[75] Ruijie Du, Deepan Muthirayan, Pramod P Khargonekar, and Yanning Shen. Long-term fairness for real-time decision making: A constrained online optimization approach. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[76] Wei Li, Shengling Wang, Yong Cui, Xiuzhen Cheng, Ran Xin, Mznah A Al-Rodhaan, and Abdullah Al-Dhelaan. Ap association for proportional fairness in multirate wlans. *IEEE/ACM Transactions On Networking*, 2013.

[77] Jiayi Li, Matthew Motoki, and Baosen Zhang. Balancing fairness and efficiency in energy resource allocations. *arXiv preprint arXiv:2403.15616*, 2024.

[78] Shahin Jabbari, Matthew Joseph, Michael Kearns, Jamie Morgenstern, and Aaron Roth. Fairness in reinforcement learning. In *International conference on machine learning*, pages 1617–1626. PMLR, 2017.

[79] Tariq Mahmood, Reza Shahbazian, and Irina Trubitsyna. Fairness-driven explainable learning in multi-agent reinforcement learning. 2024.

[80] Jiechuan Jiang and Zongqing Lu. Learning fairness in multi-agent systems. *Advances in Neural Information Processing Systems*, 32, 2019.

[81] Chi Harold Liu, Zheyu Chen, Jian Tang, Jie Xu, and Chengzhe Piao. Energy-efficient uav control for effective and fair communication coverage: A deep reinforcement learning approach. *IEEE Journal on Selected Areas in Communications*, 36(9):2059–2070, 2018.

[82] Weiwen Liu, Feng Liu, Ruiming Tang, Ben Liao, Guangyong Chen, and Pheng Ann Heng. Balancing accuracy and fairness for interactive recommendation with reinforcement learning. *arXiv preprint arXiv:2106.13386*, 2021.

[83] Hao Hao, Wei Ding, and Wei Zhang. Time-continuous computing offloading algorithm with user fairness guarantee. *Journal of Network and Computer Applications*, 223:103826, 2024.

[84] Gabriele La Malfa, Jie M Zhang, Michael Luck, and Elizabeth Black. Fairness aware reinforcement learning via proximal policy optimization. *arXiv preprint arXiv:2502.03953*, 2025.

[85] Pratik Gajane, Akrati Saxena, Maryam Tavakol, George Fletcher, and Mykola Pechenizkiy. Survey on fair reinforcement learning: Theory and practice. *arXiv preprint arXiv:2205.10032*, 2022.

[86] Violet Xinying Chen and John N Hooker. A guide to formulating fairness in an optimization model. *Annals of Operations Research*, 326(1):581–619, 2023.

[87] Alexandra Cimpean, Pieter Libin, Youri Coppens, Catholijn Jonker, and Ann Nowé. Towards fairness in reinforcement learning. In *Proceedings of the Adaptive and Learning Agents Workshop (ALA 2023)*, pages 1–5, 2023.

[88] Jingdi Chen, Yimeng Wang, and Tian Lan. Bringing fairness to actor-critic reinforcement learning for network utility optimization. In *INFOCOM 2021*, 2021.

[89] Guanbao Yu, Umer Siddique, and Paul Weng. Fair deep reinforcement learning with preferential treatment. In *ECAI 2023*. IOS Press, 2023.

[90] Umer Siddique, Paul Weng, and Matthieu Zimmer. Learning fair policies in multi-objective (deep) reinforcement learning with average and discounted rewards. In *International Conference on Machine Learning*, 2020.

[91] Peizhong Ju, Arnob Ghosh, and Ness B Shroff. Achieving fairness in multi-agent markov decision processes using reinforcement learning. *arXiv preprint arXiv:2306.00324*, 2023.

[92] Eric Yang Yu, Zhizhen Qin, Min Kyung Lee, and Sicun Gao. Policy optimization with advantage regularization for long-term fairness in decision systems. *arXiv preprint arXiv:2210.12546*, 2022.

[93] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[94] David Silver, Satinder Singh, Doina Precup, and Richard S Sutton. Reward is enough. *Artificial intelligence*, 299:103535, 2021.

[95] Peter Vamplew, Benjamin J Smith, Johan Källström, Gabriel Ramos, Roxana Rădulescu, Diederik M Roijers, Conor F Hayes, Fredrik Heintz, Patrick Mannion, Pieter JK Libin, et al. Scalar reward is not enough: A response to silver, singh, precup and sutton (2021). *Autonomous Agents and Multi-Agent Systems*, 36(2):41, 2022.

[96] T. Bu, L. Li, and R. Ramjee. Generalized proportional fair scheduling in third generation wireless data networks. In *INFOCOM 2006*.

[97] Paul Almasan, José Suárez-Varela, Arnau Badia-Sampera, Krzysztof Rusek, Pere Barlet-Ros, and Albert Cabellos-Aparicio. Deep reinforcement learning meets graph neural networks: Exploring a routing optimization use case. *arXiv preprint arXiv:1910.07421*, 2019.

[98] Hado Hasselt. Double q-learning. *Advances in neural information processing systems*, 23, 2010.

[99] Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems*, 2017.

[100] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 2008.

[101] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *5th ICLR*, 2017.

[102] Fernando Gama, Antonio G. Marques, Geert Leus, and Alejandro Ribeiro. Convolutional neural network architectures for signals supported on graphs. *IEEE Transactions on Signal Processing*, 2019.

[103] Elvin Isufi, Fernando Gama, and Alejandro Ribeiro. Edgenets:edge varying graph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[104] Fernando Gama, Antonio G. Marques, Geert Leus, and Alejandro Ribeiro. Convolutional neural network architectures for signals supported on graphs. *IEEE Transactions on Signal Processing*, 2019.

[105] Hongseok Kim, Gustavo De Veciana, Xiangying Yang, and Muthaiah Venkatachalam. Distributed α-optimal user association and cell load balancing in wireless networks. *IEEE/ACM Transactions on Networking*, 2011.

[106] Hisham Elshaer, Mandar N Kulkarni, Federico Boccardi, Jeffrey G Andrews, and Mischa Dohler. Downlink and uplink cell association with traditional macrocells and millimeter wave small cells. *IEEE Transactions on Wireless Communications*, 2016.

[107] Xin Ge, Xiuhua Li, Hu Jin, Julian Cheng, and Victor CM Leung. Joint user association and user scheduling for load balancing in heterogeneous networks. *IEEE Transactions on Wireless Communications*, 2018.

[108] Ning Wang, Ekram Hossain, and Vijay K Bhargava. Joint downlink cell association and bandwidth allocation for wireless backhauling in two-tier hetnets with large-scale antenna arrays. *IEEE Transactions on Wireless Communications*, 2016.

[109] Nan Zhao, Ying-Chang Liang, Dusit Niyato, Yiyang Pei, Minghu Wu, and Yunhao Jiang. Deep reinforcement learning for user association and resource allocation in heterogeneous cellular networks. *IEEE Transactions on Wireless Communications*, 2019.

[110] Yalin Zhang, Liang Xiong, and Jia Yu. Deep learning based user association in heterogeneous wireless networks. *IEEE Access*, 2020.

[111] Jingjing Cui, Yuanwei Liu, and Arumugam Nallanathan. Multi-agent reinforcement learning-based resource allocation for uav networks. *IEEE Transactions on Wireless Communications*, 2020.

[112] Mohammad Mozaffari, Walid Saad, Mehdi Bennis, Young-Han Nam, and Mérouane Debbah. A tutorial on uavs for wireless networks: Applications, challenges, and open problems. *IEEE Communications Surveys 'I&' Tutorials*, 2019.

[113] Qianqian Zhang, Mohammad Mozaffari, Walid Saad, Mehdi Bennis, and Merouane Debbah. Machine learning for predictive on-demand deployment of uavs for wireless communications. In *GLOBECOM*, 2018.

[114] Ursula Challita, Walid Saad, and Christian Bettstetter. Interference management for cellular-connected uavs: A deep reinforcement learning approach. *IEEE Transactions on Wireless Communications*, 2019.

[115] Pei Li, Lingyi Wang, Wei Wu, Fuhui Zhou, Baoyun Wang, and Qihui Wu. Graph neural network-based scheduling for multi-uav-enabled communications in d2d networks. *Digital Communications and Networks*, 2022.

[116] Samira Hayat, Evşen Yanmaz, and Raheeb Muzaffar. Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint. *IEEE Communications Surveys 'I&' Tutorials*, 2016.

[117] Hazim Shakhatreh, Ahmad H. Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah,

and Mohsen Guizani. Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges. *IEEE Access*, 2019.

[118] V. Dewanto and M. Gallagher. Examining average and discounted reward optimality criteria in reinforcement learning. In *AI 2022: Advances in Artificial Intelligence*. Springer International Publishing, 2022.

[119] Arjun Anand, Gustavo De Veciana, and Sanjay Shakkottai. Joint scheduling of urllc and embb traffic in 5g wireless networks. *IEEE/ACM Transactions On Networking*, 28(2):477–490, 2020.

[120] Luana Ruiz, Fernando Gama, and Alejandro Ribeiro. Graph neural networks: Architectures, stability, and transferability. *Proceedings of the IEEE*, 109(5):660–682, 2021.

[121] Gely P. Basharin, Amy N. Langville, and Valeriy A. Naumov. The life and work of a. a. markov.

[122] Charles Miller Grinstead and James Laurie Snell. *Introduction to probability*. American Mathematical Soc., 2012.

[123] F. Bonomi. On job assignment for a parallel system of processor sharing queues. *IEEE Transactions on Computers*, 1990.

[124] 3rd Generation Partnership Project (3GPP). NR; Overall description; Stage-2 (Release 15). Technical Specification TS 38.300, 3GPP, 2018. Version 15.2.0, available at https://www.3gpp.org/ftp/Specs/archive/38_series/38.300/.

[125] Jasneet Kaur, M. Arif Khan, Mohsin Iftikhar, Muhammad Imran, and Qazi Emad Ul Haq. Machine learning techniques for 5g and beyond. *IEEE Access*, 9:23472–23488, 2021.

[126] Manuel Eugenio Morocho-Cayamcela, Haeyoung Lee, and Wansu Lim. Machine learning for 5g/b5g mobile and wireless communications: Potential, limitations, and future directions. *IEEE Access*, 7:137184–137206, 2019.

[127] Aldebaro Klautau, Pedro Batista, Nuria González-Prelcic, Yuyang Wang, and Robert W Heath. 5g mimo data for machine learning: Application to beam-selection using deep learning. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–9. IEEE, 2018.

[128] Hongji Huang, Song Guo, Guan Gui, Zhen Yang, Jianhua Zhang, Hikmet Sari, and Fumiyuki Adachi. Deep learning for physical-layer 5g wireless techniques: Opportunities, challenges and solutions. *IEEE Wireless Communications*, 27(1):214–222, 2019.

[129] Moh Khalid Hasan, Md Shahjalal, Md Mainul Islam, Md Morshed Alam, Md Faisal Ahmed, and Yeong Min Jang. The role of deep learning in noma for 5g and beyond communications. In *2020 International Conference on*

Artificial Intelligence in Information and Communication (ICAIIC), pages 303–307. IEEE, 2020.

[130] Jiabin Li, Ming Liu, Zhi Xue, Xiaochen Fan, and Xiangjian He. Rtvd: A real-time volumetric detection scheme for ddos in the internet of things. *IEEE Access*, 8:36191–36201, 2020.

[131] Ying Wang, Peilong Li, Lei Jiao, Zhou Su, Nan Cheng, Xuemin Sherman Shen, and Ping Zhang. A data-driven architecture for personalized qoe management in 5g wireless networks. *IEEE Wireless Communications*, 24(1):102–110, 2016.

[132] Angel Martin, Jon Egaña, Julián Flórez, Jon Montalbán, Igor G. Olaizola, Marco Quartulli, Roberto Viola, and Mikel Zorrilla. Network resource allocation system for qoe-aware delivery of media services in 5g networks. *IEEE Transactions on Broadcasting*, 64(2):561–574, 2018.

[133] Hyoungju Ji, Sunho Park, Jeongho Yeo, Younsun Kim, Juho Lee, and Byonghyo Shim. Ultra-reliable and low-latency communications in 5g downlink: Physical layer aspects. *IEEE Wireless Communications*, 25(3):124–130, 2018.

[134] 3GPP. Etsi tr 138 912 5g; study on new radio (nr) access technology. *ETSI*, version 15.0.0(Release 15), 2018.

[135] Petar Popovski, Kasper Fløe Trillingsgaard, Osvaldo Simeone, and Giuseppe Durisi. 5g wireless network slicing for embb, urllc, and mmtc: A communication-theoretic view. *IEEE Access*, 6:55765–55779, 2018.

[136] K. Pedersen, G. Pocovi, J. Steiner, and A. Maeder. Agile 5g scheduler for improved e2e performance and flexibility for different network implementations. *IEEE Communications Magazine*, 56(3):477–490, 2018.

[137] Klaus Pedersen, Guillermo Pocovi, Jens Steiner, and Andreas Maeder. Agile 5g scheduler for improved e2e performance and flexibility for different network implementations. *IEEE Communications Magazine*, 56(3):210–217, 2018.

[138] Guillermo Pocovi, Klaus I. Pedersen, and Preben Mogensen. Joint link adaptation and scheduling for 5g ultra-reliable low-latency communications. *IEEE Access*, 6:28912–28922, 2018.

[139] Ali A. Esswie and Klaus I. Pedersen. Multi-user preemptive scheduling for critical low latency communications in 5g networks. In *2018 IEEE Symposium on Computers and Communications (ISCC)*, pages 00136–00141, 2018.

[140] Anupam Kumar Bairagi, Md. Shirajum Munir, Madyan Alsenwi, Nguyen H. Tran, and Choong Seon Hong. A matching based coexistence mechanism between embb and urllc in 5g wireless networks. In *Proceedings of*

the 34th ACM/SIGAPP Symposium on Applied Computing, SAC '19, page 2377–2384, New York, NY, USA, 2019. Association for Computing Machinery.

[141] Madyan Alsenwi, Nguyen H. Tran, Mehdi Bennis, Shashi Raj Pandey, Anupam Kumar Bairagi, and Choong Seon Hong. Intelligent resource slicing for embb and urllc coexistence in 5g and beyond: A deep reinforcement learning based approach. *IEEE Transactions on Wireless Communications*, 20(7):4585–4600, 2021.

[142] Medhat Elsayed and Melike Erol-Kantarci. Ai-enabled radio resource allocation in 5g for urllc and embb users. In *2019 IEEE 2nd 5G World Forum (5GWF)*, pages 590–595, 2019.

[143] Mohammed Almekhlafi, Mohamed Amine Arfaoui, Mohamed Elhattab, Chadi Assi, and Ali Ghrayeb. Joint resource allocation and phase shift optimization for ris-aided embb/urllc traffic multiplexing. *IEEE Transactions on Communications*, 70(2):1304–1319, 2022.

[144] Hiroyuki Otsuka, Ruxiao Tian, and Koki Senda. Transmission performance of an ofdm-based higher-order modulation scheme in multipath fading channels. *Journal of Sensor and Actuator Networks*, 8:19, 03 2019.

[145] Pierre Carpentier, Jean-Philippe Chancelier, Michel de Lara, and Tristan Rigaut. Algorithms for two-time scales stochastic optimization with applications to long term management of energy storage. February 2019. working paper or preprint.

[146] Yongjun Xu, Guan Gui, Haris Gacanin, and Fumiyuki Adachi. A survey on resource allocation for 5g heterogeneous networks: Current research, future trends, and challenges. *IEEE Communications Surveys 'I&' Tutorials*, 23(2):668–695, 2021.

[147] Deyu Zhang, Ying Qiao, Liang She, Ruyin Shen, Ju Ren, and Yaoxue Zhang. Two time-scale resource management for green internet of things networks. *IEEE Internet of Things Journal*, 6(1):545–556, 2019.

[148] Tianrui Chen, Xinruo Zhang, Minglei You, Gan Zheng, and Sangarapillai Lambotharan. A gnn-based supervised learning framework for resource allocation in wireless iot networks. *IEEE Internet of Things Journal*, 9(3):1712–1724, 2022.

[149] Zhejing Bao, Qin Zhou, Zhihui Yang, Qiang Yang, Lizhong Xu, and Ting Wu. A multi time-scale and multi energy-type coordinated microgrid scheduling solution—part i: Model and methodology. *IEEE Transactions on Power Systems*, 30(5):2257–2266, 2015.

[150] Zhaoxi Liu, Qiuwei Wu, Kang Ma, Mohammad Shahidehpour, Yusheng Xue, and Shaojun Huang. Two-stage optimal scheduling of electric vehicle

charging based on transactive control. *IEEE Transactions on Smart Grid*, 10(3):2948–2958, 2019.

[151] Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: A methodological tour d'horizon. *European Journal of Operational Research*, 290(2):405–421, 2021.

[152] Mark Eisen, Clark Zhang, Luiz FO Chamon, Daniel D Lee, and Alejandro Ribeiro. Learning optimal resource allocations in wireless systems. *IEEE Transactions on Signal Processing*, 2019.

[153] J. Aitchison. The statistical analysis of compositional data. *Journal of the Royal Statistical Society. Series B (Methodological)*, 44(2):139–177, 1982.

[154] Qinqing Zhang and S.A. Kassam. Finite-state markov model for rayleigh fading channels. *IEEE Transactions on Communications*, 47(11):1688–1692, 1999.

[155] S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83), 2016.

[156] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[157] François Chollet et al. Keras. `https://keras.io`, 2015.

This page has been intentionally left blank.

Index of tables

This page has been intentionally left blank.

Index of figures

Index of figures

Index of figures

This is the last page.
Compile on Wednesday 29th April, 2026.
http://iie.fing.edu.uy/