Connectionless transmission in wireless networks (IoT)

Iman Hmedoush

Sorbonne University

École Doctorale Informatique, Télécommunications et Électronique de Paris

@

Inria - Paris

@

Thesis Report

TO OBTAIN THE DEGREE OF DOCTEUR D'UNIVERSITÉ

Subject

CONNECTIONLESS TRANSMISSION IN WIRELESS NETWORKS (IoT)

PRESENTED BY

IMAN HMEDOUSH

Thesis Advisor : Paul Mühlethaler
Thesis Co-advisor : Cédric Adjih
Host teams : EVA-INRIA Paris

Jury

Marceau Coupechoux - Reviewer-Professor, Telecom Paris, École Polytechnique
Isabelle Guérin-Lassous - Reviewer-Professor, Université Claude Bernard, Lyon1
Bartek Blaszcyzyn - Examiner - Research Director, INRIA-Paris
Salah Eddine Elayoubi - Examiner - Professor-(L2S), Centrale Supelec, France
Kinda Khawam - Examiner - Associate Professor - Université de Versailles St-Quentin

Acknowledgments

First, I would like to thank Inria-Nokia Bell Labs common lab under research action "Network Information Theory" for providing me with the financial means to complete this thesis.

I would like to express my deepest gratitude to my advisor, Paul Mühlethaler, for his excellent guidance and his continuous support of my Ph.D. study.

I would like to express my sincere gratitude to my co-supervisor, Cédric Adjih, for his patience, guidance, and kind support. I have benefited greatly from his wealth of knowledge and meticulous editing. I am extremely grateful that I have done my thesis under his painstaking supervision.

I also would like to thank Marceau Coupechoux and Isabelle Guerin-Lassous for kindly accepting to review my thesis. Also, my thanks go to the members of the jury for the interest they showed in my research: Bartek Blaszczyzyn, Salah Eddine Elayoubi and Kinda Khawam.

I would like to thank all the co-authors of my publications during this thesis. I dedicate my deepest gratitude to Vinod Kumar, Chung Shue (Calvin) Chen, Kinda Khawam, Samer Lahoud, Lou Salaün, Charles Dumas, and Ibrahim Ayoub.

My special thoughts and my deepest gratitude to my greatest supporter, my mother, my father and my two brothers.

I would deeply thank all my friends who were strong supporters of my Ph.D.

Abstract

The origin of the idea of adding intelligence to basic objects and making them communicate has been lost to history. But in recent times, the emergence of the Internet as a global communication network has also motived the use of its architecture and protocols to connect objects (such as the soda vending machine famously connected to the ARPANET in the 1980s).

In the past two decades, many technological enhancements have been developed to enable the "Internet of Things" (IoT). A scenario of a typical IoT network is to connect embedded devices composed of environmental sensors, microcontrollers, and communication hardware, to a central collection node. The set of data gathered by these nodes will increasingly help in analyzing and precisely understanding the phenomenons and behaviors occurring in this environment. The applications of IoT technologies are endless because they are adaptable to almost any system that can provide information about its status, operation, and the environment and that one needs to monitor and control at a distance. Smart cities, healthcare, industrial automation, and wearable technology are some IoT applications that promise to make our life safer and easier.

Some research and technology challenges need to be addressed for the implementation and full popularization of IoT applications including deployment, networking, security, resilience, and power control. This massive demand for connection in IoT networks will introduce new challenges in terms of connectivity, reliability, and technology. At the radio network level, IoT networks represent a huge inflow of various devices that communicate through the same shared radio medium. However, many of these devices are difficult to secure and handle. One major challenge to deploying IoT networks is the lack of efficient solutions that allow for a massive number of connections while meeting the low-latency and low-cost demands at the same time.

In addition, recently, there has been a trend towards long-range communications systems for the IoT, including cellular networks. For many use cases, such as massive machine-type communications (mMTC), performance can be gained by moving away from the classical model of connection establishment and adopting grant-free, random access methods. Associated with physical layer techniques such as Successive Interference Cancellation (SIC), or Non-Orthogonal Multiple Access (NOMA), the performance of random access can be dramatically improved, giving rise to novel random access protocol designs.

In this thesis, we focus on one of the modern candidates for random access protocols "well-fitted" to the IoT: Irregular Repetition Slotted ALOHA (IRSA) [1]. As solutions are needed to overcome the challenges of IoT, we study the IRSA random access scheme from

new points of view and we start with an analysis of the performance of different variations through the density evolution tool. Precisely, we start by revisiting the scenario of the IRSA protocol in the case of Multiple Packet Reception (MPR) capability at the receiver. Then, we study IRSA in different scenarios where more realistic assumptions are considered, such as IRSA with multiple transmissions powers, with capture effect, and with decoding errors. In the second part of the thesis, we concentrate on learning and dynamically adjusting IRSA protocol parameters. First, we analyze the protocol performance in a centralized approach through a variant of Reinforcement Learning and in a distributed approach through Game Theory. We also optimize short frame length IRSA through a Deep Reinforcement Learning approach. Finally, we introduce a sensing capability to IRSA, in line with carrier sense principles, and we tentatively explore how one can learn part of sensing protocols with the help of Deep Learning tools.

Keywords: Random Access, mMTC, IoT, Coded Slotted Aloha (CSA), Irregular Repetition Slotted Aloha (IRSA), Successive Interference Cancellation (SIC), Density Evolution (DE), Machine Learning (ML).

Résumé de Thèse

L'origine concernant l'idée d'ajouter de l'intelligence aux objets de base et de les faire communiquer n'est pas connue précisément. Mais ces derniers temps, l'émergence d'Internet en tant que réseau de communication global a aussi motivé l'utilisation de son architecture et de ses protocoles pour connecter des objets. C'est par exemple le cas célèbre du distributeur automatique de sodas connecté à l'ARPANET dans les années 1980.

Au cours des deux dernières décennies, de nombreuses améliorations technologiques ont été développées pour rendre possible l'Internet des objets (IoT). Un scénario d'un réseau IoT typique consiste à connecter des dispositifs embarqués composés de capteurs environnementaux, de microcontrôleurs et de matériel de communication à un nœud de collecte central. L'ensemble des données recueillies par ces nœuds permettra d'analyser et de comprendre précisément les phénomènes et comportements se produisant dans cet environnement. Les applications des technologies IoT sont infinies, car elles sont adaptables à presque tous les systèmes, que l'on doit surveiller et contrôler à distance, pouvant fournir des informations sur son état, son fonctionnement et son environnement. Les villes intelligentes, les soins, l'automatisation industrielle et la technologie portable sont quelques-unes des applications de l'IoT qui promettent de rendre notre vie plus sûre et plus facile.

Certains défis en matière de recherche et de technologie doivent être relevés pour la mise en œuvre et la large dissémination des applications de l'IoT comme le déploiement, la mise en réseau, la sécurité, la résilience et le contrôle de l'alimentation des équipements. Cette demande massive de connexion dans les réseaux IoT introduit de nouveaux défis en termes de connectivité, de fiabilité et de technologie. Au niveau de la radio, les réseaux IoT représentent un énorme afflux de divers appareils qui communiquent via le même support radio partagé. Cependant, bon nombre de ces appareils sont difficiles à sécuriser et à manipuler. L'un des principaux défis du déploiement des réseaux IoT est le manque de solutions efficaces qui permettent un nombre massif de connexions tout en répondant en même temps aux exigences de faible latence et de faible coût.

De plus, il y a eu récemment une tendance vers des systèmes de communication à longue portée pour l'IoT et aussi pour les réseaux cellulaires. Pour de nombreux cas d'utilisation, tels que les communications massives de type machine (mMTC), les performances peuvent être améliorées en s'éloignant du modèle classique d'établissement de connexion et en adoptant des méthodes d'accès aléatoire sans attribution prédéterminée. Associé à des techniques de couche physique telles que l'annulation successive des interférences (SIC) ou l'accès multiple non orthogonal (NOMA), les performances de l'accès aléatoire peuvent être améliorées, donnant lieu à de nouvelles conceptions de protocoles

d'accès aléatoire.

Dans cette thèse, nous nous concentrons sur l'un des candidats modernes pour les protocoles d'accès aléatoire bien adaptés à l'IoT : ALOHA à répétition irrégulière (IRSA) [1]. Comme des solutions sont nécessaires pour surmonter les défis de l'IoT, nous étudions le schéma d'accès aléatoire IRSA sous de nouveaux points de vue et nous commençons par une analyse des performances des différentes variantes grâce à l'outil de l'évolution de la densité du débit. Précisément, nous commençons par revisiter le scénario du protocole IRSA avec la capacité de réception de paquets multiples (MPR) au niveau du récepteur. Ensuite, nous étudions IRSA dans différents scénarios où des hypothèses plus réalistes sont considérées comme : IRSA avec plusieurs puissances de transmission, avec effet de capture et avec des erreurs de décodage. Dans la deuxième partie de la thèse, nous nous concentrons sur l'apprentissage et l'ajustement dynamique des paramètres du protocole IRSA. Dans un premier temps, nous analysons les performances du protocole dans une approche centralisée via une variante de l'Apprentissage par Renforcement et dans une approche distribuée via la Théorie des Jeux. Nous optimisons également IRSA avec une courte longueur de trame grâce à une approche d'apprentissage par renforcement profond. Enfin, nous introduisons pour IRSA une capacité de détection, fonctionnant suivant les principes de la détection de porteuse, et nous explorons comment peut-on apprendre une partie des protocoles de détection à l'aide d'outils d'apprentissage profond.

Mots clés : accès aléatoire, mMTC, IoT, Coded Slotted Aloha (CSA), Irregular Repetition Slotted Aloha (IRSA), Successive Interference Cancellation (SIC), Density Evolution (DE), Machine Learning (ML).

List of Figures

List of Tables

List of Abbreviations

3GPP	The 3rd Generation Partnership Project
5G	Fifth-Generation Technology Standard
AI	Artificial Intelligence
BS	Base Station
CDMA	Code-Division Multiple Access
CRA	Contention Resolution Algorithms
CRC	Cyclic Redundancy Check
CRDSA	Contention Resolution Diversity Slotted ALOHA
CS	Compresive Sensing
CSA	Coded Slotted ALOHA
CSMA	Carrier Sense Multiple Access
CSMA/CA	Carrier-Sense Multiple Access with Collision Avoidance
CSMA/CD	Carrier-Sense Multiple Access with Collision Detection
DE	Density Evolution
Dec-POMDP	The Decentralized Partially Observable Markov Decision Process
DL	Down Link
DNN	Deep Neural Network
DQL	Deep Q-Learning
DQN	Deep Q-Network

DRL	Deep Reinforcement Learning
DS-IRSA	Deep-Learning and Sensing-based IRSA (Chapter 9)
EC-GSM	Extended Coverage GSM
eMBB	Enhanced Mobile Broadband
FDMA	Frequency-Division Multiple Access
GSM	The Global System for Mobile Communications
HTC	Human Type Communication
IoT	Internet of Things
IRSA	Irregular Repetition Slotted ALOHA
IRSA-RM	IRSA based on Regret Minimization (Chapter 6)
ITU	International Telecommunication Union
K-IRSA	IRSA with K-MPR (Chapter 3)
K-MPR	K-Multiple Packet Reception
LAN	Local Area Network
LDPC	Low-Density Parity-Check code
LPWAN	Low Power Wide Area Networks
LTE	Long-Term Evolution
LTE-M	Long-Term Evolution for MTC
M2M	Machine-to-Machine
MA	Multiple Access
MAB	Multi-Armed Bandit
MAC	Medium Access Control
MARL	Multi-Agent Reinforcement Learning
MDP	Markov Decision Process
MIMO	Multiple-Input Multiple-Output
ML	Machine Learning

mMTC	massive Machine Type Communication
MP-IRSA	Multi-Power IRSA (Chapter 4)
MPR	Multiple Packet Reception
MTC	Machine Type Communication
MUD	Multi-User Detection
NB-IoT	Narrow-Band IoT
NE	Nash Equilibrium
NOMA	Non Orthogonal Multiple Access
OFDMA	Orthogonal Frequency Divion Multiple Access
OMA	Orthogonal Multiple Access
OP	Optimization Problem
PDMA	Power Division Multiple Access
PER	Packet Error Rate
PHY	Physical Layer
PLR	Packet Loss Rate
POMDP	Partially Observable Markov Decision Process
PPO	Proximal Policy Optimization
QoS	Quality of Service
RA	Random Access
RACH	Random-Access Channel
RC-IRSA	IRSA with Random Codeword Selection (Chapter 8)
RL	Reinforcement Learning
RM	Regret Minimization
RTT	Round Time Trip
S-IRSA	Sensing-based IRSA (Chapter 9)
SA	Slotted ALOHA

SAA	Sample Average Approximation
SC	Spatial Coupling
SIC	Successive Interference Cancellation
SIR	Signal to Interference Ratio
SL	Supervised Learning
TDMA	Time-Division Multiple Access
UD	Uniquely Decipherable codes
UL	Unsupervised Learning
UL	Up link
URLLC	Ultra Reliable Low Latency Communications
WFIP	Weak Finite Improvement Property
WLAN	Wireless Local Area Network
WSN	Wireless Sensor Network
ZFD	Zero-False Drop code

Contents

Chapter 1

Introduction

1.1 Background and Motivations

The Internet of Things (IoT) is a system of interrelated devices connected to the Internet to transfer data among each other and with servers. It applies to more than just sensors or devices: it focuses on entire use cases. Smart buildings, connected cars, and industrial automata are examples of applications, where things need to "talk to each other", through complex interactions. Fig. 1.1 shows some of the existing IoT applications. The IoT can also exploit interactions between the user and the environment from one side and between the components of the environment from the other side. We expect to see huge growth in the number of IoT-connected devices to reach hundreds of billions during the next few years. Making this smart network effective requires that the necessary technologies are designed properly to meet its requirements and overcome its challenges. One of these key IoT technologies is communications.

Indeed, driven by the necessity and the prominence of IoT applications in everyday life, massive Machine Type Communication (mMTC) has emerged as a fundamental part of future communications. mMTC is a fundamental use case of IoT, where a massive number of devices transmit short packets in a sporadic way to the base station (BS). The sporadic traffic pattern and the small size of transmitted packets are the new features of mMTC that make the existing connection protocols and technologies insufficient to support the massive connectivity. The inevitable need for new connection technologies for mMTC has triggered over the recent years a lot of interest, leading to countless new research directions to design efficient new protocols for mMTC applications. In turn, the need for such mMTC -supported protocols has led to the development of very sophisticated PHY and MAC layer technologies that can better exploit the wireless channel.

ALOHA-based solutions have also arisen as a promising candidate for mMTC applications. These solutions provide a competitive performance compared to that of their coordinated counterparts in terms of throughput and reliability, leading the way for the application of modern RA protocols to many IoT scenarios of 5G and beyond [3]. The main idea of these protocols is to allow transmitters to send multiple copies of their packets towards the base station (BS). At the receiver, Successive Interference Cancellation (SIC) is applied to resolve collisions. It is possible to combine SIC with other advanced physical layer techniques to recover the packets. Despite that the research behind the applications of modern random access protocols for mMTC has shown a remarkable gain in terms of supported network load and achieved throughput, further study is necessary to understand their true potential in IoT networks and beyond 5G.

Figure 1.1: Illustration of some IoT applications

1.2 Thesis Objectives and Contributions

Motivated by the importance of IoT networks in our future, and the insistent need to solve the communication problem for mMTC, the main objective of this thesis is to propose a communication protocol for IoT applications that increases the achieved performance, enables massive scalability, and overcomes the shortcomings of other suggested protocols. In this context, we extensively studied the problem of multiple access in IoT and the main existing solutions in the literature. We focus on *modern random access* type of solutions,

mainly a new family of protocols [4], which has been lately considered as a promising solution for the particular setting of IoT. One such modern RA is the focus of this thesis, coined Irregular Repetition Slotted Aloha (IRSA) [1], and its generalization with coding, coined Coded Slotted Aloha (CSA) [5]. It has become the focus of IoT protocol designers since it has been shown to asymptotically reach the optimal throughput of one retrieved packet per slot [6], in the classical random access collision model (where the maximum throughput of slotted ALOHA is $\frac{1}{e}$). Despite that IRSA has been proposed in 2011 [1], there is a substantial literature about its possible promised application for IoT networks. Additionally, many IRSA variants have been proposed to enhance its performance, making sometimes a good use of the advanced physical layer techniques.

The first contribution of this thesis, presented in Chapter 3, studies the IRSA variant with K multiple packet reception (K-MPR), called K-IRSA. We extensively revisit the case of IRSA with MPR. One of our major results is the proof that K-IRSA cannot reach the natural bound of throughput of K retrieved packets per slot, and we prove a new, lower bound for its performance. We also give a simple expression for its excellent loss rate at lower loads. The results also hint that the method for finding the optimal parameters (soliton distribution families [6]) for IRSA (1-IRSA), might not work for K-IRSA with $K > 1$. Finally, we show how to compute numerically optimal parameters for K-IRSA.

The second contribution of this thesis is presented in Chapter 4. We focus on the scenario of an IoT network where the packets of different nodes are received with different powers at the base station, either per design due to different transmission powers, or induced by the fact that the nodes are at different distances from the base station. In such a scenario, the capture effect emerges at the receiver, which in turn enhances the IRSA performance. We propose a version of IRSA called Multi-Power IRSA, and develop the needed mathematical tools to evaluate it. We analyze the protocol behavior, explore the achievable throughput and its associated gain, and show the excellent performance of Multi-Power IRSA.

In Chapter 5, we study a novel extension of IRSA, taking into account realistic effects, namely errors due to an imperfect SIC process. The main contribution of this chapter is to propose a method to obtain the optimal distributions (in terms of maximum load threshold, or equivalently, throughput) under the assumption of SIC errors.

Recently, the approaches based on Reinforcement Learning (RL) have appeared as promising solutions to IRSA performance issues, especially considering that optimizing IRSA parameters is not an easily solved problem. Motivated by these reasons, in Chapter 6, we use a form of centralized reinforcement learning approach for that purpose. We

adopt one specific variant of RL, Regret Minimization, to learn the protocol parameters. Our proposed variant shows the excellent performance of IRSA when it is optimized with Regret Minimization.

Then, we study the IRSA random access scheme in a distributed and competitive setting in Chapter 7. To that aim, we adopt a distributed game-theoretic approach where two classes of IoT devices learn autonomously their optimal IRSA protocol parameters to optimize selfishly their own effective throughput. The main contribution of this chapter is to show that our IRSA game attains the Nash equilibrium (NE) via the "better reply" strategy in some general conditions.

Motivated by the need to optimize IRSA for short frame length, and the shift to an ML-based protocol design, we propose in Chapter 8, a Deep Reinforcement Learning (DRL) approach to optimize IRSA for small frame sizes. Our DRL approach is close to the optimal throughput that we calculated through Differential Evolution (DE). The proposed DRL approach in this chapter is a base for the proposed IRSA variant in the next chapter.

The final contribution of this thesis is in Chapter 9. In this chapter, we propose DS-IRSA, a Deep Learning Sensing-based IRSA variant, where we add a sensing phase before IRSA transmission. The goal of introducing the sensing for IRSA is to learn how to best interact to synchronize the nodes in the transmission phase to avoid collisions. Our proposed protocol has been shown to achieve higher throughput than classical IRSA protocol or any other Deep Learning-based IRSA variant.

1.3 Thesis Outline

The thesis is organized as follows:

Chapter 2 presents the background and an overview of the existing work performed in the content of mMTC and IoT needs, specifically the communication problem. It also highlights and overviews the fundamental properties of CSA protocols, in particular IRSA.

Chapter 3 revisits a new variant of IRSA, "K-IRSA", which combines the design of IRSA with the capability of K multiple packet reception at the receiver. The main goal is to find optimal degree distributions for K-IRSA and to find some bounds on its performance.

Chapter 4 studies the scenario of an IoT network with capture effect at the receiver. Under this assumption, it presents an analysis of IRSA behavior using a new density evolution (a mathematical tool used in IRSA), which is based on dividing nodes into classes with different powers.

In Chapter 5, we study the design of IRSA protocol, accounting for a non-zero proba-

bility of error due to imperfect Interference Cancellation (IC). We adapt and optimize the density evolution to analyze IRSA under this assumption.

Chapter 6 proposes a new variant of IRSA protocol, "IRSA-RM", which uses a reinforcement learning approach, specifically, Regret Minimization to learn the protocol parameters. We explain why it is selected, and how to apply it to our problem with centralized offline learning.

In Chapter 7, we study IRSA in a distributed setting. We use a distributed game-theoretic approach to optimize the protocol parameters. The distributed approach is modeled as a non-cooperative game where the two classes compete and optimize autonomously and selfishly their own throughput.

Chapter 8 presents a DRL approach to optimize IRSA for short frame length. We formalize the problem of finding an optimal codebook for IRSA as an optimization problem. We also point out the difficulties of solving mathematically or analytically this problem for IRSA with ≥ 10 users. Then, we justify through simulations that our DRL approach is an excellent solution to optimize IRSA with short frame length.

The main idea of Chapter 9 is to introduce sensing capability to the nodes, aiming to find some form of coordination between them. A deep reinforcement learning approach is used to optimize the transmission strategy of IRSA using the collected information through the sensing phase.

Finally, Chapter 10 concludes the thesis and proposes some future perspectives.

1.4 Thesis Publications

Journal Papers

- Charles Dumas, Lou Salaün, **Iman Hmedoush**, Cédric Adjih and Chung Shue Chen, *Design of Coded Slotted ALOHA with Interference Cancellation Errors*, in IEEE Transactions on Vehicular Technology, vol. 70, no. 12, pp. 12742-12757, Dec. 2021.
- **Iman Hmedoush**, Cédric Adjih, Kinda Khawam and Paul Mühlethaler, *A Game Theoretic Approach to Irregular Repetition Slotted Aloha*, in IEEE Access, vol. 10, pp. 4600-4614, Jan. 2022.

Conference Papers

- **Iman Hmedoush**, Cédric Adjih, Paul Mühlethaler and Vinod Kumar *On the Performance of Irregular Repetition Slotted Aloha with Multiple Packet Reception*, 2020 International Wireless Communications and Mobile Computing (IWCMC), Jun. 2020, pp. 557-564.
- **Iman Hmedoush**, Cédric Adjih, Paul Mühlethaler and Lou Salaün, *Multi-Power Irregular Repetition Slotted ALOHA in Heterogeneous IoT networks*, 9th IFIP International Conference on Performance Evaluation and Modeling in Wireless Networks (PEMWN), Dec. 2020, pp. 1-6.
- **Iman Hmedoush**, Cédric Adjih and Paul Mühlethaler, *A Regret Minimization Approach to Frameless Irregular Repetition Slotted Aloha: IRSA-RM*, International Conference on Machine Learning for Networking, Nov. 2020, Paris / Virtual, France. hal-03043877.
- Ibrahim Ayoub, **Iman Hmedoush**, Cédric Adjih, Kinda Khawam and Samer Lahoud, *Deep-IRSA: A Deep Reinforcement Learning Approach to Irregular Repetition Slotted ALOHA* , 10th IFIP International Conference on Performance Evaluation and Modeling in Wireless and Wired Networks (PEMWN), Nov. 2021, pp. 1-6.

Chapter 2

State of the Art

2.1 IoT Overview

The Internet of Things can be defined as making every network-enabled device connected. The concept of "Things" from the network infrastructure perspective refers to any real or virtual network-connected components such as sensors, machines, or intelligent software devices. The purpose of the IoT network is to connect the maximum of devices in a given environment so that the huge set of data produced by these devices will help us to understand this environment. IoT offers a great opportunity to collect data about our surroundings, saving us time, work, and money in many applications. As a result, IoT-related technologies allow to dramatically improve existing applications and pave the way for new ones.

The applications of IoT can explain the fast development and the growing integration of IoT in our life. Healthcare, smart cities, infrastructural-related, commercial, environmental and industrial applications are a few of application domains for IoT [7]. IoT represents a vast influx of new devices from different environments which intend to connect, in addition to that many of these devices are difficult to secure and manage. Generally, this massive growth of IoT networks will introduce new challenges in terms of business, society, and technology.

One of the main challenges of IoT is related to communications. When considering cellular wireless networks, one difficulty is to ensure efficient connectivity for all devices in dense networks of massive numbers of nodes with irregular traffic demands. The traditional paradigm of resource reservation in cellular networks becomes inefficient when the traffic is infrequent, unpredictable, and limited to a few packets per node. One can think about going further and designing new physical layer and Medium Access Control (MAC) protocols

for IoT networks and alternately, many of the research and development efforts went into adapting existing cellular networks protocols in such a way that fits the new requirements of IoT as in LTE-M and NB-IoT [8]. A variety of wireless connectivity technologies to support different use cases in the IoT setting will be discussed in the next section.

2.2 IoT Communication Challenges

2.2.1 Introduction

In the past few decades, the fast technological evolution has led to the use of groundbreaking communication technologies to exchange information and connect people worldwide. Machine to Machine communication, also abbreviated as M2M is a type of communication between devices or machines without or with a little human interaction. Starting from the early 20th century, M2M has started by using wired signaling. Later on, a wealth of radio-communication technologies had been introduced during that century to connect devices, including three notable families during the 90ies: Wi-Fi [9], second-generation cellular networks, along with wireless sensor networks [10].

In particular, with the cellular world, M2M communication has moved from wired to wireless communication, leading to different cellular generations that support M2M communications. This fast development helped to connect more devices together and has made wireless connectivity universally available in wide areas, and ultimately in entire countries. Recently, there has been a trend from both academia and industry to continuously improve towards massive connectivity between machines and between humans and machines. According to Statistica [11] report, in 2025, the total number of connected devices in the world will be approximately 75.44 billion. This huge expected number of connected devices underlies the new communication paradigm that is defined now as massive Machine-Type Communications (mMTC). Current and future cellular generations are expected to support Machine-Type Communication (MTC) services along with Human-Type Communication (HTC) services. mMTC is one of the key technologies for the fifth generation of wireless communication (5G) and beyond. mMTC has been defined along with enhanced Mobile Broadband (eMBB) and Ultra-Reliable Low-Latency Communications (URLLC) by the International Telecommunication Union (ITU) and third Generation Partnership Project (3GPP) [12]. Each of these three network use cases has been defined based on the diversity of the connected devices and the needed Quality of Service (QoS). Enhanced Mobile Broadband (eMBB) is the extension of the services enabled by the 4G Long Term Evolu-

tion (LTE) networks. It is a data-driven use case where high data rates are required across a wide coverage area. eMBB usage refers to Human-Type Communications (HTC), where the number of devices is lower and the volume of exchanged data per device is large [13]. Large-scale events, public transportation, hot spots are some possible applications of eMBB to provide broadband access everywhere. URLLC service was introduced in 5G to support low latency (e.g., 1 ms) and high reliability (e.g., 99.999%) use cases. Examples include public safety, emergency applications, remote surgery, autonomous cars, and industrial automation.

On the contrary, mMTC and URLLC use cases of the IoT exhibit very different features from the HTC [13]. This remarkable shift from HTC toward mMTC is mainly driven by the fast expansion of the IoT. The Internet of Things is expected to provide communication capabilities to billions of objects. It encompasses heterogeneous types of devices, applications, requirements, and it introduces new challenges with respect to security, scalability, and connection technologies. In dense networks, the target for connection density should be $1,000,000$ devices/km^2 in urban environment [14]. In massive IoT or mMTC, the data is transmitted in small-size packets, with high energy efficiency requirements, with no or partial human intervention. The different features of the mMTC traffic have changed the design of physical/medium-access-control layers protocols. Nowadays, the massive connectivity challenge is one of the richest research topics for academia and industry. In the next section, we describe various options of wireless technologies to provide massive connectivity for mMTC applications. In [13], an advanced classification of these technologies has been proposed based on [15], which classifies IoT connection technologies based on the coverage range and QoS requirements. Short-range wireless technologies, such as WiFi, Bluetooth, and Zigbee, are examples of the technologies that support short-range connectivity with limited QoS, e.g. low data rate and security requirements typically applicable to a home or indoor environment and are not well suited to dense deployments. Meanwhile, unlicensed Low Power Wide Area Networks (LPWAN), provided by, for example, Sigfox and LoRa, have been developed for MTC applications addressing the ultra-low-end device segment, with very limited throughput and QoS requirements, but with the advantage of a longer range of communication [15].

2.2.2 Wireless Connectivity Standards for mMTC

The IoT framework includes different use cases that can be supported by different wireless connectivity standards. Each standard addresses the mMTC problem differently. In the non-cellular applications, LPWANs solutions, such as LoRaWAN and Sigfox, have

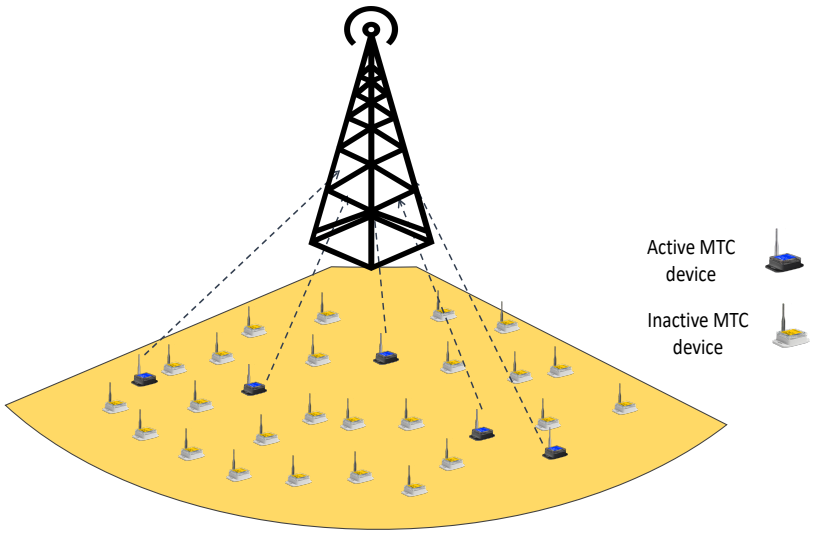


Figure 2.1: Illustration of sparse user activity in massive MTC

been popular for mMTC. However, they have low data-rates and the deployment of these technologies requires end-to-end establishment of a dedicated access network and core infrastructure, and can suffer from interference and competition in the unlicensed spectrum.

In the cellular world, 3GPP has introduced different technologies to support the requirements of emerging mMTC applications that need excellent coverage and high support for mobile users. These suggested technologies lead to a range of cellular solutions that provide low-power connectivity over a wide area. Examples of these technologies include: (a). Extended Coverage GSM (EC-GSM) which maps new data and control channels over the traditional GSM, (b). LTE for MTC (LTE-M) which is a type of LPWAN radio technology standard to provide a wide range of cellular devices and services with low power functionalities, (c). Narrow-Band IoT (NB-IoT), which is a LPWAN radio technology standard that focuses on indoor coverage, low cost, long battery life, and high connection density, and that goes further than LTE-M in the spirit of trading lower throughput and higher latency for lower energy consumption, (d). Current efforts in 3GPPP with the release-17 study item on 5G NR Light.

These technologies promise to provide an excellent coverage, high mobility support, high data rates, high reliability and low latency. The development of these technologies is carried on and is driven towards low-power IoT applications. The current challenges that face the evolution of such cellular technologies for mMTC are the need for devices of lower cost with improved battery life and better coverage. Some described standards use the traditional multiple access techniques, e.g. Code-Division Multiple Access (CDMA),

Polarization-Division Multiple Access (PDMA), modern multiple access techniques such as Non-Orthogonal Multiple Access (NOMA), modern random access protocols, etc., to provide multiple access to the wireless channel and to enhance the communications in IoT. Some used techniques will be the focus of the next section.

2.2.3 Massive Connectivity Techniques

In this section, we introduce various connectivity technologies that can be used for IoT networks. We also shed the light on the main technical challenges behind using some existing connectivity technologies for mMTC and IoT.

One of the main challenges to providing massive connectivity is the limited number of available channel resources. The present Multiple Access (MA) techniques allocate the radio resources either for one user/device as in Orthogonal MA (OMA) or for multiple users/devices as in Non-Orthogonal Multiple Access (NOMA). NOMA is considered as one of the promising technologies to enable massive connectivity as it supports sharing the same radio resource by different users/devices and can effectively increase the number of multiple access channels without any bandwidth expansion. The key idea of NOMA for massive access is to allow different signals to overlap over the same time-frequency resource via Power-Domain Multiplexing (PDM) or Code Domain Multiplexing (CDM). At the base station, Successive Interference Cancellation (SIC) can be used to perform separate decoding for each signal. Most of the studies of applying NOMA technology for massive connectivity scenarios are done from the HTC perspective [16], where a limited number of users are considered to be already connected to the base station. Consequently, existing detection algorithms, decoding strategies, and interference reduction techniques, need to be revisited, especially for ultra-dense networks, where capacity maximization is the key objective. On the other hand, optimizing what is called factor graph needs to be considered for a good trade-off between overloading factor and receiver complexity in code-domain NOMA [16].

Another important challenge is the way the device will access the radio resource. In existing wireless networks, each device reserves time slots before the transmission of its packets. The reservation is done through a contention-based random access phase, where the device sends a request for reservation.[1] In the current settings, this process is considered as one of the main deadlocks of employing current connection technologies to mMTC, as it is a source of excessive delay and signaling overhead. An alternative solution is

[1]Note that this is the principle of the Random Access Channel (RACH), which is used by wireless nodes to access the mobile network.

to use grant-free/contention-based schemes, where devices can transmit the data directly without passing through the random access grant process to reserve resources. This process can be seen as a merging between the random access phase and the data transmission. In this context, grant-free/contention-based transmission using NOMA is considered as a promising alternative [16].

Along with the inefficient radio resource allocation and usage, the bottleneck of existing technologies for the massive connectivity challenge is indeed related to the high signaling overhead. Fig. 2.1 shows the sporadic nature of mMTC traffic, as each device is active only when it has a packet to send. This feature of mMTC traffic has led to various compressive Grant-Free RA (cGFRA) schemes, as in [17] and [18]. The sparse user activity allows formulating the activity detection problem as a Compressive Sensing (CS) problem and the use of CS algorithms to solve it [16]. The principle of the cGFRA schemes is to spread the wireless signal of each device with a unique binary sequence or a sparse sequence to increase the number of supported mMTC devices. However, the performance of the cGFRA schemes degrades roughly when the number of active users is larger than the coherence interval, or the preamble sequence length.

In parallel, a new family of protocols, often labeled as modern random access [4], has been lately considered as a promising solution to provide massive connectivity for IoT. The background for these protocols is related to the idea of modifying the random access scheme of ALOHA in [19]. The modified random access scheme is known as Diversity ALOHA, where active users are allowed to send multiple copies of their packets over the shared medium. The goal is to receive at least one of these copies successfully at the destination, despite the increased channel load that can cause more collisions. The modern random access protocols adopt the same idea, but with the integration of new protocol design concepts with the usage of advanced physical layer techniques. Combining advanced system processing along with modern detection techniques (e.g., multi-user detection and Successive Interference Cancellation, SIC), can enhance the performance of RA, making it possible to support mMTC applications.

Differing from the NOMA literature, and in the tradition of random access literature, these schemes sometimes assume idealized physical layer assumptions (perfect user identification, perfect packet reception, perfect SIC, collision channel, ...). This is for several reasons: one of them is that the obtained results can be compared to the rich, existing, random access literature. Another reason is that studying accurately the full system down to every detail, such as modulation, detection, fading, SIC errors, error-correcting codes, etc. could lead to a system where the impact of each design parameter may become unclear, and

where the performance of the system might only be obtained numerically. Related to that, a precise description of the system makes the results dependent on the assumed technology, with the benefit of being more accurate, but at the expense of potentially invalidating them when one part of the system is modified (e.g. CDMA to OFDM), or when the technology naturally improves or changes. The last reason is that once results have been obtained with some idealized assumptions, they can serve as excellent benchmarks when more realistic assumptions are introduced and can offer optimistic (or pessimistic) bounds. These results can help to measure the gap between ideal and realistic assumptions, and also assess how much can be gained by closing this gap if technology is improved.

The idealized assumptions have been also applied in the context of studying modern variants of random access methods for the design of IoT protocols to satisfy the critical requirements of IoT applications. One such method is the focus of this thesis.

2.3 Literature Overview

A recent family of *modern random access* protocols [20], frequently referred to as the Coded Slotted Aloha (CSA) family, is a good candidate to address the massive connectivity issue in IoT networks. It has become the target of IoT protocol designers' attention since it has been shown to asymptotically reach the optimal throughput of one retrieved packet per slot [6] compared to the classical random access collision model where the maximum throughput is $\frac{1}{e}$ for slotted ALOHA.[2]

The main underlying principle of these protocols is to make each terminal physically send multiple copies of the same MAC packet (also called replicas), with identical preamble and payload information bits. The payload contains signaling information concerning the temporal positions of the corresponding replicas of each packet. This enables an iterative decoding process whereby successive interference cancellation (SIC) is applied to resolve collisions between different packets by physically removing the signal of a successfully decoded packet, at the position of its replicas.

2.3.1 General Family of IRSA Protocols

This operating mode has been introduced by Contention Resolution Diversity Slotted Aloha (CRDSA) [21] where time is divided into frames. Each frame is divided into slots of equal

[2]Note that a classical collision model, here, is when two or more packets arrive simultaneously at a receiver that can handle only one at any given time.

sizes. Each packet is sent twice in the random-access frame. The slots with single replicas are considered always as successfully decoded. SIC is used to subtract the decoded packet and retrieve it from its other position on the frame. The resulting throughput is up to $T \approx 0.55$ packets/slot.

In Irregular Repetition Slotted Aloha (IRSA) [1], the number of repetitions can be different for different users (i.e. it can be randomly selected from a predefined probability distribution). The SIC process is similar to the decoding process of low-density parity-check (LDPC) codes. This allows applying the analytical tools of LDPC codes (namely, *density evolution*) to analyze and then optimize the asymptotic performance of IRSA as pioneered by [1]. Part of the research of IRSA and its variants focuses on optimizing the probability distribution used to randomly select the repetition degree of each user. Indeed, in the original IRSA article [1], thanks to the developed analysis tools, a probability distribution that could reach asymptotically the throughput of ≈ 0.97 packet/slot (reached when the frame size grows towards infinity) was numerically found. Shortly after, by exploiting the same tools, but furthering the analogy with LDPC codes, [6] proved that the soliton distribution is also suited to IRSA, and would yield asymptotically a throughput of 1 packet/slot. This is the maximum that can be obtained with a classical collision model.

The authors of [5, 22] introduce Coded Slotted ALOHA (CSA) which is a generalized version of IRSA. In CSA, each data block of a user is split into several packets before applying a packet-oriented linear block code yielding coded packets, instead of simply using a repetition code like IRSA. The linear block code is selected from a predefined set, according to a *code probability distribution*. The *code rate* is defined as the inverse of the average number of packets repetitions for IRSA and as the average code rate of the linear block codes for CSA. Then, in CSA, each slot and each packet is divided into k segments and each packet is randomly coded with a code of rate $\frac{k}{n}$ where n is drawn from a probability distribution for each user in each frame. On the receiver side, SIC is integrated with decoding of the local component codes to recover collisions.

2.3.2 Variants of IRSA Protocol

Recently, many studies conducted research on CSA-type modern random access protocols due to their expected advantages in terms of improving the random access of devices in IoT networks and providing high efficiency. The family of Coded Slotted ALOHA (CSA) protocols is very generic: it can be applied as soon as packets are repeated, and the receiver is equipped with SIC. For that reason, there have been many variations of the CSA protocols. As indicated, IRSA is known to reach asymptotically the maximum throughput of

1 [packet/slot] in a collision model; but it turns out that it is even possible to go beyond this performance, with more *realistic* model assumptions. Thus, some CSA family variations have explored different system assumptions; for instance, different realistic channel conditions or modern physical layer designs that have been proved to enhance the protocol performance. In this section, we describe some of such studies that have considered Multiple Packet Reception (MPR) at the receiver, varying transmission powers, fading and erasure channels, leading to potentially beneficial capture effects. Because properties such as fading are usually associated with packet losses, some other related research studies have assessed their negative impact and have handled the decoding imperfection due to transmissions or SIC errors.

First, IRSA schemes, for receivers capable of decoding multiple colliding packets jointly, have been introduced in multiple research works. Usually, the capability of the receiver to receive K packets at the same time is referred to as K-MPR and can be usually achieved in a wireless network with some diversity technique [23]. Note that K-MPR would allow significant throughput gain, but from an information-theoretic capacity perspective, one would expect that it requires an equivalent decrease in individual capacity by a factor K or greater.

[24], studied the effect of K-MPR (under the name of *multiuser detection*) on Irregular Repetition Slotted ALOHA (IRSA) which increases the chance of decoding more packets under large system loads. It has been shown by simulations that a multiuser detector supports larger loads for a low number of slots jointly processed. Another work in [2] has tackled the K-MPR capabilities at the receiver for Coded Slotted Aloha (CSA). Their converse bound on the asymptotic normalized load threshold of coded slotted ALOHA schemes (G/K) appears to increase with K. They also numerically show for $K = 1$, $K = 2$ and $K = 3$ that some IRSA schemes, seem to achieve a performance close to K [packets/slot]. This shows that exploiting the K-MPR capability is maybe useful, even if the amount of resources required to achieve it scales linearly with K. In [25], an optimized transmission probability distribution was analytically derived for IRSA with 2-MPR.

Another major question in CSA literature is the achievable throughput in realistic network scenarios, considering fading, limited frame length, decoding errors, or the probability to lose some packets. At the physical layer level, the highest-power packet might be successfully recovered despite the presence of other interfering packets. This is called the *capture effect*. The capture effect may be exploited to recover the underlying packets by removing the recovered high-power packets. In [26], the analysis of IRSA considered a block fading channel model, and also assumed capture effect at the receiver. New density

evolution equations are required to study such a scenario. The work optimized some repetition degree distributions in a finite frame length setting. The derived distributions are shown to achieve throughputs largely exceeding 1 [packet/slot]. In [27], Extended IRSA (E-IRSA) has been proposed as an extended operation of IRSA that exploits the capture effect to perform iterative decoding. Intra-slot SIC has been applied to decode more than one collided packet per slot. Simulation results show that the E-IRSA protocol allows reaching the maximum theoretical throughput even in scenarios where the number of active users is higher than the number of slots per frame. CSA performance has been evaluated assuming a fading channel in [28–30]. The impact of the capture effect on the SIC-enabled slotted ALOHA framework has been also addressed in [31, 32]. In general, the capture effect, combined with SIC, helps to increase throughput.

A related direction of studies of modern random access protocols is to consider the control of the transmission power. Packet replicas can also be transmitted in different power levels that are deliberately chosen to exploit the capture effect at the receiver. In this case, when intra-SIC dominates, packet collisions can be resolved in the power domain. New density evolution should be formulated in this case. In [33], a packet is transmitted with a power level chosen according to a power-choice distribution, such that multiple packets sent by different nodes might be decoded at the receiver in a single slot, by ensuring that their Signal-to-Interference-and-Noise Ratio (SINR) is above the decoding threshold of the successive interference cancellation receiver. In [34], an optimal *Power-Limited IRSA* (PL-IRSA) has been proposed as a variant of IRSA for the Internet-of-Things (IoT)-oriented satellite networks. A developed density evolution analysis considering total transmission power constraints and iterative IC processing has been derived to compute the PLR of PL-IRSA and to find the optimal degree distributions that maximize the normalized throughput. The work concludes that different total transmit power constraints and code rates can lead to completely different optimal degree distributions. Therefore, optimal degree distributions should be calculated according to the total transmit power constraint and code rate for practical implementations. A novel paradigm whereby devices adapt their data rates and/or transmit powers based on their chosen repetition degrees has been introduced in [35]. In [36], another approach is proposed to improve the throughput of the IRSA protocol. The total transmit power of all the users is considered to be the same, while the transmit power of a replica is determined by the number of replicas. The transmit power of each replica of different users can also be different. The study did not consider channel fading.

Another IRSA variant using temporal diversity and the capture effect has been introduced in [37]. The work optimized the repetition degree distribution along with the param-

eter of power control to maximize the total system throughput. In general, adjusting the transmission power is an excellent way to control the capture effect and further improve the performance. When modeling IRSA or CSA protocols (e.g., with density evolution), a common assumption in the CSA literature is that interference cancellation (IC) can always be applied perfectly and added that a singleton packet (the packet that is alone on the slot) can be successfully decoded with probability 1. Nevertheless, there have been some studies with the assumption of SIC imperfection. Two possible causes: the first one is that there might have been transmission errors, such as one replica being subject to strong fading and being received with very low power (leading to a "packet erasure"), or alternately the presence of interference on one slot (leading to a "slot erasure"). The second one can be errors in signal parameters' estimation (amplitude, phase, frequency offset) before the decoding that might lead to incorrect signal subtraction (cancellation). After each SIC operation, residual energy is present due to the imperfection of the cancellation process. The residual energy would accumulate, and SIC errors would become more likely as the number of SIC operations on the same slot increases. CSA over a packet erasure channel and a slot erasure channel has been studied and optimized in [38, 39]. [40] presented a framework to design degree distributions for finite frame length CSA system with different classes of users, which have different error rate requirements. The classes are assigned different distributions. The work shows that multi-class CSA is capable of providing different levels of protection at high channel loads, as well as a smaller average decoding delay for better-protected users.

Combining one of the CSA protocols with NOMA has also been widely inspected. Literally, CSA protocols are "non-orthogonal" and hence are NOMA methods. But in general, compared to CSA, many of the proposed NOMA methods in the literature have more detailed assumptions on the physical layer. This might owe to the act that NOMA had been a study item in the 3GPP for 5G Release 16 (see for instance [41]), and has often been studied in that context. Thus compared to plain CSA or plain IRSA, some NOMA methods can include features such as joint-decoding, multichannel transmissions, pilot-based user detection, etc.; although it can also happen that some NOMA methods are equivalent to some CSA methods. In [42] a scalable, energy-efficient pure ALOHA with power domain non-orthogonal multiple access (NOMA) has been proposed. The results of combining ALOHA and NOMA show that there is a greater than linear increase in throughput as the number of active IoT devices increases. In [43], the transmission power of each packet was chosen by a power distribution, where discrete power levels pre-determined by the NOMA method have been used to send replicas. It is shown that the proposed scheme in [43]

can outperform the existing IRSA schemes. Another combination of slotted-ALOHA and NOMA has been proposed in [44]. NOMA has been also combined with multichannel ALOHA in [45, 46].

In the next section, we introduce the IRSA protocol fundamentals, and we explain in detail its decoding process.

2.4 Irregular Repetition Slotted Aloha (IRSA)

We adopt a system model commonly found in the IRSA literature. An IoT network with M user nodes contending to send their packets to a single base station through a connectionless random access scheme. The time is divided into slots of equal length, and successive slots are regrouped in fixed-size frames of N slots. The load is defined by, $G = \frac{M}{N}$ and it measures the average number of users per slot. We focus on one frame and assume that each user has exactly one data packet to transmit in the frame. A collision channel model is assumed to study IRSA. The collision channel model has been widely used for designing IRSA schemes, where packets in a collision are lost and packets without a collision are successfully decoded [43].

2.4.1 IRSA Concept

IRSA works as follows: Each user sends his packet ℓ times within the same MAC frame, where the repetition rate of each user is selected randomly from a probability distribution Λ defined by $L+1$ probabilities : $(\Lambda_i)_{i=2,\dots,L}$. Λ_i is the probability that the packet is repeated i times. At the end of the frame, the receiver attempts to decode the packets: each time that a packet is successfully received, e.g. only one transmission is present on the slot, IRSA will use SIC to remove the physical copy of this packet on all other slots. This process is iterative, as new decoding opportunities can appear.

In the example of Fig. 2.2, 5 users are transmitting in a frame of 5 slots; the packet of user D can be recovered on the slot 4. When its copy is physically subtracted from the slot 5, the packet of user B on the slot 5 can be recovered and removed from slots $1, 2$ and 3. Then, the packet of user A on slot 3 could be decoded and its copies on slots 1 and 2 could be extracted. Finally, the packets of users C and E form a stopping set as they sent two copies on the same two slots, so they can not be decoded.

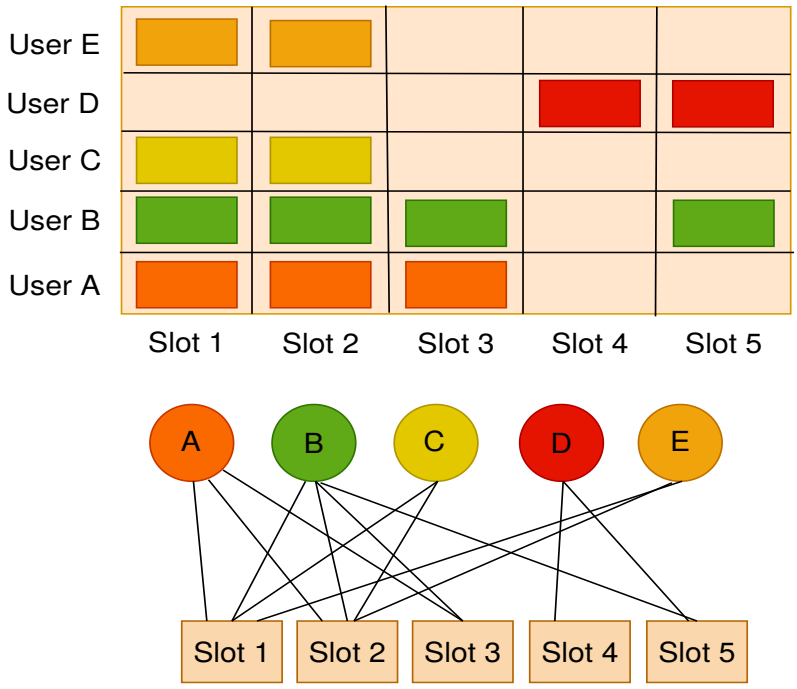


Figure 2.2: IRSA representation: transmissions of users in slots (top), coding theory representation to model the decoding process (Tanner graph, bottom). Note that transmissions are in the same frequency channel, hence when two users are transmitting on the same slot, there is a collision.

2.4.2 IRSA Decoding Process Modeling

In [1], Liva introduced, by analogy with the decoding of LDPC codes (on a binary erasure channel [20]) the way to analyze the decoding process of IRSA as follows: as for codes, one can construct a Tanner graph, that is, a bipartite graph $G(B, S, E)$, where B is the set of *burst nodes* (corresponding to users), S, is the set of *slot nodes* (corresponding to slots) and E is the set of edges. An edge connects a burst node $b_i \in B$ and to a slot node $s_j \in S$, if and only if a replica of the i-th burst is transmitted in the j-th slot. The number of edges connected to a node is referred to as the node degree.

Fig. 2.2 (on the bottom) shows an example of a graph representation of the IRSA. The circles correspond to the burst nodes, while the rectangles correspond to the slot nodes. The IC process starts by searching for the slots where only one replica was sent (degree-1 slots), which are also known as singletons, and decoding these replicas. If the burst that is transmitted on a certain slot is revealed, the corresponding edge is labeled by '1', otherwise, it is labeled by '0' (or equivalently, it is *removed*). The contribution of each revealed burst is removed from the other slots, where a replica of this burst was transmitted through SIC. Thus, we can say that the first iteration has ended.

After the first iteration, new singletons can be recovered. The IC process can be blocked if, in one iteration, no singletons could be recovered. The analysis of IRSA assumes that we can always decode singletons where no collision has happened and then, we can remove their contributions in other slots.

2.4.3 Density Evolution

Density evolution (DE) is a method to analyze the asymptotic performance of different (framed) variants of IRSA protocol. It is based on an analogy with LDPC codes for which DE was initially introduced, as an analytical tool for analyzing the asymptotic network capability to approach the error-correcting codes [47]. DE helps to track the iterative decoding process and have an estimation of the number of decoded packets at the end of the process. Using DE and given the system parameters Λ, M, N, we can answer the following question: can we expect to recover all packets, or how many of them?

As we have already shown, the iterative decoding process of IRSA can be represented by the Tanner graph. A general approach to analyze random graphs with a given degree distribution [48] is to introduce a form of a generating function as $\sum_{k=0}^{\infty} p_k x^k$ [49]. This form of function helps for calculating various local and global quantities on large unipartite undirected graphs with an arbitrary probability distribution of the degrees of their vertices (nodes) [50].

Following [1], and the traditions of the LDPC coding literature (see [47, Eq. 3.21] for instance), we define the polynomial representations (probability generating functions) of burst- and slot-perspective degree distributions which will be helpful for further calculations:

$$\Lambda(x) \triangleq \sum_{\ell=0}^{\ell=L} \Lambda_\ell x^\ell \text{ and } \Psi(x) \triangleq \sum_{\ell=0}^{\ell=N} \Psi_\ell x^\ell \tag{2.1}$$

where Λ_ℓ, is the probability that a user will send ℓ replicas of his packet, and Ψ_ℓ is the probability that a slot has ℓ collided packets. $\Lambda'(1) = \sum \ell \Lambda_\ell$ is the average burst repetition rate. We define the *rate* R as $R \triangleq \frac{1}{\Lambda'(1)}$. Notice, in IRSA, the "rate" R is loosely connected to throughput since with SIC, removed replicas do not consume slots. On the other hand, it is strongly connected to the energy-efficiency, as the average energy cost would be proportional to $1/R$. The probability that one user sends a packet on a given slot is: $\frac{\Lambda'(1)}{N} = \frac{G}{RM}$. Then Ψ_ℓ follows a binomial distribution $\text{Bin}(M, \frac{G}{RM})$. When $M \to \infty$, the distribution $(\Psi_\ell)_{\ell \geq 0}$ becomes a Poisson distribution with a parameter $\frac{G}{R}$. [3]

[3] It is actually an application of the classical Poisson approximation, i.e., the Poisson distribution with

If we take a random edge and consider the burst node that it connects to, such an edge arrives at the burst node with a probability proportional to the degree of that node. Therefore, the burst node associated with a random edge itself has a probability distribution of degree proportional to $\ell\Lambda_\ell$. The correctly normalized probability generating function for the degrees of the burst nodes sampled in the described way is given by:

$$\frac{1}{\sum_\ell \Lambda_\ell \ell} \sum_\ell \Lambda_\ell \ell x^\ell = x\frac{\Lambda'(x)}{\Lambda'(1)} \tag{2.2}$$

If we start at a random burst node and pick one of its edges: we are interested in the distribution of the number of remaining edges that the burst node has. This distribution is generated by the function in Eq. (2.2) except for less than one power of x, since it is only the distribution of the number of the remaining edges. It is customary [47] to use instead of the corresponding edge distributions from the burst and slot perspective which are respectively:

$$\lambda(x) \triangleq \sum_\ell \lambda_\ell x^{\ell-1} = \frac{\Lambda'(x)}{\Lambda'(1)} \text{ and } \rho(x) \triangleq \sum_\ell \rho_\ell x^{\ell-1} = \frac{\Psi'(x)}{\Psi'(1)} \tag{2.3}$$

where λ_ℓ is the probability that an edge connected to a burst node of degree ℓ, and ρ_ℓ is the probability that an edge connected to a slot node of degree ℓ and they are given by the following definitions:

$$\lambda_\ell = \frac{\ell\Lambda_\ell}{\sum_\ell \ell\Lambda_\ell} \text{ and } \rho_\ell = \frac{\ell\Psi_\ell}{\sum_\ell \ell\Psi_\ell} \tag{2.4}$$

Note that the rate could be also defined as: $R \triangleq \sum_\ell \frac{\lambda_\ell}{\ell}$.

When $M \to \infty$, from the Poisson approximation of $(\Psi_\ell)_{\ell\geq 0}$, we have $\rho(x) \to e^{-\frac{G}{R}(1-x)}$.

For further clarification, the burst and edge probability distributions are computed for the example shown in Fig. 2.4.

Still following [1], the decoding process is iterative and is modeled probabilistically, tracking variables representing probabilities that the decoding information (slot content) is unknown over iterations. Each edge in Fig. 2.2 connects one slot node, and one burst node. Let p be the probability that a randomly selected edge is connected to a burst node that is not recovered yet, and q is the probability that a randomly selected edge is connected to an *unknown* slot node.

We can decode the packet of a user if a replica of this packet has been decoded at least once on any other slot. Hence:

$$q = p^{\ell-1} \tag{2.5}$$

parameter $\mu = \alpha\beta$ can be used as an approximation of the binomial distribution $\text{Bin}(\alpha, \beta)$ for large α and small β. In our case, $\alpha = M, \beta = \frac{G}{RM}$ and $\mu = \frac{G}{R}$.

Similarly, in classical IRSA, we can remove the collision on the last edge which connects to a slot node of degree ℓ, if we have already removed the $\ell - 1$ contributions of the other collided packets on the same slot. Hence:

$$(1 - p) = (1 - q)^{\ell - 1} \tag{2.6}$$

Fig. 2.3 illustrates how to obtain the DE equations in Eq. (2.5) and Eq. (2.6).

The above described one iteration from p to q and then back from q to p, assuming an imaginary graph where all nodes would have the same degree ℓ. Density evolution aims to find the limit (minimum) values of p and q after several (infinitely many) iterations on all the edges of any general Tanner graph obtained from a given probability distribution Λ (thus not only Tanner graphs with nodes with fixed degree ℓ).

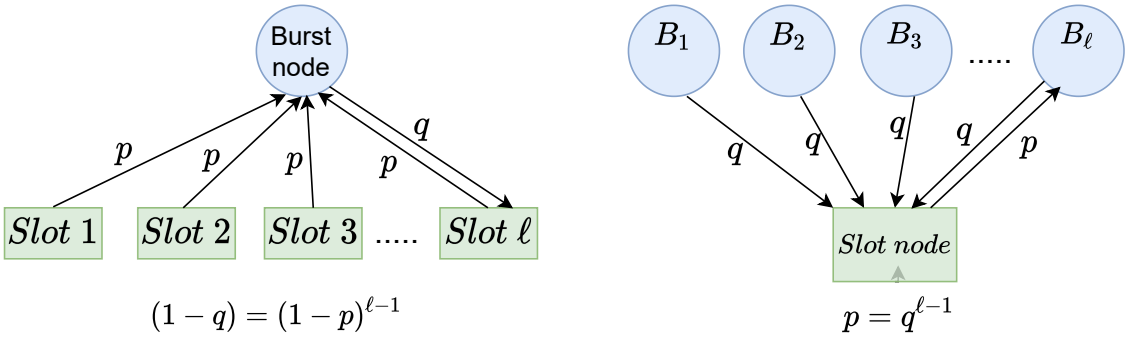


Figure 2.3: A simple illustration of basic density evolution equations

Following the tree analysis argument of [51], by averaging the two expressions in Eq. (2.5) and Eq. (2.6) on all possible degrees, we get respectively:

$$\bar{q} = \sum_{\ell} \lambda_\ell \bar{p}^{\ell - 1} \tag{2.7}$$

$$\bar{p} = 1 - \sum_{\ell} \rho_\ell \, (1 - \bar{q})^{\ell - 1} \tag{2.8}$$

For simplicity, we note the average probabilities $\bar{p}$ and $\bar{q}$ over all the edges and their evolution through iterative decoding as: p_i and q_i at iteration i. Then, we can derive the evolution of the average probabilities during the i-th iteration as:

$$q_i = f_b(p_{i-1}) \text{ and } p_i = f_s(q_i), \text{ thus } p_i = f_s(f_b(p_{i-1})) \tag{2.9}$$

with, functions corresponding to [1], for classical IRSA:

$$f_b(p) = \lambda(p) \text{ and } f_s(q) = 1 - \rho(1 - q) \tag{2.10}$$

$$\text{when } M \to \infty\text{: } f_s(q) \to F(\frac{G}{R} q) \text{ with } F(x) \triangleq 1 - e^{-x} \tag{2.11}$$

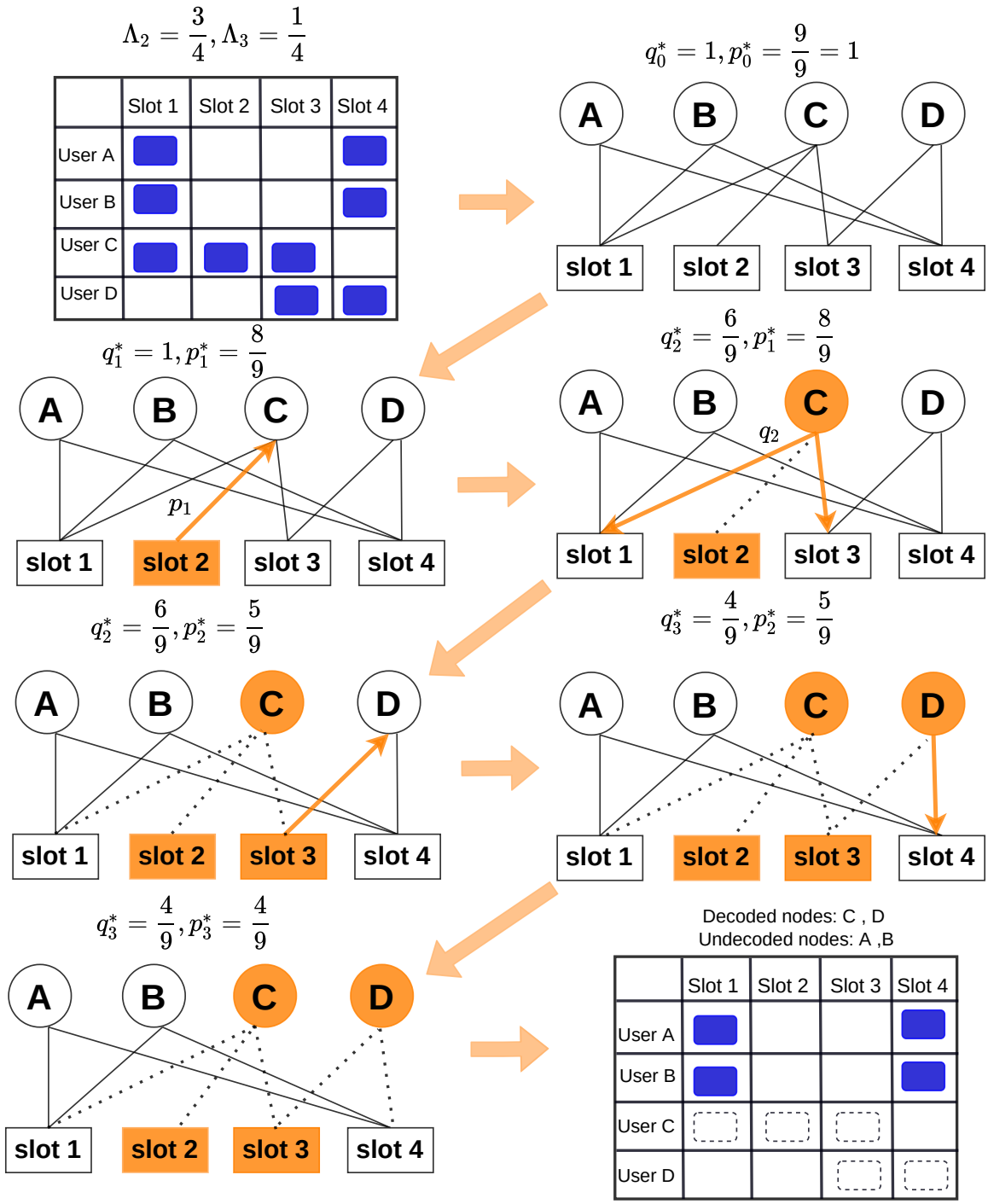


Figure 2.4: Example of the decoding process of IRSA with Density Evolution parameters p^* and q^*

Note that the sequence $(p_i)_{i=0,\ldots,\infty}$ characterizes how many of the edges in the graph correspond to undecoded users at each iteration i. Note that the sequence $(q_i)_{i=0,\ldots,\infty}$ can equivalently be considered. Initial values are $q_0 = 1$, hence $p_0 = 1 - \rho(0)$. Alternately, picking $p_0 = 1$ just shifts the sequence by one, as it implies that $q_1 = \lambda(1) = 1$.

Fig. 2.4 shows a detailed example of the decoding process of 4 users who compete for 4 slots. The users are using the degree 2 for $\frac{3}{4}$ of the time and the degree 3 for $\frac{1}{4}$ of the time. In this example, we illustrate the computation of the burst degree and the edge degree distributions, and we follow in detail the decoding process with the density evolution parameters. The equivalent Tanner graph represents the users in circles, the slots in rectangles, and the transmissions of the users on the slots as 9 edges between the users and the slots. The number of edges represents the number of sent packets on the radio access frame. An important piece of information should be noted: the values of p_i^* and q_i^* given in the Fig. 2.4 are the actual empirical values computed on this specific instance of a Tanner graph. The values of p_i and q_i obtained from Eq. (2.9) are estimates (expectations) of these p_i^* and q_i^* for random instances of all such Tanner graphs, when furthermore Λ and load G are kept fixed, but frame size becomes infinitely large.

- **The degree distributions**: The user degree distribution, based on the presented definition in Eq. (2.1), is given as: $\Lambda(x) = \frac{3}{4}x^2 + \frac{1}{4}x^3$. Now considering the edge degree distribution of the edges, the probability that a randomly selected edge is connected to a node of degree k is proportional to kN_k, where: k is the node degree and N_k is the number of degree k nodes. Thus, the probability that a randomly selected edge is connected to a burst node of degree 2 is $\lambda_2 = \frac{2\times N_2}{\sum_i iN_i} = \frac{2\times 3}{2\times 3+3\times 1} = \frac{6}{9}$, as there are 3 burst nodes that have a degree 2. Following the same manner: $\lambda_3 = \frac{2\times N_3}{\sum_i iN_i} = \frac{3\times 1}{2\times 3+3\times 1} = \frac{3}{9}$. Therefore, the edge degree distribution from the burst perspective, based on the presented definition in Eq. (2.3), is given as: $\lambda(x) = \frac{6}{9}x + \frac{3}{9}x^2$, which also can be computed directly by taking the first derivative of $\Lambda(x)$ and normalizing, as in Eq. (2.3).

  Equivalently, we can compute the edge distribution from the slot node perspective. The probability that a randomly selected edge is connected to a slot node of degree ℓ is also proportional to ℓN_ℓ, where N_ℓ is the number of the slot nodes that have a degree ℓ. Thus, the probability that a randomly selected edge is connected to a slot node of degree 1 is $\rho_1 = \frac{1}{9}$, as there is only one edge out of 9 edges connected to a degree 1 slot node. Consequently, the probability that a randomly selected edge is connected to a slot node of degree 2 is $\rho_2 = \frac{1\times 2}{9}$ and to a slot node of degree 3 is $\rho_3 = \frac{6}{9}$. As a result, the edge degree distribution from the user perspective, based on the presented definition in Eq. 2.3, is give as: $\rho(x) = \frac{1}{9} + \frac{2}{9}x + \frac{6}{9}x^2$.
- **The decoding process**: Before the decoding starts, the probability that a randomly selected edge is unknown q_0, or the probability that a randomly selected edge can not be revealed p_0 are initialized as previously described, i.e. either $q_0 = 1$ or $p_0 = 1$.

  Now, back to density evolution: how can one estimate whether that all the packets will

be ultimately decoded? From the decoding point of view, and for an infinite frame size, we will be able to decode if the probabilities p_i are decreasing towards zero, which requires that $p_i < p_{i-1}$. When $M \to \infty$, $p_i = F(\frac{G}{R}\lambda(p_{i-1}))$, thus this is obtained when (necessary condition):

$$F(\frac{G}{R}\lambda(x)) < x \text{ for all } x \in (0, 1) \tag{2.12}$$

Given the parameters (Λ, G, ...), one can simply check the curve, e.g. equation (2.12), to determine if and where the decoding process will stop (asymptotically for $M \to \infty$).

Following [1], one main property is that: given Λ, there exists a *load threshold* G^*, such that, when the frame size converges towards infinity ($M \to \infty$), for any load $G < G^*$, all the packets are decoded with vanishing error probability. G^* depends on the choice of the distribution Λ. Finding the Λ with the largest G^* is our prime interest.

For later reference, Table 2.1 summarizes the main system parameters and variables used in this thesis.

2.4.4 General Problem Statement for IRSA

Many studies have explored and analyzed the performance of IRSA in an IoT network. The main purpose of these studies is generally to find and study a better variant of this protocol. Given any IRSA system, the goal is often to find an optimized user degree distribution that maximizes a certain metric (throughput, achievable load,..., etc.). This problem is usually formulated as an optimization problem, which can be described as follows:

$$\begin{aligned} &\underset{(\Lambda_i)}{\text{maximize}} && \mathcal{O}(\Lambda_0, \Lambda_1, .., \Lambda_D) && (\mathcal{P}_1) \\ &\text{subject to} && 0 \leq \Lambda_i \leq 1 \ \forall i \\ & && \sum_{i=0}^{i=D} \Lambda_i = 1 \end{aligned}$$

where $\mathcal{O}$ is the system criteria that needs to be optimized and D is the maximum degree. Depending on the system model, more constraints can be added to the optimization problem. Notice that the system is initially a stochastic optimization problem, as the performance depends on the random variables of the users' arrivals and their degree selections. The mentioned Density Evolution (DE) method allows analyzing the expectation of the asymptotic performance of different (framed) variants of the IRSA protocol deterministically. For several IRSA variants, the problem may still be difficult to solve, for example, in the case of the non-convexity or non-linearity of the constraints. But the optimization

Table 2.1: Main Parameters and Variables

General IRSA Notations

Notation	Description
M	Number of users
N	Number of slots

IRSA Performance Metrics and Objectives

Notation	Description
G	Load of the system: $G \triangleq \frac{M}{N}$
T	Throughput of the system: $T \triangleq \frac{S}{N}$, where S is the number of decoded users
PLR	Packet loss rate (PLR): the proportion of non-decoded users at the end of the iterative process, PLR $\triangleq$ $G\,(1 - PLR(G))$
G^*	Load threshold: the maximum load below which (almost) all the packets are decoded when $M \to \infty$

Density Evolution Analysis

Notation	Description
Λ	Burst node degree distribution: $\Lambda(x) \triangleq \sum_\ell \Lambda_\ell x^\ell$
ψ	Slot node degree distribution: $\Psi(x) \triangleq \sum_\ell \Psi_\ell x^\ell$
λ	Polynomial representation of the burst edge degree distribution: $\lambda(x) \triangleq \sum_\ell \lambda_\ell x^{\ell-1}$
ρ	Polynomial representation of the slot edge degree distribution: $\rho(x) \triangleq \sum_\ell \rho_\ell x^{\ell-1}$
R	Rate of the distribution: The rate defines the average packet repetition rate. $R \triangleq \frac{1}{\Lambda'(1)}$
p_i	Probability that a randomly chosen edge (modeling a packet) cannot be retrieved at the i-th iteration of the iterative process by applying SIC on its slot
q_i	Probability that a randomly chosen edge cannot be decoded at the i-th iteration of the iterative decoding process.

problem formulation may be useful. In the following chapters, for some variants of IRSA, we will adapt the OP described in ($\mathcal{P}_1$) to optimize the performance of IRSA.

2.5 Conclusion

Our starting point is the Internet of Things (IoT). It is a recent set of technologies that creates connections between billions of devices and machines to enable them to communicate and exchange data. The main challenge for future IoT networks is the need to provide connectivity for IoT devices with stringent latency requirements. Therefore, the massive connectivity has become one of the main use cases to study for future IoT applications.

In this chapter, we highlighted the importance of IoT, we referred to its benefits, and we addressed the challenges on which this thesis focuses, mainly, the massive connectivity. We described the features and the requirements of massive connectivity problem (also denoted mMTC). Then, we presented the main wireless connectivity standards, and in what directions they are evolving to provide solutions for the mMTC problem. We also described some main wireless techniques, mainly, random access techniques that have been used for many suggested wireless standards. In this thesis, we focus on one of these techniques, which is the family of modern random access of protocols, CSA, and fundamentally, one of the recent protocols IRSA. We showed in detail the concept underlying and the operation of this protocol, in addition to its decoding process modeling.

In the rest of the thesis, the behavior and variations of the IRSA protocol will be our focus.

Chapter 3

IRSA with Multiple Packet Reception

3.1 Introduction

As mentioned previously, the main topic of the thesis is on the family of protocols IRSA and CSA. In the first part of the thesis, and in this chapter in particular, we are exploring how, and how much, one can go *beyond* the "optimal" performance of IRSA of 1 [packet/slot] by adopting more realistic assumptions of communication channels. Indeed, some recent publications [2,24,52] have focused on specific variants of CSA protocols to handle the massive connectivity issue assuming a different model of the collision channel: in these new variants, the receiver has the capability of simultaneously decoding more than one packet from multiple concurrent transmissions, denoted *Multiple Packet Reception* [53] (MPR). MPR is the capability of simultaneous decoding of more than one packet from multiple concurrent transmissions. There are several techniques to allow simultaneous decoding of packets on a receiver [54]. In [23], a classification based on transmitter perspective, trans-receiver perspective, and receiver perspective has been introduced. Orthogonal multiplexing systems such as Time Division Multiple Access (TDMA), Code Division Multiple Access (CDMA) or Orthogonal Frequency Divion Multiple Access (OFDMA) are examples of multiuser techniques that combine different users' signals by using different time slices, orthogonal codes and multiple frequency ranges respectively [55]. From a transceiver perspective, to enable MPR, transmitters and receivers should cooperate on some operations. Multiantenna MIMO systems [56] can achieve MPR by exploiting the spatial diversity of the transmissions. Alternately, the MPR capability can be shifted from the transmitter to the receiver by using a bank of Matched Filters to decode packets coded with spreading codes that does not need to be orthogonal.

The ability to decode K or fewer packets simultaneously is usually denoted *K-MPR*,

and a derivative of the *collision channel* described in Section 2.4 can be introduced as the K-MPR collision channel, where packets are retrieved if and only if less than or exactly K packets are transmitted on the same slot.

The importance of combining multiple packet reception feature with IRSA comes from the fact that MPR exploits the structure of the interfering signals, which increases the number of users that the system can sustain. We point out in Section 3.2.3 some studies of CSA protocols with MPR capability. These studies have proved that equipped IRSA with MPR increases the chance of decoding more packets under large system loads. But, to our knowledge, there have been no existing result that provides optimal degree distributions that reach K packets per slot, equivalently to the soliton distribution that attains the bound of 1 [packet/slot] for classical IRSA under a collision channel model.

Motivated by this observation, in this chapter we focus on IRSA with K-MPR. It is denoted K-IRSA, and supposes that the transmissions occur in fixed slots (as for slotted ALOHA). Considering the *throughput*, under this K-MPR collision channel model, K-IRSA can recover at most K packets per slot. Classical IRSA, which is 1-IRSA, asymptotically reaches this performance [6]. Hence, a natural question arises: can K-IRSA reach the bound of K? With which distributions? as, K-MPR can typically be obtained at a cost of a decrease in modulation rate by a factor K.

Previous studies have explored the question, [2, 24], by describing a non-asymptotic bound for K-IRSA, and numerically exhibiting some specific modification of CSA. These studies show results close to their bound for $K = 3$ and 4, but they are not able to fully answer the question of the exact bound, and of reaching it with K-IRSA.

The main contribution of this chapter is to prove that classical K-IRSA cannot actually reach this bound of K packet per slot, even in the asymptotic case (when frame size grows towards infinity). A new bound is provided. The results also hint that the method for finding the optimal soliton distribution families [6] for IRSA (1-IRSA), might not work for K-IRSA with $K > 1$. Furthermore, we formulate an efficient search for optimal K-IRSA parameters (namely the probability distributions) using a sequence of linear programs, and we show numerically how the new bound can be approached by the computed distributions. To complement the work, this chapter also studies the behavior of K-IRSA for low loads: we provide a simple asymptotic expression for the loss rate at low loads (the *error floor*), evidencing that increasing K dramatically decreases it.

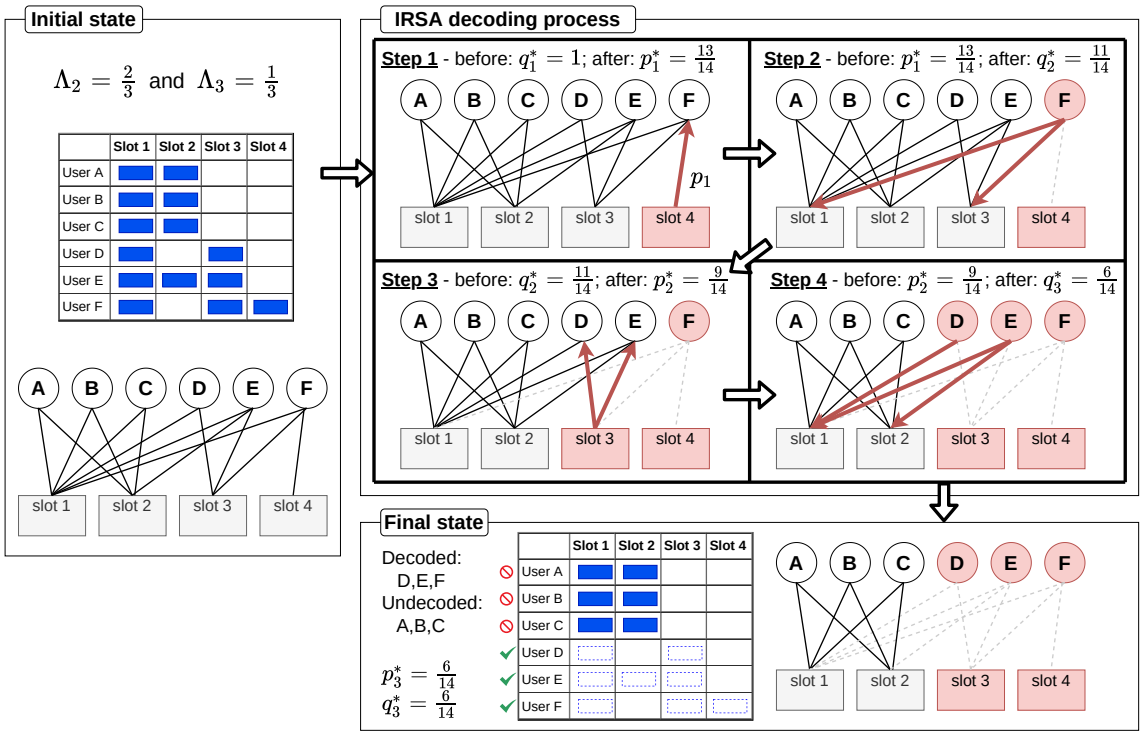


Figure 3.1: Example with 2-IRSA (K=2)

3.2 System Overview and Related Work

3.2.1 System Description

We adopt the IRSA system model explained in Section 2.4. For the K-IRSA variant, we modify it, and we assume that the receiver has K multiple packet reception capability. At the end of the frame, the K-MPR receiver attempts to decode the packets: each time that there are K or fewer packets on the slot, the K-MPR receiver can retrieve the packets (this is the K-MPR collision channel model). Then, K-IRSA will use SIC to remove the physical copy of the decoded packets at their positions on other slots. This process is iterative, as new decoding opportunities can appear.

The example, shown in Fig. 3.1, illustrates 6 users transmitting their packets in a frame of 4 slots. The users are using IRSA protocol to send their packets towards a 2-MPR receiver. As the figure shows, $\frac{1}{3}$ of the time, the users use the degree 3, and $\frac{2}{3}$ of the time, the users repeat their packets two times. The decoding process starts by searching for singletons (slots of a single transmission) or the slots of two colliding packets (as the receiver has 2-MPR capability).

As in Section 2.4.3, we can consider the sequences of probabilities p_i, q_i, the probabilities that a random edge connected to one burst node or a slot node is unknown or

unrevealed. We start with no packet decoded: $p_0 = 1$ and thus $q_1 = 1$. In this section, we also denote the empirical probabilities obtained on the instance of the graph obtained in Fig. 3.1 as p_i^*, q_i^*.

At the first decoding iteration, user F on slot 4 is decoded and the edge between this user and the slot 4 can be revealed. Consequently, the probability that one edge cannot be retrieved from the slot perspective, p^*, after the first iteration is $\frac{13}{14}$. This probability represents the number of unrevealed edges in the corresponding bipartite graph over the total number of initial edges in the graph. In the second iteration, the two other copies of user F on slots 1 and 3 can be extracted. This makes the probability that an edge is unknown from the user perspective equal to $q_2^* = \frac{11}{14}$, which represents:

$$\frac{\text{The number of total edges} - \text{The number of retrieved edges}}{\text{The number of total edges}}$$

The revealing of the packet of user, F followed by its removal with SIC on the slot 3, leaves 2 collisions from users D and E on the slot 3. As the receiver is 2-MPR type, the two collided packets on slot 3 can be decoded, and their copies on slots: user D on slot 1, and user E on slots $1, 2$ can be removed. This process of iterative decoding continues until all packets are decoded, or the decoding process is blocked by a stopping set. In this example, users A, B and C sent their packets twice on the same slots 1 and 2. This creates a stopping set of three colliding users on the slots 1 and 2, making it impossible to retrieve the packets of those users.

3.2.2 Density Evolution

We use density evolution equations to track the decoding process of K-IRSA (see Section 2.4.3). For that aim, we need to adapt the density evolution equations for the K-MPR scenario.

For later mathematical convenience, and given the system parameters Λ, M, N and K, we reintroduce the concept of burst- and slot-perspective degree distributions, which was already presented in Chapter 2, mainly in Section 2.4.3. We will use the set of basic IRSA model equations in Table 3.1 for later analysis.

In the next section, we will highlight the main works on combing CSA protocols with MPR. Then, we will update and formulate the density evolution equations in Table 3.1 for the K-MPR case.

General IRSA Notations

Equation	Reference
$\Lambda(x) \triangleq \sum_\ell \Lambda_\ell x^\ell$	Burst node degree distribution in Eq. (2.1)
$\Psi(x) \triangleq \sum_\ell \Psi_\ell x^\ell$	Slot node degree distribution in Eq. (2.1)
$\lambda(x) \triangleq \sum_\ell \lambda_\ell x^{\ell-1}$	Polynomial representation of the burst edge degree distribution in Eq. (2.3)
$\rho(x) \triangleq \sum_\ell \rho_\ell x^{\ell-1}$	Polynomial representation of the slot edge degree distribution in Eq. (2.3)
$q = p^{\ell-1}$	Probability on the edge towards a slot node in Eq. (2.5)
$(1-p) = (1-q)^{\ell-1}$	Probability on the edge towards a burst node in Eq. (2.6)
$q_i = f_b(p_{i-1})$ and $p_i = f_s(q_i)$	In Eq. (2.9)
when $M \to \infty$: $f_s(q) \to F(\frac{G}{R}q)$ with $F(x) \triangleq 1 - e^{-x}$	Asympotic expression from Eq. (2.11)

Table 3.1: Density Evolution equations used for K-IRSA analysis

3.2.3 K-IRSA Related Work

In this section, we give insights about existing research studies that applied MPR to enhance the performance of CSA protocols. We also present the analysis of the conditions under which the K-MPR receiver is able to decode the packets successfully.

Different variants of IRSA and CSA have been studied, often by establishing new functions $f_b(q)$ and $f_s(p)$ (and $F(x)$) for Eq. (2.12) for the density evolution.

A random selected edge connected to a slot node of degree ℓ can be revealed with probability $1-p$, after the first iteration, if the total number of unrevealed edges of the slot node is less than or equal to K. Therefore, in case of K-MPR receiver, the functions in Eq. (2.5) and Eq. (2.6) become:

$$\begin{aligned} q &= p^\ell \\ 1-p &= \sum_{k=0}^{min(K,\ell)-1} \binom{\ell-1}{k} q^k (1-q)^{l-k-1} \end{aligned} \tag{3.1}$$

[24] analyzed IRSA with K-MPR for the case of an infinite user population. They show that providing the receiver with multi-user detection capability allows larger load in the systems with low number of slots per frame. By averaging on all possible connected

edges, their density evolution equations in Eq. (3.1) correspond to:

$$f_b(p) = \lambda(p) \text{ and } f_s(q) = 1 - \sum_{k=0}^{K-1} \frac{\rho^{(k)}(1-q)}{k!} q^k \tag{3.2}$$

where $\rho^{(k)}(x)$is the k-th derivative of $\rho(x)$

$$f_s(q) \to F_K(\frac{G}{R}q) \text{ with } F_K(x) \triangleq 1 - e^{-x} \sum_{k=0}^{K-1} \frac{x^k}{k!} \tag{3.3}$$

when $M \to \infty$

A major question is the performance of IRSA and K-IRSA: it is obvious that K-IRSA can at most recover K different users per slots, which thus yields an upper bound of the load threshold of $G^*/K \leq 1$. But: what is the maximum load up to which we can decode all packets G^* and what is the distribution Λ that can reach it? [6] proved that for $K = 1$ (IRSA), the load threshold is asymptotically $G^* \to 1$ for a sequence of (truncated) soliton distributions. In [24], IRSA was studied in an infinite setting, considering that the receiver has K-MPR capability. Simulation results have been done for finite number of users have shown that multiuser detection capability at the receiver allows larger loads on the system with low number of slots per frame, which in turn reduces the total transmission delay and power consumption. Some distributions were experimented with various values of K, and the load threshold was found to be G^*/K between 0.8 and 0.96 (their [24, Fig. 4]).

[2] developed a bound on the asymptotic load threshold of Coded Slotted ALOHA (CSA) schemes with K-MPR, that also applies to K-IRSA. They give examples close to the bound but this was not obtained for plain K-IRSA itself, but a variant using Spatially-Coupled CSA (SC-CSA) [57], with a more structured slot selection.

Their bound [2, (17)] in (3.4) yields $G^*/K \to 1$ when $R \to 0$:

$$\frac{G^*}{K} \leq 1 - \frac{1}{K} e^{-\frac{G^*}{R}} \sum_{k=0}^{K-1} \frac{K-k}{k!} \left(\frac{G^*}{R}\right)^k \tag{3.4}$$

SC-CSA has an interesting performance, and it has other variants such as Irregular Repetition Spatially-Coupled Slotted ALOHA (IRSC-SA), which have also been proposed and analyzed recently in [58]. But all these variants require "super-frames" (i.e. frames of frames), thus typically require several orders of magnitude more slots, hence our continued focus is on K-IRSA[1].

[1]for instance [58, Figure 6] presents simulation results of $L = 40$ super-frames of $\times 400$ slots$= 16,000$ slots. In [58, Table 1] SC-SA performance of 0.9585 with $d = 4$ with $L = 30$ or $L = 40$ super-frames is still well below the result 0.9767 of [57] for super-frames of $L = 200$ sub-frames. Hence, possibly super-frames of $\geq 100,000$ slots in practice

In 2021, a recent work (in progress) [25] has studied some families of repetition probability distribution for K-IRSA (based on an approximation for $G\Lambda'(x)$): they are able to find good distributions, and propose an algorithm to search for the optimal parameters for that family of distributions for K-MPR with $K = 2$. They obtain an optimized distribution in their [25, Eq.(11))] that achieves $G^* = 1.68$. This is still below our own results (see later).

3.2.4 IRSA and K-IRSA (Error Floor): Related Work and Results

In IRSA and K-IRSA, the decoding process is still subject to failure due to *stopping sets* in a similar way to LDPC codes [1], due to the presence of cycles in the Tanner graph (see Fig. 3.5). Stopping sets have a negative impact on the decoder performance and lead to an *error floor* of the decoding failure probability. Previous works including [52] present a finite length analysis of *frameless* ALOHA with K-IRSA in the error floor region. Their recursive approach obtains the decoder state probabilities, allowing to compute numerically the Packet Error Rate (PER), also denoted Packet Loss Rate (PLR). Another analysis of the error floor due to stopping sets of CSA for finite frame lengths over the packet erasure channel based on combinatorics was presented in [39], among others.

3.3 Finding Optimal Distributions

3.3.1 Objectives and Notations

In this section, we provide a method that finds one distribution $\Lambda(x)$ (e.g., to find $\lambda(x)$ first) with the highest load threshold G^* when the number of coefficients is limited by n (e.g. $\Lambda_i = 0, \forall i > n$).We denote the distribution by $\mathcal{L}_n^K$, and the load threshold by $\mathcal{G}_n^K$. Notice that n is also the maximum number of replicas sent by a user and K is the maximum number of colliding packets that could be decoded at the same time by K-MPR a receiver.

3.3.2 Formulating an Optimization Problem

We start from condition (2.12) on $\lambda(x)$, G, and R and apply it to the iteration functions for K-IRSA (3.3). We obtain the following inequality for the case where $K = 2$:

$$F_2(\frac{G}{R}\lambda(x)) < x \implies 1 - e^{-\frac{G}{R}\lambda(x)}(1 + \frac{G}{R}\lambda(x)) < x \tag{3.5}$$

Taking the limit case of equality, we are able to find this closed formula for the function that satisfies the following equality:

$$\lambda(x) = \frac{R}{G} h_2(x) \text{ with } h_2(x) = -W_{-1}\left(\frac{x-1}{e}\right) - 1 \tag{3.6}$$

where $h_2(x)$ is expressed in terms of one of the branches of the Lambert W function. The Lambert function [59] is the inverse of the function $w \mapsto we^w = z$, thus the function $w(z)e^{w(z)} = z$, is a multivalued function defined in general for z complex and assuming complex values of $W(z)$. For $z \in [\frac{-1}{e}, 0[$, there are two real branches, one of which, satisfying $W(z) \leq -1$, is denoted $W_{-1}(z)$, which is used here.

As it satisfies the equality, the function $h_2(x)$ still provides a bound for any Λ distribution. Indeed, any G, λ satisfying (3.5) must satisfy $\lambda(x) < \frac{R}{G} h_2(x)$ equivalently.

This can be generalized for any K:

$$\lambda(x) < \frac{R}{G} h_K(x) \text{ for } x \in]0, 1[\text{ with } h_K(x) \triangleq F_K^{-1}(x) \tag{3.7}$$

Notice that F_K^{-1} cannot be written in closed form for $K > 2$ (or would require "Generalized Lambert functions"), and has the derivative $\frac{dF_K^{-1}}{dx}(x) \to +\infty$ when $x \to 0^+$ for $K \geq 2$.

3.3.3 On the Solutions of the Optimization Problem

One optimal family of solutions for 1-IRSA is based on the soliton distribution [6]. It can be found in the following way (as presented in [60, slide 35]): consider Eq. (3.7), that has a simple expression for $K = 1$. Indeed, $h_1(x) = -\log(1 - x)$. Then, it is possible to write a Taylor series of $h_1(x)$ in $x = 0$ and find an expression for the coefficients of λ_i (and Λ_i) assuming the equality in Eq. (3.7), and recover the soliton distribution(s).

It would seem interesting to apply the same method with $h_2(x)$: It seems that taking the power series of $h_2(x)$, and by truncating it, we should be able to find functions $\lambda(x)$ that correspond to (near) optimal distributions. However, $h_2(x)$ does not have a proper power series as its derivative in 0 is infinite (and behaves like $\alpha\sqrt{x}$ near 0^+).

Similarly, the fact that $\frac{dF_K^{-1}}{dx}(x) \to +\infty$ when $x \to 0^+$ for $K \geq 2$, shows that, in general, unfortunately, one cannot expect to write Taylor series in 0^+, and then easily analytically derive some optimal distributions. Highlighting this difficulty is one of the contributions of this thesis, and it is also the reason why, in the following, we focus on finding optimal distributions numerically.

3.3.4 Finding $\mathcal{L}_n^K$ Through Its Optimal Edge Distribution $\lambda(x)$

We start by fixing n, and search for $\mathcal{L}_n^K$ through its edge distribution $\lambda(x)$. $\lambda(x)$ verifies the bound formulated in (3.7) and provides the largest load threshold G^*: it is an optimization problem.

The conditions on $\lambda(x)$ can be derived from the following:

- The condition $\lambda(1) = 1$ (because it is a probability distribution)
- The property that R is actually expressed in terms of $(\lambda_i)_i$ as: $R = R(\lambda) = \frac{1}{2}\lambda_2 + \frac{1}{3}\lambda_3 + \ldots + \frac{1}{n}\lambda_n$
- The left inequality in (3.7), multiplied by G

This yields the following optimization problem on the variables G, λ_2, λ_3 $\ldots\lambda_n$ (in IRSA and K-IRSA: $\lambda_1 = 0$):

$$
\begin{aligned}
&\underset{(\lambda_i)}{\text{Maximize}} && G && (\mathcal{P}_2)\\
&\text{subject to} && C_1 : 0 \leq \lambda_i \leq 1 \ \forall i\\
& && C_2 : \sum_{i=2}^{i=n} \lambda_i = 1\\
& && C_3 : G \sum_{i=2}^{i=n} \lambda_i x^{i-1} < (\sum_{i=2}^{i=n} \frac{\lambda_i}{i}) h_K(x), \forall x \in]0,1[
\end{aligned}
$$

Now, by using a technique introduced for LDPC codes, the last inequality of $(\mathcal{P}_2)$ can be transformed into multiple inequalities by taking a finite set of values for x, (see for instance [61, Sec. 3.18, p114]). Furthermore, when G is fixed, this is a linear program in the variables $(\lambda_i)_{i=2\ldots n}$, and thus can be efficiently solved.

Hence, our efficient technique is to perform a bisection on G: once G is fixed, we solve the linear program $(\mathcal{P}_2)$ with arbitrary objective function (such as maximize $R(\lambda)$), finding whether the constraints can be satisfied, and if so, finding one λ. The largest G admissible through bisection [2] yields $\mathcal{L}_n^K$ and $\mathcal{G}_n^K$.

We present the pseudo-code of this procedure in Algorithm 1. The algorithm takes as input ϵ, the error tolerance at termination of the bisection search. The initial values $G_{\min} \leftarrow 0$ and $G_{\max} \leftarrow K$ are chosen so that $G_{\min}$ is feasible and $G_{\max}$ is a theoretical upper-bound of $G^*_{p_{\min}}$. The bisection search stops whenever a solution ϵ-close to the optimal is found. This only requires $O(\log_2(1/\epsilon))$ iterations.

[2] Indeed, the method has been proposed for LDPC codes as in [61, Sec. 3.18, p114], but the equivalent of G is considered fixed, hence it was a straightforward linear program. Here we have to introduce this bisection.

Algorithm 1 Load maximization bisection search

input: ϵ

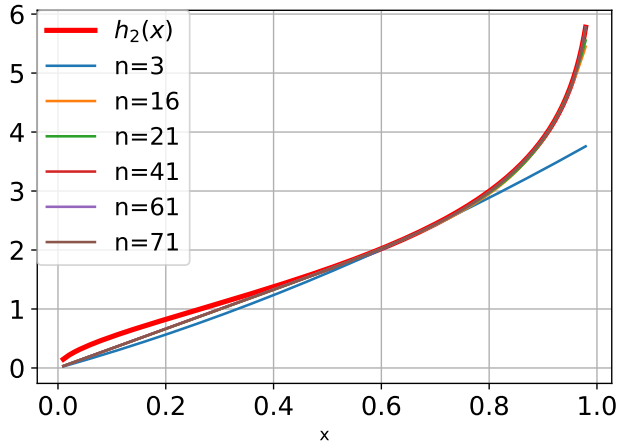
initialization: $G_{\min} \leftarrow 0, G_{\max} \leftarrow K$

while $G_{\max} - G_{\min} > \epsilon$ **do**
  $G \leftarrow (G_{\min} + G_{\max})/2$
  Run a linear solver to check the feasibility of $\mathcal{P}_2$ given a load G
  if $\mathcal{P}_2$ is feasible **then**
    $G_{\min} \leftarrow G$
  else
    $G_{\max} \leftarrow G$
  end if
end while
return $G_{\min}$

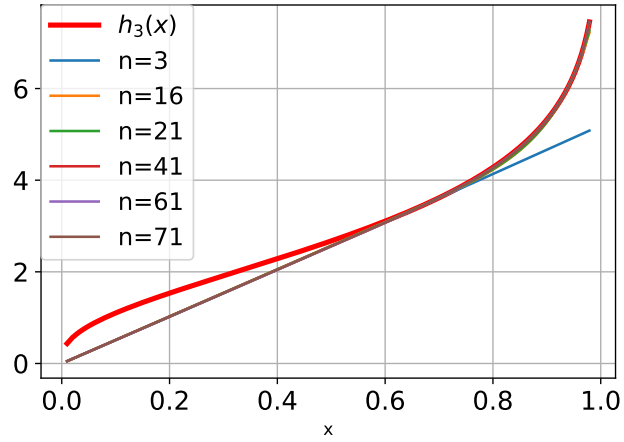
Note that at the end of this chapter, we show in tables Table 3.3, Table 3.4, Table 3.5, Table 3.6 and Table 3.7, some optimized distributions $\mathcal{L}_n^K$ with the corresponding maximum achieved load $\mathcal{G}_n^K$ for $1, 2, 3, 4$ and 5-MPR respectively. We also provide additional optimal distributions for 2-MPR and 3-MPR in case of high-degree coefficient distributions in Table 3.8.

3.3.5 Numerical Results of Optimal Distributions

In this section, we explore the behavior of the optimal $\mathcal{L}_n^K$ numerically. By solving the described problem in $(\mathcal{P}_2)$ which was presented in the previous section, we obtain optimal edge degree distributions for different number of coefficients and different K. The load threshold $\mathcal{G}_n^K$ of these distributions will be analyzed later. But because the distributions are obtained through an optimization program with constraints, most insight can be obtained by considering how tightly the solutions meet these constraints. Ultimately, notice that the primary constraint is derived from the inequality (3.7) that is $\lambda(x) < \frac{R}{G} h_K(x)$. To check for tightness, we equivalently scale (normalize) the $\lambda(x)$ from numerically computed optimal solutions $\mathcal{L}_n^K$, as $\frac{G}{R} \times \lambda(x)$ and compare them to $h_K(x)$, in Fig. 3.2. The y-axis represents the normalized edge degree distribution $h(p) = G\Lambda'(1)\lambda(p)$, while the x-axis represents the different values of p for which the OP was solved. It can be seen from Fig. 3.2a and Fig. 3.2b that as we increase the number of coefficients, we get closer to the bound. But one can notice that, there is always a gap between the bound and the scaled $\lambda(x)$ of optimal distributions, increasing with K ($K = 3$ vs $K = 2$).



(a) $\lambda(x)$ for different 2-MPR optimal distributions $\mathcal{L}_n^2$ for different maximum coefficient n compared with $h_2(x)$



(b) $\lambda(x)$ of 3-MPR optimal distributions $\mathcal{L}_n^3$ for different n compared with $h_3(x)$

Figure 3.2: h_2 and h_3 compared to rescaled $\lambda(x)$

To confirm this effect, Fig. 3.3 directly shows the differences between each such optimal edge distribution and the theoretical bound: $h_K(x) - \frac{G}{R}\lambda(x)$ for 2-MPR; we can see that in the range $x \in [0.6, 1]$, scaled distributions are getting closer to $h_K(x)$ as n increases. However, this is not the case in $[0, 0.6]$, and they seem to converge quickly to a curve above 0 (e.g. for $n > 3$).

Fig 3.2a shows optimal normalized edge degree distributions for 2-MPR case and for different number of coefficients compared with the bound which was found in Eq. (3.5). The y-axis represents the normalized edge degree distribution $G\Lambda'(1)\lambda(p)$, while the x-axis represents the different values of p for which the OP was solved. This graph shows the relation between the probability p and the normalized $\lambda(p)$ where, $q_{normalized} = G\Lambda'(1)\lambda(p)$.

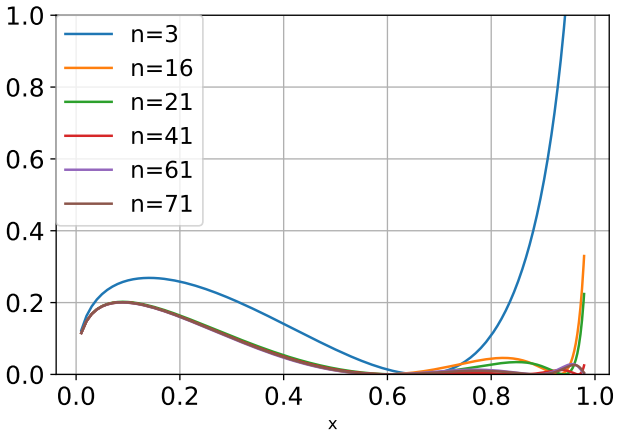


Figure 3.3: For 2-MPR, and for different n: difference between $h_2(x)$ and $\lambda(x)$ of $\mathcal{L}_n^3$: $h_2(x) - \frac{G}{R}\lambda(x)$

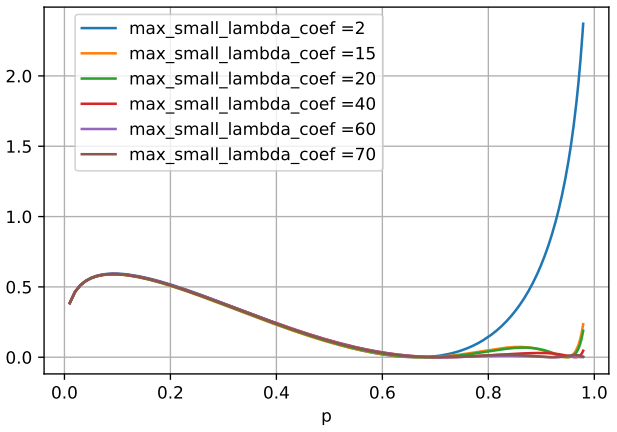


Figure 3.4: Differences between optimal degree distributions and theorical bound for K-MPR=3

Table 3.2 shows some optimal degree distributions for different K-MPR values. Note that the distributions for $K = 1$ are similar to the distributions found in [Table 1, [62]], and the values of G.

3.4 Performance Bound

3.4.1 A New Bound on the Load Threshold of K-IRSA

The empirical results of the previous section made clear (through Fig. 3.3) that there always exists a gap between any scaled $\lambda(x)$ and the function h_K that it should approach, for values x until around $x = 0.5$. One question is whether the existence of this gap can be proven, and

K	n	Distribution $\Lambda(x)$=$\mathcal{L}_n^K$	G=$\mathcal{G}_n^K$
1	4	$0.51988x^2 + 0.48012x^4$	0.8683
1	8	$0.509x^2 + 0.271x^3 + 0.220x^8$	0.9407
1	16	$0.5144x^2 + 0.1827x^3 + 0.1975x^5 + 0.1054x^{16}$	0.9711
2	11	$0.8793x^2 + 0.0003x^7 + 0.1204x^{11}$	1.8992
3	11	$0.929x^2 + 0.071x^{11}$	2.7247
4	11	$0.9514x^2 + 0.0486x^{11}$	3.4889

Table 3.2: Examples of computed optimal distributions

whether it always exists for any Λ, for any K-MPR case. This may answer the proposed question in our introduction: can K-IRSA reach the bound of $G^* = K$?

The two following theorems of this section provide answers.

Theorem 1 *There exist a value x_K, and a function $\theta_K(x)$, such that for any $x \in]0, x_K[$ and λ, R, G satisfying the necessary condition* (3.7)*: the gap between the bound, $h_K(x)$ and any normalized edge degree distribution, $\frac{G}{R}\lambda(x)$ verifies: $h_K(x) - \frac{G}{R}\lambda(x) > \theta_K(x) > 0$*

Proof: Consider the point $M(x_K, y_K = h_K(x_K))$ on the curve $h_K(x)$ such that the tangent of h_K in this point passes through the origin $O(0, 0)$.

The function $h_K(x)$ is always above the tangent up to this point M.

This can be analyzed and proved by introducing the function which computes the gap between h_K and that specific tangent in M:

$$\theta_K(x) \triangleq h_K(x) - h'_K(x_K)x \tag{3.8}$$

This function will pass in zero for the first time, and then in M, i.e:

$$\theta_K(x_K) \triangleq h_K(x_K) - h'_K(x_K)x_K = 0$$

By looking at this function in $O(0, 0)$, we can deduce that $h_K(x)$ is above the tangent near zero, since: $h'_K(x) \to \infty$ for $x \to 0^+$. By combining the two facts: (a). the function $\theta_K(x)$ is positive in zero and (b). the function $\theta_K(x)$ is zero in M, then, it can not be negative in $]0, x_K[$, since it would pass through zero, which is in contradiction with the fact that it passes through zero for the first time in M. Then $\theta_K(x) > 0$ for $0 < x < x_K$.

Consider any valid edge degree distribution $\lambda(x)$ (and associated $G < G^*$, R). λ is always a convex function (as a polynomial with only positive coefficients). Thus, in the range $0 < x < x_K$, $\lambda(x)$ is below the line passing through points $(0, 0)$ to $(x_K, \lambda(x_K))$, which is the tangent passing through points $(0, 0)$ to $(x_K, h_K(x_K))$. Now, considering that: our

bound $h_K(x)$ is always above its tangent in M, and any $\lambda(x)$ is below the tangent passing in $O(0,0)$ and in M, the gap between $h_K(x)$ and $\lambda(x)$ must always exist. This proves the theorem. ■

The theorem also explicitly gives one such function $\theta_K(x)$ in (3.8), and the method to compute x_K, e.g. x_K satisfies:

$$h_K(x_K) = h'_K(x_K)x_K \tag{3.9}$$

Notice that in the general case $K > 2$, h_K has no closed form, but using the definition $h_K = F_K^{-1}$, one can write $h'_K(x) = 1/F'_k(h_K(x))$ and find x_K. For $K = 2$, x_2 is the solution of the following equation (which yields $x_2 = 0.535...$):

$$(x_2 - 1 + e^{W_{-1}(\frac{x_2-1}{e})+1})(W_{-1}(\frac{x_2-1}{e}) + 1) - x_2 = 0 \tag{3.10}$$

One of our main result is then the following theorem:

Theorem 2 *With K-IRSA, and for $K > 0$, the maximum load threshold verifies $G^*/K \leq 1 - \Delta_K$ where $\Delta_K > 0$*

Proof: Consider any $\lambda(x)$ and its associated load threshold G^* and R. For any $G < G^*$, using a similar technique to [2], we compute areas: we define the "gap" as the area between h_K and $\frac{G}{R}\lambda(x)$ as in Fig. 3.3 (and Fig. 3.2a).

$$A_K = \int_{x=0}^{x=1} (h_K(x) - \frac{G}{R}\lambda(x))dx \tag{3.11}$$

First $\int_{x=0}^{x=1} \frac{G}{R}\lambda(x)dx = \frac{G}{R}\left[\frac{1}{\Lambda'(1)}\Lambda(x)dx\right]_0^1 = G\Lambda(1) = G$. Then for $K = 2$, we can directly compute

$$\int_0^1 h_2(x)dx = -\int_0^1 (W_{-1}(\frac{x-1}{e}) + 1)dx = 2$$

while for all K, we can compute the integral considering $h_K = F_K^{-1}$ and computing the integral area from the symmetric graph of F_K instead.

$$\int_0^1 h_K(x)dx = \int_0^\infty (1 - F_K(y))dy = \int_0^\infty e^{-y} \sum_{i=0}^{K-1} \frac{y^i}{i!}dy = \left[-e^{-y}(\sum_{i=0}^{K-1} \frac{(K-i)y^i}{i!})\right]_0^\infty = K$$

Hence $A_K = K - G$. On the other hand, the inequality from theorem 1, implies:

$$A_K \geq \int_{x=0}^{x=x_K} (h_K(x) - \frac{G}{R}\lambda(x))dx > \int_0^{x_K} \theta_K(x)dx$$

Combining the two, we obtain:

$$G/K < 1 - \Delta_K \text{ with } \Delta_K \triangleq \frac{1}{K}\int_0^{x_K} \theta_K(x)dx \tag{3.12}$$

true for any λ and valid G, which proves the theorem. ■

Theorem 1 and Theorem 2 provide a way to compute Δ_K. First compute x_K through equation (3.9), then Δ_K with (3.12) using the definition of θ_K in (3.8) simplified with (3.9). For instance for $K = 2$, we obtain $2\Delta_2 = 0.0553..$, hence a bound $G^* \leq 1.9448$

3.4.2 Error Floor Approximation

In this section, we propose approximations of the error floor of K-IRSA, that is, the PLR persists even when the load is low. Although the Multiple Packet Reception capability at the receiver can enhance the performance of IRSA, the decoding process is still subject to failure due to stopping sets similarly to LDPC codes, where a finite frame length scheme can impose an error floor due to the presence of stopping sets (cycles) in the graph. This results that the PLR steadily decreases in the form of a curve as the G^*/K condition becomes better.

Compared to [52], we stay in the framed version of K-IRSA, and obtain closed formulas, and compared to [39], we consider K-IRSA (not just IRSA), and our expressions can be simpler through further "urn and ball" approximations [63].

We start from their observations: in K-IRSA, at low channel loads, a most probable stopping set occurs when at least $K + 1$, degree-2 users send their packets on the same two slots (see [39, 52] and others). The probability of having such a stopping set is the same as having at least $K + 1$ users in the same "virtual" square in Fig. 3.5.(b), where each square represents the choice of two unordered slots. The total number of all possible choices of two unordered slots for each user (number of squares) is $m = \binom{N}{2}$.

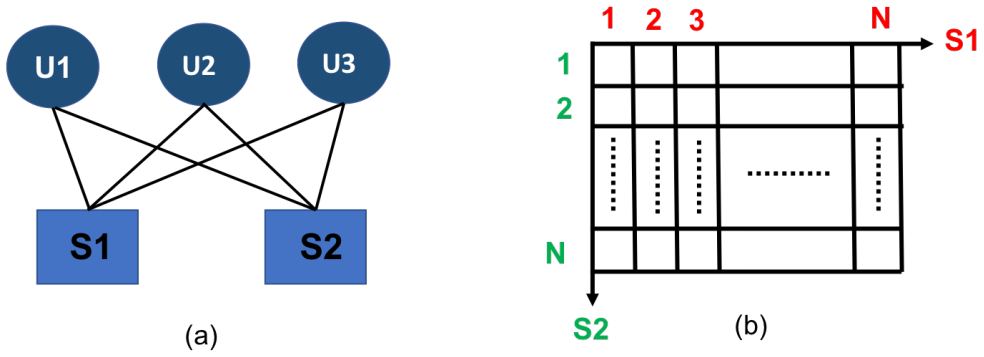


Figure 3.5: Most probable stopping set for 2-MPR (left)

The problem of user distribution in the squares corresponds to the classic urns and balls problem(s) in the probability literature [63]. The squares can be seen as urns, and the users can be seen as balls randomly thrown in urns. With M^* users, the probability of having m_r squares (urns) such that each square contains exactly r users (collisions) is given by [63, pp. 114 –116]:

$$Pr\left[M_r = m_r\right] = \sum_{k=0}^{m-m_r} (-1)^k \binom{m_r + k}{m_r} S_{m_r+k}$$

where:

$$S_k = \binom{m}{k} \frac{M^*!}{(r!)^k (M^* - kr)!} \frac{(m-k)^{M^*-kr}}{m^{M^*}}$$

These expressions are, in general, difficult to evaluate. In the following, we use well-known approximations, considering limiting cases such as $m \to \infty$ and $M^* \to \infty$. [63, pp. 315–320] has surveyed results for the approximation of the distribution of M_r with $r \geq 2$. We use the notation of [63] but except denoting their λ as ℓ, and their M as M^*, applied to our problem:

$$\alpha \triangleq \frac{M^*}{m} = \frac{2M^*}{N(N-1)}; a_r \triangleq \frac{\alpha^r}{r!} e^{-\alpha} \text{ and } \ell_r \triangleq m a_r \tag{3.13}$$

In our case, only users transmitting exactly two replicas are considered for stopping sets, so $M^* \approx \Lambda_2 M$. And $M^* \to \infty$, $\alpha \to 0$ (at fixed load $g^* = M^*/N$) and $\ell_r \to 0$ (hence bounded). [63, Table 6.1 p320] has reproduced a summary of asymptotic distributions of M_r depending on α and ℓ_r (quoting Kolchin), and for us, the case is: M_r has an asymptotic Poisson distribution with parameter ℓ_r. Then, the packet loss rate (PLR) of all users is approximated as:

$$\text{PLR} = \frac{1}{M} \sum_{r=K+1}^{r=\infty} \sum_{u=1}^{u=\infty} ur \Pr[M_r = u] \approx \sum_{r=K+1}^{r=\infty} \sum_{u=1}^{u=\infty} \frac{ur}{M} \frac{\ell_r^u e^{-\ell_r}}{u!} \tag{3.14}$$

At low loads, a simpler expression is obtained by considering only the dominant term, e.g., for $r = K + 1$, $u = 1$:

$$\text{PLR} \approx \frac{(K+1)\ell_{K+1}}{M} e^{-\ell_{K+1}} = \frac{\Lambda_2 \alpha^K}{K!} e^{-\alpha} e^{-\ell_{K+1}} \tag{3.15}$$

3.5 Experimental Results and Numerical Insights

Our first focus is the asymptotic performance of K-IRSA on the normalized load threshold G^*/K, which is bounded absolutely by 1. Remember that, it is a metric of interest because it represents the limit up to which K-IRSA can be used with vanishing packet losses (when $M \to \infty$). Thus, it also corresponds to the maximum throughput of K-IRSA as "decoded packet/slot" with vanishing packet losses. To understand the performance, for $K = 2, 3, 4, 5$, we varied the maximum degree of Λ: $n = 2, \ldots 100$. Then, we computed, as previously, each optimal distribution $\mathcal{L}_n^K$ with the same method as in Section 3.3, and its associated load threshold G^* and rate R. In Fig. 3.6, we represent the numerical results of the normalized load G^*/K for these distributions $\mathcal{L}_n^K$ as plain lines with points. We also represent the bound of [2] computed from Eq. (3.4) as dotted lines (from the rate also in Fig. 3.6). Finally, we represent our bound: $1 - \Delta_k$, which was computed as in Section 3.4. It appears as thick horizontal lines in Fig. 3.6.

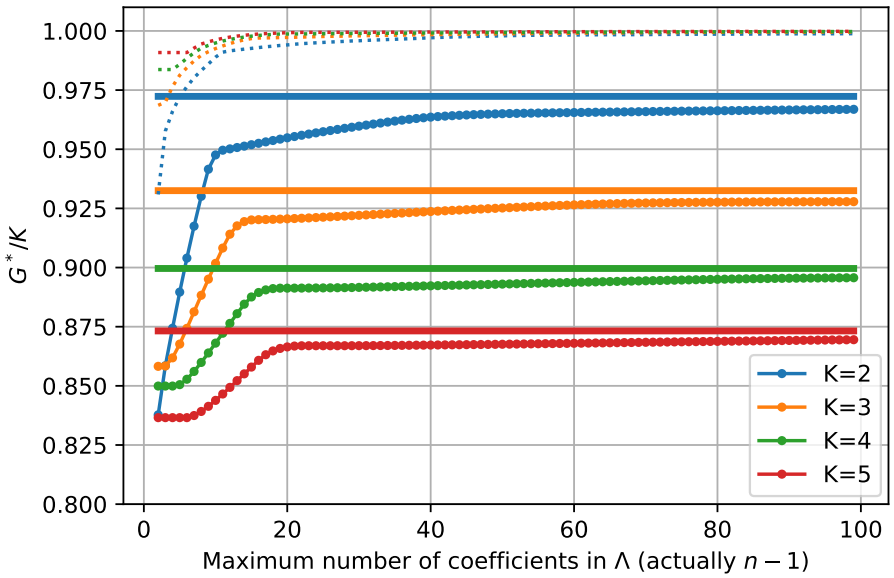


Figure 3.6: *Normalized load threshold G^*/K; dotted lines represent optimal G^*/K from $\mathcal{L}_n^K$, that are optimal distributions Λ (with max. degree n, which varies on the x-axis); thick horizontal lines represent our bound $1 - \Delta_K$ from theorem 1; dotted lines near line $y = 1$ represent the bound of [2] with the rate R of $\mathcal{L}_n^K$*

We are then able to observe that: first, the known bound Eq. (3.4) is useful for higher rates as found in CSA for which it was first introduced, here, becomes quickly too loose for the optimal $\mathcal{L}_n^K$. Indeed, the dotted lines representing it are close to 1 for $n \geq 20$. Our bound $1 - \Delta_K$ is closer to the actual performance G^*/K of the optimal distributions $\mathcal{L}_n^K$, and thus captures well their asymptotic behavior. Finally, one can observe that the performance of K-IRSA gradually decreases as K increases, and moves away from 1, contrary to what could have been previously thought. Computing for large $K = 50$, we get $1 - \Delta_{50} = 0.6555....$ An open question is whether that new bound is asymptotically reached by $\mathcal{L}_n^K$?. Another open question is related to the rate R: $G^*/K \to 1$ implied $R \to 0$, but we proved that $G^*/K \not\to 1$ for K-IRSA, hence this might be revisited.

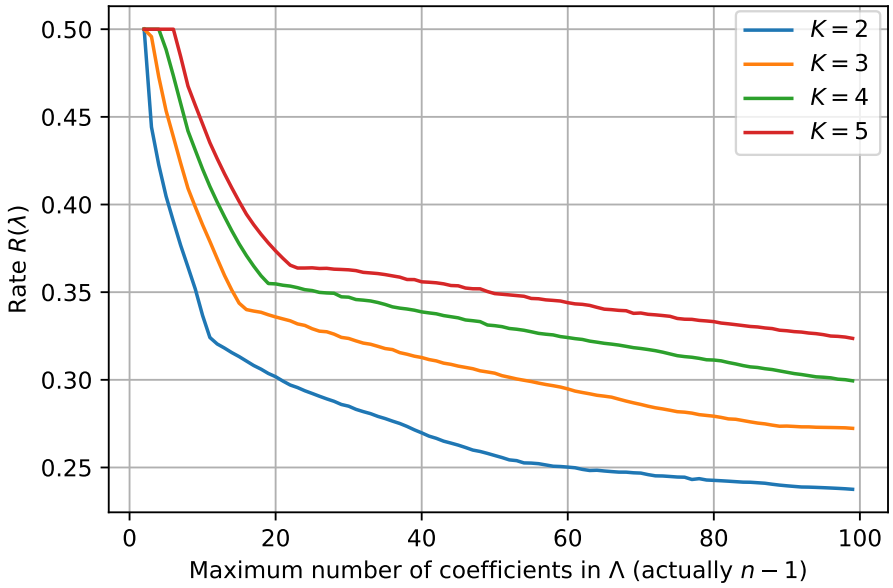


Figure 3.7: *Rate R of optimal distribution $\mathcal{L}_n^K$ when max. degree n increases*

Fig. 3.6 compares the optimal load G^* with the number of coefficients used in the edge distribution for different cases of K-MPR. It is obvious that using a high degree edge distribution for any case of K-MPR will increase the maximum load and this is due to the fact that $G = \frac{\Psi'(1)}{\Lambda'(1)} = \frac{\Psi'(1)}{\sum \frac{\lambda_i}{i}}$, so that when we increase the number of coefficients, the average number of replicas (i.e. the rate) will increase, which yields in the increasing of the load. Note that in [25], the maximum achieved load for 2-MPR case is 1.68, hence, $G^*/K = 0.84$, which is less from our obtained results for 2-MPR case.

Fig. 3.7, shows the evolution of the rate R of $\mathcal{L}_n^K$ with n for different K-MPR cases. Each curve shows the same behavior of decreasing when the number of coefficients is increasing. It is unclear whether the rate R converges towards 0. Still, we conjecture it is

the case (based on observing the last inequality of the optimization problem ($\mathcal{P}_2$) and the constraints on λ_i).

Notice also that the (relatively) higher rate of the distributions for higher K, implies better energy-efficiency.

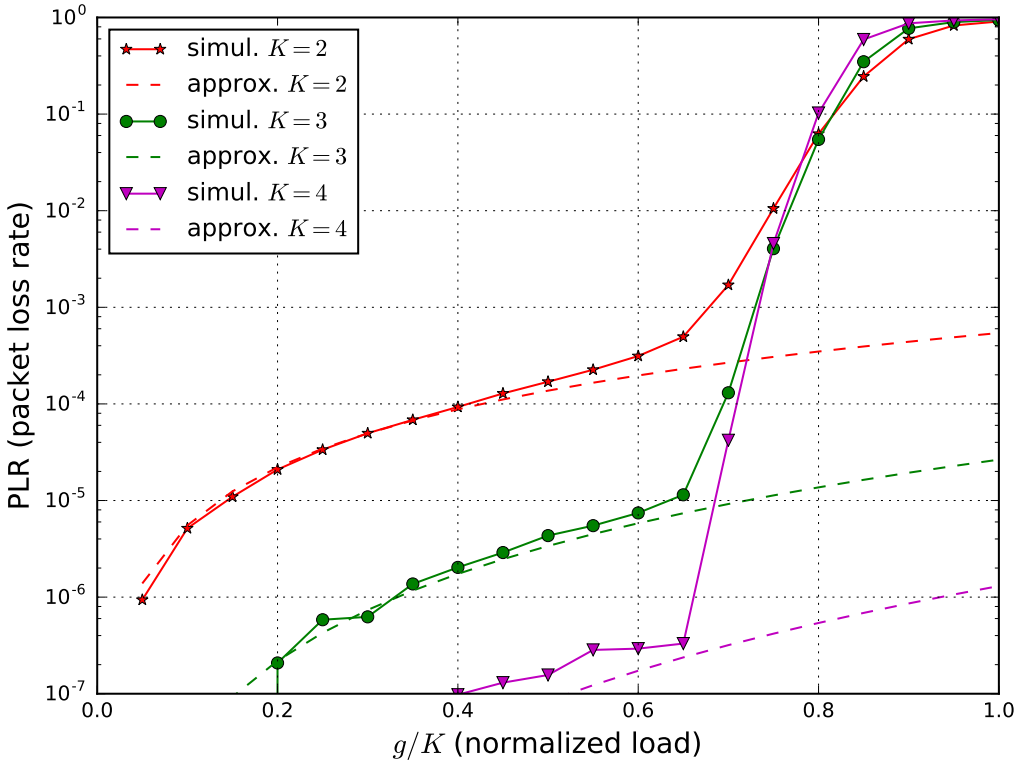


Figure 3.8: Comparing between the error floor for different KMPR cases obtained by simulations and the asymptotic error floor

The previous results focused on the performance of K-IRSA in terms of throughput, thus G/K^*. Additionally, we implemented a K-IRSA simulator in Python, validated it through density evolution (finding similar curves to the literature such as in [1]). Then, we have studied the probability for one packet not to be decoded (packet loss rate, PLR). We selected a frame of size $M = 100$, different values of K, using optimal distributions of degree 11 from Table 3.2 ($\mathcal{L}^2_{11}$, $\mathcal{L}^3_{11}$, $\mathcal{L}^4_{11}$), and varied the normalized load G/K. The simulation results are in Fig. 3.8 (with 10,000 to 640,000 simulations per point) with also corresponding plots of our approximation Eq. (3.14). First, as one can see, the approximation is very close to the actual performance at low load (should be a lower bound). Further, for all data points of our simulated scenarios, comparing numerically Eq. (3.14) to Eq. (3.15), we observed a relative difference of less than 5%; and we also observed $\alpha < 0.0769$, and $\ell_{K+1} < 0.0357$ (thus $e^{-\alpha} > 0.92$ and $e^{-\ell_{K+1}} > 0.96$).

Then, at lower loads, the PLR itself appears to decrease by one or two orders of magnitude as K increases. Indeed, the event of "$K+1$ or more users selecting the same 3 slots" has a very small probability to happen at low loads, and this is the reason why we considered only degree-2 users in our PLR analysis. This dramatic decrease of PLR shows that, for reliable communications, K-IRSA with $K > 1$ is a good choice. Eq. (3.15) helps to understand why: with $e^{-\alpha}$ and $e^{-\ell_{K+1}}$ close to 1, almost all variation in Eq. (3.15) will come from the other factors. Keeping G fixed, assuming Λ_2 does not vary (too much), we observe: (a) an increase of K by 1 yields a multiplication of the PLR by a factor $\frac{\alpha}{K+1}$ essentially, hence a strong decrease knowing that $\alpha \ll 1$; and also (b) a relative increase of the frame size N by a factor of $\eta > 1$ (with an identical increase of the number of users), leads to a division of the PLR by a factor $\approx \eta^K$. Both facts are notable for design.

Finally, note a different link between the low(er) error floor of K-IRSA with the fact that K-IRSA cannot reach the bound $G^*/K = 1$ (for $K > 1$): the fact that at low loads K users are unlikely to be undecoded is linked to the fact that $f_s(q)$ from Eq. (3.2), and as a consequence F_K, quickly approaches 0 for q near 0, e.g. $O(x^2)$. In turn, it results in an infinite derivative in 0^+ for $h_K = F_K^{-1}$. The infinite derivative causes concavity and the impossibility to tightly approach it with a polynomial with positive coefficients and a bound $G^*/K < 1$. Thus, this presents a tradeoff between throughput and reliability.

3.6 Conclusion

In this chapter, K-IRSA was studied. It is an essential aspect of investigating how to improve IRSA performance given that modern transmissions methods can increase receiver diversity and provide for an equivalent of the K-MPR ability. Additionally, the optimal performance of K-IRSA is not as well known/investigated as the optimal performance of IRSA.

We presented inequalities that should be verified by K-IRSA degree distributions, and provided some insights about why it is more difficult to find analytically optimal ones for $K \geq 2$. We presented a method to compute optimal K-IRSA degree distributions $\mathcal{L}_n^K$ with a given maximum degree n based on these inequalities.

A new, tighter bound Eq. (3.12) for the load threshold (G^*/K) was proven, showing that plain K-IRSA cannot reach the known asymptotic bound $G^*/K = 1$ for $K > 1$. Open questions are whether the tighter bound Eq. (3.12) can be reached and whether the rate R converges towards 0. Numerical results illustrate that optimal degree distributions tend to approach this bound. We also analyzed the error floor behavior of K-IRSA and provided

an insightful approximation Eq. (3.15) of the PLR at low loads, and showing its excellent performance, and giving some insights on how PLR decreases with K or with an increase of the frame size.

$K = 1$	$\mathcal{L}_2^{(1)}$	$1.0000x^2$	$\mathcal{G}_1^{(2)} = 0.5025$
$K = 1$	$\mathcal{L}_3^{(1)}$	$0.1850x^2 + 0.8150x^3$	$\mathcal{G}_1^{(3)} = 0.8240$
$K = 1$	$\mathcal{L}_4^{(1)}$	$0.5199x^2 + 0.4801x^4$	$\mathcal{G}_1^{(4)} = 0.8683$
$K = 1$	$\mathcal{L}_5^{(1)}$	$0.5589x^2 + 0.0588x^3 + 0.3823x^5$	$\mathcal{G}_1^{(5)} = 0.8977$
$K = 1$	$\mathcal{L}_6^{(1)}$	$0.5465x^2 + 0.1662x^3 + 0.2873x^6$	$\mathcal{G}_1^{(6)} = 0.9153$
$K = 1$	$\mathcal{L}_7^{(1)}$	$0.5322x^2 + 0.2230x^3 + 0.2448x^7$	$\mathcal{G}_1^{(7)} = 0.9301$
$K = 1$	$\mathcal{L}_8^{(1)}$	$0.5094x^2 + 0.2708x^3 + 0.2198x^8$	$\mathcal{G}_1^{(8)} = 0.9407$
$K = 1$	$\mathcal{L}_9^{(1)}$	$0.4833x^2 + 0.3144x^3 + 0.2023x^9$	$\mathcal{G}_1^{(9)} = 0.9479$
$K = 1$	$\mathcal{L}_{10}^{(1)}$	$0.4936x^2 + 0.2515x^3 + 0.0782x^4 + 0.1766x^{10}$	$\mathcal{G}_1^{(10)} = 0.9528$
$K = 1$	$\mathcal{L}_{20}^{(1)}$	$0.5039x^2 + 0.2179x^3 + 0.0335x^5 + 0.1620x^6 + 0.0827x^{20}$	$\mathcal{G}_1^{(20)} = 0.9771$
$K = 1$	$\mathcal{L}_{30}^{(1)}$	$0.5079x^2 + 0.1619x^3 + 0.1072x^4 + 0.0493x^5 + 0.0354x^8 + 0.0839x^9 + 0.0544x^{30}$	$\mathcal{G}_1^{(30)} = 0.9847$
$K = 1$	$\mathcal{L}_{40}^{(1)}$	$0.5039x^2 + 0.1878x^3 + 0.0108x^4 + 0.1603x^5 + 0.0693x^{11} + 0.0278x^{12} + 0.0401x^{40}$	$\mathcal{G}_1^{(40)} = 0.9886$
$K = 1$	$\mathcal{L}_{50}^{(1)}$	$0.5046x^2 + 0.1663x^3 + 0.1012x^4 + 0.0764x^6 + 0.0467x^7 + 0.0511x^{14} + 0.0219x^{15} + 0.0318x^{50}$	$\mathcal{G}_1^{(50)} = 0.9909$
$K = 1$	$\mathcal{L}_{60}^{(1)}$	$0.5039x^2 + 0.1656x^3 + 0.0947x^4 + 0.0440x^5 + 0.0572x^7 + 0.0470x^8 + 0.0561x^{17} + 0.0054x^{18} + 0.0262x^{60}$	$\mathcal{G}_1^{(60)} = 0.9924$
$K = 1$	$\mathcal{L}_{70}^{(1)}$	$0.5031x^2 + 0.1701x^3 + 0.0688x^4 + 0.0908x^5 + 0.0325x^8 + 0.0606x^9 + 0.0517x^{20} + 0.0224x^{70}$	$\mathcal{G}_1^{(70)} = 0.9935$
$K = 1$	$\mathcal{L}_{80}^{(1)}$	$0.5023x^2 + 0.1746x^3 + 0.0514x^4 + 0.1135x^5 + 0.0577x^9 + 0.0342x^{10} + 0.0200x^{22} + 0.0268x^{23} + 0.0196x^{80}$	$\mathcal{G}_1^{(80)} = 0.9943$
$K = 1$	$\mathcal{L}_{100}^{(1)}$	$0.5023x^2 + 0.1658x^3 + 0.0984x^4 + 0.1023x^6 + 0.0073x^7 + 0.0183x^{11} + 0.0532x^{12} + 0.0264x^{28} + 0.0106x^{29} + 0.0155x^{100}$	$\mathcal{G}_1^{(100)} = 0.9955$

Table 3.3: Optimized repetition degree distributions for 1-MPR with the associated maximum achieved load

$K = 2$	$\mathcal{L}_2^{(2)}$	$1.0000x^2$	$\mathcal{G}_2^{(2)} = 1.6755$
$K = 2$	$\mathcal{L}_3^{(2)}$	$0.7487x^2 + 0.2513x^3$	$\mathcal{G}_2^{(3)} = 1.7170$
$K = 2$	$\mathcal{L}_4^{(2)}$	$0.8170x^2 + 0.1830x^4$	$\mathcal{G}_2^{(4)} = 1.7487$
$K = 2$	$\mathcal{L}_5^{(2)}$	$0.8430x^2 + 0.1570x^5$	$\mathcal{G}_2^{(5)} = 1.7793$
$K = 2$	$\mathcal{L}_6^{(2)}$	$0.8597x^2 + 0.1403x^6$	$\mathcal{G}_2^{(6)} = 1.8080$
$K = 2$	$\mathcal{L}_7^{(2)}$	$0.8690x^2 + 0.1310x^7$	$\mathcal{G}_2^{(7)} = 1.8349$
$K = 2$	$\mathcal{L}_8^{(2)}$	$0.8759x^2 + 0.1241x^8$	$\mathcal{G}_2^{(8)} = 1.8602$
$K = 2$	$\mathcal{L}_9^{(2)}$	$0.8796x^2 + 0.1203x^9$	$\mathcal{G}_2^{(9)} = 1.8831$
$K = 2$	$\mathcal{L}_{10}^{(2)}$	$0.8788x^2 + 0.1212x^{10}$	$\mathcal{G}_2^{(10)} = 1.8952$
$K = 2$	$\mathcal{L}_{20}^{(2)}$	$0.8650x^2 + 0.0930x^8 + 0.0420x^{20}$	$\mathcal{G}_2^{(20)} = 1.9097$
$K = 2$	$\mathcal{L}_{30}^{(2)}$	$0.8606x^2 + 0.0484x^8 + 0.0633x^9 + 0.0277x^{30}$	$\mathcal{G}_2^{(30)} = 1.9195$
$K = 2$	$\mathcal{L}_{40}^{(2)}$	$0.8596x^2 + 0.1167x^9 + 0.0003x^{10} + 0.0234x^{40}$	$\mathcal{G}_2^{(40)} = 1.9271$
$K = 2$	$\mathcal{L}_{50}^{(2)}$	$0.8606x^2 + 0.0630x^9 + 0.0554x^{10} + 0.0210x^{50}$	$\mathcal{G}_2^{(50)} = 1.9300$
$K = 2$	$\mathcal{L}_{60}^{(2)}$	$0.8602x^2 + 0.0633x^9 + 0.0537x^{10} + 0.0051x^{28} + 0.0010x^{29} + 0.0166x^{60}$	$\mathcal{G}_2^{(60)} = 1.9310$
$K = 2$	$\mathcal{L}_{70}^{(2)}$	$0.8588x^2 + 0.0934x^9 + 0.0220x^{10} + 0.0131x^{29} + 0.0128x^{70}$	$\mathcal{G}_2^{(70)} = 1.9318$
$K = 2$	$\mathcal{L}_{80}^{(2)}$	$0.8584x^2 + 0.0939x^9 + 0.0212x^{10} + 0.0090x^{30} + 0.0065x^{31} + 0.0110x^{80}$	$\mathcal{G}_2^{(80)} = 1.9325$
$K = 2$	$\mathcal{L}_{90}^{(2)}$	$0.8573x^2 + 0.1145x^9 + 0.0062x^{30} + 0.0125x^{31} + 0.0095x^{90}$	$\mathcal{G}_2^{(90)} = 1.9332$
$K = 2$	$\mathcal{L}_{100}^{(2)}$	$0.8569x^2 + 0.1148x^9 + 0.0014x^{31} + 0.0183x^{32} + 0.0085x^{100}$	$\mathcal{G}_2^{(100)} = 1.9338$

Table 3.4: Optimized repetition degree distributions for 2-MPR with the associated maximum achieved load

$K = 3$	$\mathcal{L}_2^{(3)}$	$1.0000x^2$	$\mathcal{G}_3^{(2)} = 2.5747$
$K = 3$	$\mathcal{L}_3^{(3)}$	$0.9830x^2 + 0.0170x^3$	$\mathcal{G}_3^{(3)} = 2.5752$
$K = 3$	$\mathcal{L}_4^{(3)}$	$0.9425x^2 + 0.0575x^4$	$\mathcal{G}_3^{(4)} = 2.5855$
$K = 3$	$\mathcal{L}_5^{(3)}$	$0.9317x^2 + 0.0683x^5$	$\mathcal{G}_3^{(5)} = 2.6030$
$K = 3$	$\mathcal{L}_6^{(3)}$	$0.9300x^2 + 0.0700x^6$	$\mathcal{G}_3^{(6)} = 2.6229$
$K = 3$	$\mathcal{L}_7^{(3)}$	$0.9278x^2 + 0.0722x^7$	$\mathcal{G}_3^{(7)} = 2.6439$
$K = 3$	$\mathcal{L}_8^{(3)}$	$0.9261x^2 + 0.0739x^8$	$\mathcal{G}_3^{(8)} = 2.6648$
$K = 3$	$\mathcal{L}_9^{(3)}$	$0.9273x^2 + 0.0727x^9$	$\mathcal{G}_3^{(9)} = 2.6853$
$K = 3$	$\mathcal{L}_{10}^{(3)}$	$0.9282x^2 + 0.0718x^{10}$	$\mathcal{G}_3^{(10)} = 2.7053$
$K = 3$	$\mathcal{L}_{20}^{(3)}$	$0.9280x^2 + 0.0248x^{12} + 0.0171x^{13} + 0.0301x^{20}$	$\mathcal{G}_3^{(20)} = 2.7618$
$K = 3$	$\mathcal{L}_{30}^{(3)}$	$0.9238x^2 + 0.0580x^{12} + 0.0182x^{30}$	$\mathcal{G}_3^{(30)} = 2.7662$
$K = 3$	$\mathcal{L}_{40}^{(3)}$	$0.9212x^2 + 0.0582x^{12} + 0.0062x^{13} + 0.0144x^{40}$	$\mathcal{G}_3^{(40)} = 2.7710$
$K = 3$	$\mathcal{L}_{50}^{(3)}$	$0.9201x^2 + 0.0385x^{12} + 0.0292x^{13} + 0.0122x^{50}$	$\mathcal{G}_3^{(50)} = 2.7755$
$K = 3$	$\mathcal{L}_{60}^{(3)}$	$0.9190x^2 + 0.0261x^{12} + 0.0437x^{13} + 0.0112x^{60}$	$\mathcal{G}_3^{(60)} = 2.7794$
$K = 3$	$\mathcal{L}_{70}^{(3)}$	$0.9195x^2 + 0.0672x^{13} + 0.0027x^{14} + 0.0107x^{70}$	$\mathcal{G}_3^{(70)} = 2.7819$
$K = 3$	$\mathcal{L}_{80}^{(3)}$	$0.9200x^2 + 0.0378x^{13} + 0.0322x^{14} + 0.0100x^{80}$	$\mathcal{G}_3^{(80)} = 2.7829$
$K = 3$	$\mathcal{L}_{90}^{(3)}$	$0.9202x^2 + 0.0293x^{13} + 0.0406x^{14} + 0.0007x^{52} + 0.0092x^{90}$	$\mathcal{G}_3^{(90)} = 2.7833$
$K = 3$	$\mathcal{L}_{100}^{(3)}$	$0.9201x^2 + 0.0307x^{13} + 0.0389x^{14} + 0.0032x^{55} + 0.0072x^{100}$	$\mathcal{G}_3^{(100)} = 2.7836$

Table 3.5: Optimized repetition degree distributions for 3-MPR with the associated maximum achieved load

$K = 4$	$\mathcal{L}_2^{(4)}$	$1.0000x^2$	$\mathcal{G}_4^{(2)} = 3.3997$
$K = 4$	$\mathcal{L}_3^{(4)}$	$1.0000x^2$	$\mathcal{G}_4^{(3)} = 3.3997$
$K = 4$	$\mathcal{L}_4^{(4)}$	$1.0000x^2$	$\mathcal{G}_4^{(4)} = 3.3997$
$K = 4$	$\mathcal{L}_5^{(4)}$	$0.9841x^2 + 0.0159x^5$	$\mathcal{G}_4^{(5)} = 3.4020$
$K = 4$	$\mathcal{L}_6^{(4)}$	$0.9719x^2 + 0.0281x^6$	$\mathcal{G}_4^{(6)} = 3.4112$
$K = 4$	$\mathcal{L}_7^{(4)}$	$0.9629x^2 + 0.0371x^7$	$\mathcal{G}_4^{(7)} = 3.4244$
$K = 4$	$\mathcal{L}_8^{(4)}$	$0.9562x^2 + 0.0438x^8$	$\mathcal{G}_4^{(8)} = 3.4396$
$K = 4$	$\mathcal{L}_9^{(4)}$	$0.9543x^2 + 0.0457x^9$	$\mathcal{G}_4^{(9)} = 3.4556$
$K = 4$	$\mathcal{L}_{10}^{(4)}$	$0.9525x^2 + 0.0475x^{10}$	$\mathcal{G}_4^{(10)} = 3.4722$
$K = 4$	$\mathcal{L}_{20}^{(4)}$	$0.9515x^2 + 0.0269x^{18} + 0.0216x^{20}$	$\mathcal{G}_4^{(20)} = 3.5652$
$K = 4$	$\mathcal{L}_{30}^{(4)}$	$0.9496x^2 + 0.0379x^{16} + 0.0125x^{30}$	$\mathcal{G}_4^{(30)} = 3.5664$
$K = 4$	$\mathcal{L}_{40}^{(4)}$	$0.9484x^2 + 0.0421x^{16} + 0.0096x^{40}$	$\mathcal{G}_4^{(40)} = 3.5691$
$K = 4$	$\mathcal{L}_{50}^{(4)}$	$0.9465x^2 + 0.0189x^{15} + 0.0260x^{16} + 0.0086x^{50}$	$\mathcal{G}_4^{(50)} = 3.5720$
$K = 4$	$\mathcal{L}_{60}^{(4)}$	$0.9459x^2 + 0.0083x^{15} + 0.0381x^{16} + 0.0076x^{60}$	$\mathcal{G}_4^{(60)} = 3.5749$
$K = 4$	$\mathcal{L}_{70}^{(4)}$	$0.9453x^2 + 0.0003x^{15} + 0.0474x^{16} + 0.0070x^{70}$	$\mathcal{G}_4^{(70)} = 3.5776$
$K = 4$	$\mathcal{L}_{80}^{(4)}$	$0.9443x^2 + 0.0038x^{15} + 0.0451x^{16} + 0.0068x^{80}$	$\mathcal{G}_4^{(80)} = 3.5801$
$K = 4$	$\mathcal{L}_{90}^{(4)}$	$0.9443x^2 + 0.0387x^{16} + 0.0103x^{17} + 0.0067x^{90}$	$\mathcal{G}_4^{(90)} = 3.5819$
$K = 4$	$\mathcal{L}_{100}^{(4)}$	$0.9446x^2 + 0.0183x^{16} + 0.0306x^{17} + 0.0064x^{100}$	$\mathcal{G}_4^{(100)} = 3.5828$

Table 3.6: Optimized repetition degree distributions for 4-MPR with the associated maximum achieved load

$K = 5$	$\mathcal{L}_2^{(5)}$	$1.0000x^2$	$\mathcal{G}_5^{(2)} = 4.1828$
$K = 5$	$\mathcal{L}_3^{(5)}$	$1.0000x^2$	$\mathcal{G}_5^{(3)} = 4.1828$
$K = 5$	$\mathcal{L}_4^{(5)}$	$1.0000x^2$	$\mathcal{G}_5^{(4)} = 4.1828$
$K = 5$	$\mathcal{L}_5^{(5)}$	$1.0000x^2$	$\mathcal{G}_5^{(5)} = 4.1828$
$K = 5$	$\mathcal{L}_6^{(5)}$	$1.0000x^2$	$\mathcal{G}_5^{(6)} = 4.1828$
$K = 5$	$\mathcal{L}_7^{(5)}$	$0.9871x^2 + 0.0129x^7$	$\mathcal{G}_5^{(7)} = 4.1873$
$K = 5$	$\mathcal{L}_8^{(5)}$	$0.9769x^2 + 0.0231x^8$	$\mathcal{G}_5^{(8)} = 4.1959$
$K = 5$	$\mathcal{L}_9^{(5)}$	$0.9729x^2 + 0.0271x^9$	$\mathcal{G}_5^{(9)} = 4.2067$
$K = 5$	$\mathcal{L}_{10}^{(5)}$	$0.9697x^2 + 0.0303x^{10}$	$\mathcal{G}_5^{(10)} = 4.2193$
$K = 5$	$\mathcal{L}_{20}^{(5)}$	$0.9624x^2 + 0.0001x^{19} + 0.0375x^{20}$	$\mathcal{G}_5^{(20)} = 4.3323$
$K = 5$	$\mathcal{L}_{30}^{(5)}$	$0.9629x^2 + 0.0312x^{21} + 0.0058x^{30}$	$\mathcal{G}_5^{(30)} = 4.3349$
$K = 5$	$\mathcal{L}_{40}^{(5)}$	$0.9619x^2 + 0.0121x^{19} + 0.0192x^{20} + 0.0068x^{40}$	$\mathcal{G}_5^{(40)} = 4.3362$
$K = 5$	$\mathcal{L}_{50}^{(5)}$	$0.9608x^2 + 0.0328x^{19} + 0.0064x^{50}$	$\mathcal{G}_5^{(50)} = 4.3379$
$K = 5$	$\mathcal{L}_{60}^{(5)}$	$0.9602x^2 + 0.0342x^{19} + 0.0056x^{60}$	$\mathcal{G}_5^{(60)} = 4.3399$
$K = 5$	$\mathcal{L}_{70}^{(5)}$	$0.9593x^2 + 0.0099x^{18} + 0.0254x^{19} + 0.0054x^{70}$	$\mathcal{G}_5^{(70)} = 4.3420$
$K = 5$	$\mathcal{L}_{80}^{(5)}$	$0.9589x^2 + 0.0043x^{18} + 0.0318x^{19} + 0.0050x^{80}$	$\mathcal{G}_5^{(80)} = 4.3440$
$K = 5$	$\mathcal{L}_{90}^{(5)}$	$0.9585x^2 + 0.0365x^{19} + 0.0048x^{90}$	$\mathcal{G}_5^{(90)} = 4.3459$
$K = 5$	$\mathcal{L}_{100}^{(5)}$	$0.9580x^2 + 0.0373x^{19} + 0.0047x^{100}$	$\mathcal{G}_5^{(100)} = 4.3477$

Table 3.7: Optimized repetition degree distributions for 5-MPR with the associated maximum achieved load

$K = 2$	$\mathcal{L}^{(2)}_{100}$	$0.8569x^2 + 0.1148x^9 + 0.0014x^{31} + 0.0183x^{32} + 0.0085x^{100}$	$\mathcal{G}^{(100)}_2 = 1.9338$
$K = 2$	$\mathcal{L}^{(2)}_{200}$	$0.8566x^2 + 0.0872x^9 + 0.0296x^{10} + 0.0084x^{40} + 0.0126x^{41} + 0.0007x^{79} + 0.0048x^{200}$	$\mathcal{G}^{(200)}_2 = 1.9368$
$K = 2$	$\mathcal{L}^{(2)}_{300}$	$0.8559x^2 + 0.0982x^9 + 0.0179x^{10} + 0.0205x^{37} + 0.0035x^{104} + 0.0013x^{105} + 0.0027x^{300}$	$\mathcal{G}^{(300)}_2 = 1.9375$
$K = 2$	$\mathcal{L}^{(2)}_{400}$	$0.8553x^2 + 0.1070x^9 + 0.0088x^{10} + 0.0169x^{36} + 0.0044x^{37} + 0.0045x^{117} + 0.0011x^{118} + 0.0020x^{400}$	$\mathcal{G}^{(400)}_2 = 1.9380$
$K = 2$	$\mathcal{L}^{(2)}_{500}$	$0.8556x^2 + 0.0980x^9 + 0.0182x^{10} + 0.0052x^{37} + 0.0159x^{38} + 0.0002x^{130} + 0.0052x^{131} + 0.0017x^{500}$	$\mathcal{G}^{(500)}_2 = 1.9383$
$K = 3$	$\mathcal{L}^{(3)}_{100}$	$0.9201x^2 + 0.0307x^{13} + 0.0389x^{14} + 0.0032x^{55} + 0.0072x^{100}$	$\mathcal{G}^{(100)}_3 = 2.7836$
$K = 3$	$\mathcal{L}^{(3)}_{200}$	$0.9182x^2 + 0.0690x^{13} + 0.0005x^{57} + 0.0092x^{58} + 0.0031x^{200}$	$\mathcal{G}^{(200)}_3 = 2.7858$
$K = 3$	$\mathcal{L}^{(3)}_{300}$	$0.9177x^2 + 0.0691x^{13} + 0.0003x^{14} + 0.0096x^{64} + 0.0009x^{65} + 0.0024x^{300}$	$\mathcal{G}^{(300)}_3 = 2.7871$
$K = 3$	$\mathcal{L}^{(3)}_{400}$	$0.9178x^2 + 0.0602x^{13} + 0.0093x^{14} + 0.0039x^{66} + 0.0062x^{67} + 0.0006x^{167} + 0.0019x^{400}$	$\mathcal{G}^{(400)}_3 = 2.7875$
$K = 3$	$\mathcal{L}^{(3)}_{500}$	$0.9175x^2 + 0.0681x^{13} + 0.0013x^{14} + 0.0101x^{63} + 0.0010x^{189} + 0.0008x^{190} + 0.0012x^{500}$	$\mathcal{G}^{(500)}_3 = 2.7877$

Table 3.8: Optimized repetition degree distributions of high maximum degree for 2-MPR and 3-MPR with the associated maximum achieved load

Chapter 4

Multi-Power IRSA

4.1 Introduction

Multiple previous studies of IRSA protocol have explored the question of: How can we enhance the performance of IRSA? One way to increase the achieved throughput of this protocol is through capture. The capture effect happens when multiple signals collide at the receiver with different received powers. The colliding signals could have different or equal transmission powers, but they arrive with different power levels at the receiver. These differences are the results of propagation phenomena, such as fading, shadowing and the near-far effect [31]. Prior work about the impact of capture effect on wireless networks has assumed overlapping of two or more signals in one of two situations. In the first situation, the strong signals can be always recovered despite the presence of other colliding signals on the same slot. In the second situation, the weak signals arrive before the strong signals, resulting in both signals capturing the channel and being lost because the strong signals corrupt the weak signals [64]. The choice of modulation, channel fading and noise, interference from co-channels and space diversity will also affect the capture effect at the receiver.

CSA protocols have been widely studied under the capture effect. The main assumption is that captures occur when two (or more) packets are received with sufficiently different powers. The packet with the highest power is recovered first, and then the second after SIC. This paves the way for potentially achieving $T > 1$. An important point is that the capture effect combines very well with the SIC mechanism that is assumed at the core of IRSA: it inherently increases the possibility of packet retrieval (followed by SIC), without requiring any fundamental change of the IRSA base functioning.

Motivated by the fact that the capture effect is a natural result of a realistic assumption of

communication channel, in this chapter, we focus on the scenario of an IoT network where the packets of different nodes are received with different powers at the base station. The difference in the signals' received powers is either per design due to different transmission power, or induced by the fact that the nodes are at different distances from the base station, thus they encounter different fading coefficients. We presented a new direction in analyzing IRSA: basically, one node has always the same transmission power. This changes the behavior of the protocol, and makes the density evolution (arguably) more difficult. We analyze the protocol behavior using a new density evolution, which is based on dividing nodes into classes with different received powers.

By computing the probability to decode a packet in the presence of the interference, we explore the achievable throughput and its associated gain and show the excellent performance of Multi-Power IRSA.

4.1.1 Capture, Related Work and Our Approach Multi-Power IRSA

In a heterogeneous setting, where IoT nodes are placed in different positions from the base station, the different path loss factors experienced from different channel conditions naturally give rise to the capture effect at the receiver. The effect of such path loss in wireless communication has been extensively studied. By developing stochastic geometry methods, in [65], the received interference and capture probability from a set of nodes in a uniform wireless network is computed (without SIC). Other studies [66] have considered the case of capture effect with the ability of performing SIC at the receiver for the CSA protocol family. The stability of a slotted ALOHA system with capture effect is evaluated in [31], where terminals are divided into two groups and the capture effect is modelled by capture probabilities. In [32], a study of Irregular Repetition Slotted Aloha (IRSA) in the presence of capture effect and SIC was presented: there, the transmission power is identical for all users and the difference of received power is entirely caused by path loss from distance.

Several works on the CSA family have considered the impact of the capture effect on the protocol performance. Some of these studies have not changed the transmission power, but only considered the different received power signals due to capture effect. In [26], the analysis of IRSA assumed capture effect at the receiver. The work has optimized some repetition degree distributions in a finite frame length setting. The derived distributions are shown to achieve throughput largely exceeding 1 [packet/slot]. [27], proposes Extended IRSA (E-IRSA) as an extended version of IRSA that exploits the capture effect to perform SIC at the receiver. Intra-slot SIC has been applied to decode more than one collided packet

per slot. Simulation results show that E-IRSA protocol allows reaching the maximum theoretical throughput even in scenarios where the number of active users is higher than the number of slots per frame.

CSA performance has been also evaluated assuming a fading channel in [28–30]. The impact of the capture effect on the SIC-enabled slotted ALOHA framework has been also addressed in [31, 32]. In general, the capture effect, combined with SIC, helps to increase throughput.

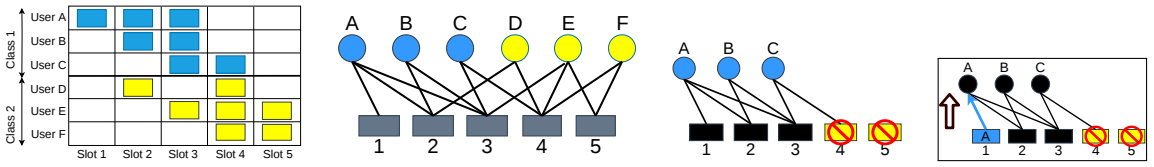
Several schemes where the transmitter power is changed at each transmission have been introduced. They appear to be a natural generalization of IRSA: for each transmission, the users randomly select their degree, and then they randomly select their slots; they could randomly or deterministically select their transmission power as well.

In [43], a NOMA-Based IRSA scheme with different transmission powers is proposed, and typical Density Evolution is used for studying the system while the Differential Evolution method is used for optimizing the parameters. In [67], CRDSA with transmit power diversity has been considered. The transmission power distribution was optimized by differential evolution with respect to a total power budget to enhance the throughput.

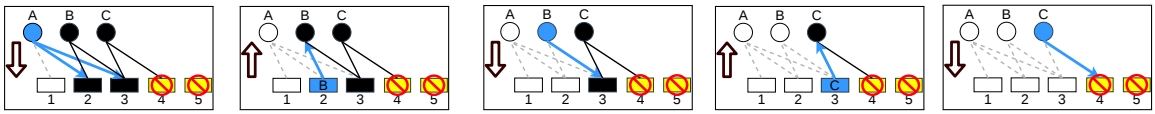
Our approach called *Multi-Power IRSA (MP-IRSA)* differs from previous work by introducing the important differences that *the replicas of the same packet have the same power*, and that the users are *grouped into classes* corresponding to *identical received power*. MP-IRSA is well-suited to heterogeneous IoT networks, *e.g.* the signals of the farthest terminals arrive at the base station with lower strength, and they might be grouped in classes with lower power: their adjusted transmission power stays low, providing energy savings. MP-IRSA allows the phenomenon of *cascading* decoding, which can increase the throughput.

We analyze MP-IRSA through a new variant of multi-class density evolution that relies on the grouping of users with the same received signal power into one class. With that, we are able to study experimentally the performance of the protocol: the influence of each user class on the decoding process and the performance of the other classes are also studied. The impact of the density and the power of the class on the achievable throughput and the maximum load is analyzed.

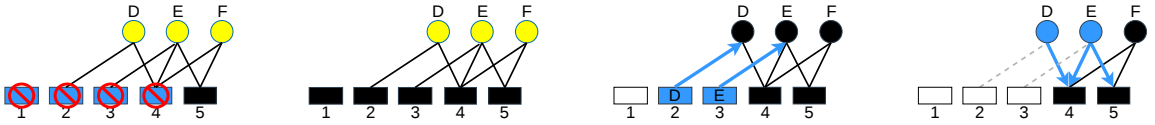
In the next section, we describe the system model of MP-IRSA with capture effect at the receiver, and we provide an illustrated example.



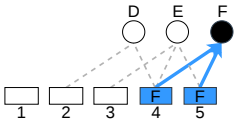
(a) User transmissions on slots (b) Tanner Graph representation (c) View from Class 1 (d) Class 1 step 1



(e) Class 1 step 2 (f) Class 1 step 3 (g) Class 1 step 4 (h) Class 1 step 5 (i) Class 1 step 6



(j) View from Class 2 (k) Class 2 after Class 1 decoding (l) Class 2 step 1 (m) Class 2 step 2



(n) Class 2 step 3

Figure 4.1: Example of MP-IRSA, with $\Lambda_2 = \frac{2}{3}$ and $\Lambda_3 = 13$ - and of the decoding process

4.2 Principle of IRSA with Capture and Multiple Classes

We start from the IRSA system model as in Section 2.4, and introduce some changes. We consider κ classes of M_c users for each class, $c \in K$ where $K = \{1, 2, \ldots, \kappa\}$ is the set of classes.

Each class contains users whose signals are received with the same power by the base station. Each user has one single packet to transmit (all packets have identical sizes and fit a slot). In line with IRSA system model: one considers a frame of size N slots. Each user transmits identical copies of its packet on randomly chosen slots. The number of replicas is picked by each user at the beginning of the frame from a random discrete distribution Λ_c which is common for all users of the same class c, in our assumption. Precisely, we define:

$$\Lambda_{c,i} \triangleq \Pr(\text{a user of class } c \text{ repeats his packet } i \text{ times}) \tag{4.1}$$

At the end of the frame, the receiver receives the superposition of the physical signals sent by collided users on each slot. The major departure from classical IRSA is that the collision channel model is changed. We abstract the collision model as:

consider a given slot, where $n_1 + n_2 + \ldots + n_k$ replicas that had been sent from k classes, with n_i being the number of users of class i transmitting on that slot. Let us denote the number of replicas of class i that can be recovered in the presence of the interference of other classes as $D_i(n_1, n_2, \ldots n_k)$. Here, we also adopt an intra-class collision model, i.e. whenever more than two users of the same class are on one slot ($n_i \geq 2$), no packet from class i can be recovered on the slot. Packets from other classes, however, might sometimes still be recovered when their received power is higher.

The decoding process is iterative [1], and we consider the following rules while performing the decoding of class i:

- We consider all slots containing exactly one packet from class i. In each of these slots, this packet can be recovered if the interference from other classes is sufficiently small (lower than a given threshold).
- Once one packet has been recovered on a certain slot, the copies of this packet can be removed from other slots at the physical signal level by *Successive Interference Cancellation (SIC)*.

A simple example is given in Fig. 4.1a (with Tanner graph in Fig. 4.1b: bipartite graph with burst nodes as circles, slot nodes as rectangles and one edge for each transmission), and the detailed explanation of the steps of the decoding process is shown in Fig. 4.1. This figure shows two classes, with 3 nodes in each class where the received power of class 2 (in yellow) is lower than the received power of class 1 (in blue). The transmissions from the two classes are colliding on the slots: $2, 3$ and 4. The decoding process starts with class 1 (Fig. 4.1c) as follows: the intra-class decoding starts first by searching for the packets without collisions (singletons). The packet of burst node A of class 1 is a singleton received on slot node 1, so it is decoded and removed from its replicas positions on slot nodes 2 and 3 as shown in Fig. 4.1d and Fig. 4.1e. The removal of the packet from burst node A on slot node 2 makes the packet of burst node B on slot node 2 colliding only with a replica from the burst node D from class 2. Since the received power of class 2 is considerably lower than the received power of class 1, the packet of burst node B could still be considered as a singleton. In this case with an "under-threshold" interference from class 2, it is in turn decoded and removed from slot nodes 2 and 3 as well as shown in Fig. 4.1f and Fig. 4.1g. Removing the packet of burst node B from slot node 3 makes the packet of burst node C decodable even in the presence of a replica of burst node E from class 2. Note that the packets of class 1 can be removed in the presence of under-threshold interference from class 2 but the opposite is not true. The same process of iterative decoding of class 2 continues (Fig. 4.1k) by removing the packets of burst nodes D and E on slot nodes 4 and

5, as shown in Fig. 4.1l and Fig. 4.1m and recovering user F in Fig. 4.1n. Note that in this example, performing the decoding of class 2 packets is impossible before the decoding of class 1 packets (Fig. 4.1j).

In classical IRSA [1], there is only one class. In our work, various decoding order strategies can be adopted depending on the moment at which we switch from decoding packets from one class to decoding packets from another class. When all packets have been retrieved, or no packet from any class can be decoded anymore, the decoding process is considered to be ended. Note that the decoding order matters only for the modeling (or simulation) of the decoding process, as a real MP-IRSA decoder would retrieve any packet on a slot where capture is possible; but the important point is that all the strategies will lead to the same final state, hence all orders, including the ones easing the analysis, are equally valid.

In the next section, we explain in detail our new density evolution equations for IRSA with power classes (MP-IRSA).

4.3 Density Evolution of MP-IRSA

In line with the basic density evolution equations in Section 2.4.3, we provide a new density evolution formulation for MP-IRSA, and use it later for our performance analysis. The concept of degree distributions remains the same for MP-IRSA however, we introduce a different node degree distribution for each class. The following function is introduced to represent the repetition degree distribution for class c:

$$\Lambda_c(x) = \sum_{i=0}^{L} \Lambda_{c,i} x^i \quad \text{with } \Lambda_{c,i} \text{ defined in Eq. (4.1)}$$

The edge degree distribution from the user perspective $\lambda(x)$, defined in Eq. (2.3), becomes related to the class of user. On the other hand, The edge degree distribution from the slot perspective $\rho(x)$ remains the same as in Eq. (2.3), as the slots are not grouped in classes. We define the edge degree distribution as the following:

$$\lambda_c(x) = \sum_{i=0}^{L} \lambda_{c,i} x^{i-1}, \quad \rho(x) = \sum_{i=0}^{M} \rho_i x^{i-1}$$

$\lambda_{c,i}$ defines the probability that a randomly selected edge from the Tanner graph in Fig. 4.1b is connected to a burst node of class c which has a degree i.

ρ_i defines the probability that an edge is connected to a slot node of degree i.

We consider a random edge corresponding to one given user of class c transmitting on a random slot:

- $q_{c,i}$ is the probability that the edge which is connected to one user of class c is not known.
- $p_{c,i}$ is the probability the edge that connects a user of class c to a random slot is not revealed.

where: i is the iteration number.

The last packet from the class c on a slot can be recovered if all the packets from the same class and the classes with a higher power have been already decoded in the previous iterations:

$$(1 - p_{c,i}) = \delta_c \left(1 - q_{c,i}\right)^{\ell-1} \tag{4.2}$$

Where: δ_c is the probability that a packet from class c can be decoded in the presence of interference from other classes, and:

$$\delta_c = f_c \left(q_{1,i}, q_{2,i}, ..., q_{c-1,i}\right) \tag{4.3}$$

Where f_c is the function of the probability of the unknown edges of all classes, which will be clearly defined in Eq. (4.8) in the next section. A packet from class c can be recovered if there is at least one copy of this packet that has been decoded on another slot:

$$q_{c,i} = p_{c,i-1}^{\ell-1} \tag{4.4}$$

Using the polynomial representations of burst degree and edge degree distributions, we can average the edge probabilities for class c in (4.2) and (4.4) as follows:

$$p_{c,i} = 1 - \delta_c \cdot \rho \left(1 - \lambda_c \left(p_{c,i-1}\right)\right) \tag{4.5}$$

By letting $M \to \infty$ (asymptotic case), we can write:

$$p_{c,i} = 1 - \delta_c e^{-\frac{G_c}{R_c}\lambda_c(p_{c,i-1})} \tag{4.6}$$

Where: G_c is the load of class c which can be defined as the average number of users per slot of class c, i.e, $G_c = \frac{M_c}{N}$. R_c is the rate of class c and is defined as $R_c \triangleq \frac{1}{\Lambda_c'(1)}$. The necessary condition to decode more packets in each class at each decoding iteration is : $p_{c,i} < p_{c,i-1}$ which can be written more precisely using (4.6) as follows:

$$1 - \delta_c e^{-\frac{G_c}{R_c}\lambda_c(p_c)} < p_c \tag{4.7}$$

Given the parameters $(G_c, \Lambda_c, \delta_c)$ for each class $c \in K$, one can simply follow the evolution of the decoding process to ultimately estimate the number of decoded packets at the end of

the decoding process and also to identify the best degree distribution to be given to each class in order to achieve the highest load.

In the next section, we write and discuss the mathematical formula of the probability δ_c, that a packet from class c can be decoded in the presence of interference from other classes.

4.4 Inter-Class Interference Model

In a heterogeneous IRSA setting, where capture effect is considered, the analysis of the decoding process of a packet from a given class c has to take into account two types of packets: the packets from other classes for any class $n \in K$ with $n \neq c$ (considered as interference) that collide on the same slot and also collisions from the same-power packets (same class c). Thus, the density evolution of a class-based IRSA links two interference models: intra-class interference that can be studied and processed as in classical IRSA in [1] and inter-class interference that appears in the factor δ_c for each class. In case of capture at the receiver, the weaker signals are ignored when decoding the strong signal. This gives a condition on the inter-class interference on each slot which affects the decoding process, and it can be expressed as:

$$\delta_c = Pr\left\{ \sum_{\substack{n=1 \\ n \neq c}}^{\kappa} X_n \Pi_n \leq T_h \right\} \tag{4.8}$$

where: Π_n is the received power of class n, X_n is a random variable representing the number of undecoded packets from class n on the slot, and T_h is the interference threshold (SINR) beyond which a packet can be decoded. Indeed, Eq. (4.8) integrates the interference with the effect of path loss and fading on the received power, which has been widely covered in wireless networks. One can wonder if there is a simple distribution model for such interference, but when the nodes are placed on different distance radius around the base station, no simple, closed-formula, mathematical models could fit the left part of Eq. (4.8). Notice that Eq. (4.8) is linked to stochastic geometry [65], in particular when the received power directly depends on distance but also depends on the stage of the decoding process.

We can write that X_n follows a Poisson distribution with a mean B_n which represents

the average number of undecoded users from class n on the slot:

$$B_n = \frac{\lambda_n(q_n) M_n \sum_\ell \ell \Lambda_{n,\ell}}{N} \tag{4.9}$$

Where: $\lambda_n(q_n)$ represents the probability that an edge on class n has not been revealed and M_n is the total number of users in class n. The sum $\sum_\ell \ell \Lambda_{n,\ell}$, represents the average number of replicas of a packet in class n. The left side of the inequality in (4.8) is a sum of scaled Poisson random variables, which generally has not a closed formula for its distribution. However, a good approximation can be computed numerically, which is what we do in this chapter. In the next section, we provide an analytical analysis for MP-IRSA through simulations.

4.5 Numerical Results

We mainly focus on the asymptotic performance of MP-IRSA using our density evolution, which is based on classes (Section 4.3). The primary goal is to show that isolating the users with the same received power in a class and taking into account the interference from other classes has a strong and notable impact on the decoding process, since the classes with high received power have more probability to be decoded first, and followed by decoding lower-power classes and so on. Indeed, results evidence a cascading effect on the decoding process starting from the highest-power class to the lowest-power class. Our second objective is to compare the throughput of MP-IRSA with the optimized multi-power CRDSA which was proposed in [67] and also to confirm that it has higher than the throughput of classical IRSA in [1]. This emphasizes the importance of favoring captures into the system design. Our third objective is to explore the impact of the various parameters and of their numerical values.

In our numerical experiences, we study the influence of the number of classes, the *density* of each class (defined as the proportion of the users in that class, e.g. density 30% or 0.3 means that 30% of all the users are in that class), and the (transmission) power of each class on the achievable throughput and its associated gain, in different scenarios. In the legends of the figures, $c1$, $c2$ denote class 1, class 2 respectively, d denotes the density and p denotes the transmission power. In our analysis, the users' degree distribution for IRSA is the soliton distribution from [6]. As the main goal of this work is to study the impact of different power IRSA classes, we have used the same soliton distribution for all classes, since it has been proven to achieve a throughput of 1 [packet/slot] in a different setting with a collision channel model. The selection of the optimized class degree distribution is out of

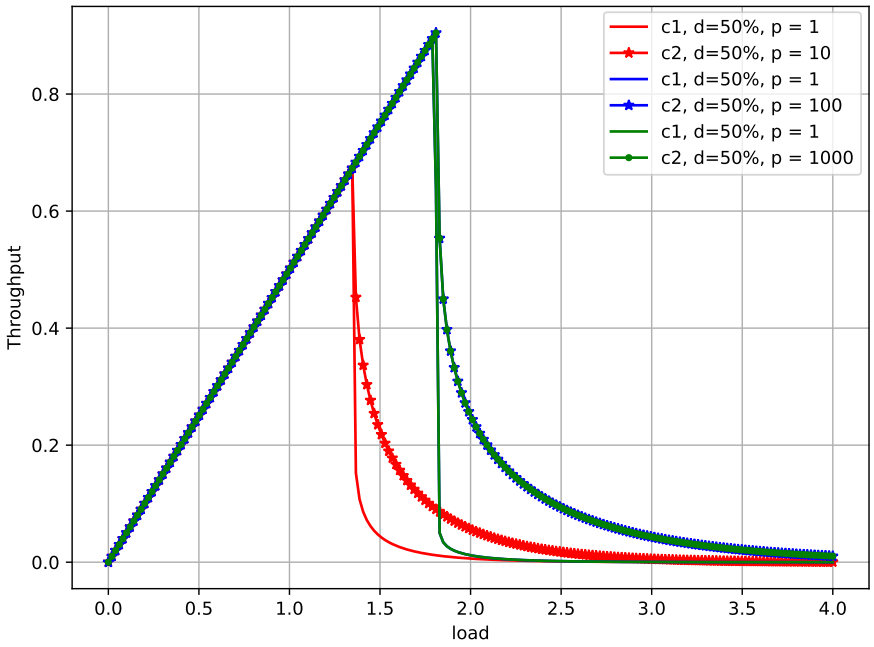


Figure 4.2: The throughput of 2 classes with different powers and identical density 50%. Each color is for one scenario with 2 classes.

the scope of this work. We gradually increase the load (x-axis of the curves) and plot the corresponding throughput for each class. The total throughput of the network will be the sum of the throughput of all classes.

Fig. 4.2 shows the throughput achieved by IRSA for two equal density classes and different powers. We tested three different cases with a power difference between both classes corresponding to 10 dB, 20 dB, and 30 dB respectively. Typical of IRSA, the curve starts to grow linearly (with near 0% loss) until a threshold, after which the throughput falls quickly.

For a sufficiently large power difference, *i.e.* with $P = 100$ (+20 dB) or with $P = 1000$ (+30 dB), the maximum achieved throughput is the near-maximal value: $T_{max} = 0.894$ for the first class and $T_{max} = 0.904$ for the second class. This means that the total throughput is $1.798 = 0.894 + 0.904$, a near-doubling of the 1 [packet/slot] bound without capture. This illustrates from the *cascading* effect of MP-IRSA with high power difference: the decoder can finish decoding the packets of the highest power class (intra-class decoding) almost entirely as if the other class does not exist, and then moves to decode the packets of next class (inter-class decoding).

This phenomenon cannot occur with smaller power differences, and this is illustrated for $P = 10$: where $T_{max} = 0.673$ for both classes. The decoding process of class 2 can be blocked by class 1, and the receiver probably will have to decode by *cycling* back and

forth between the classes. The maximum total throughput is 1.346, significantly less than before.

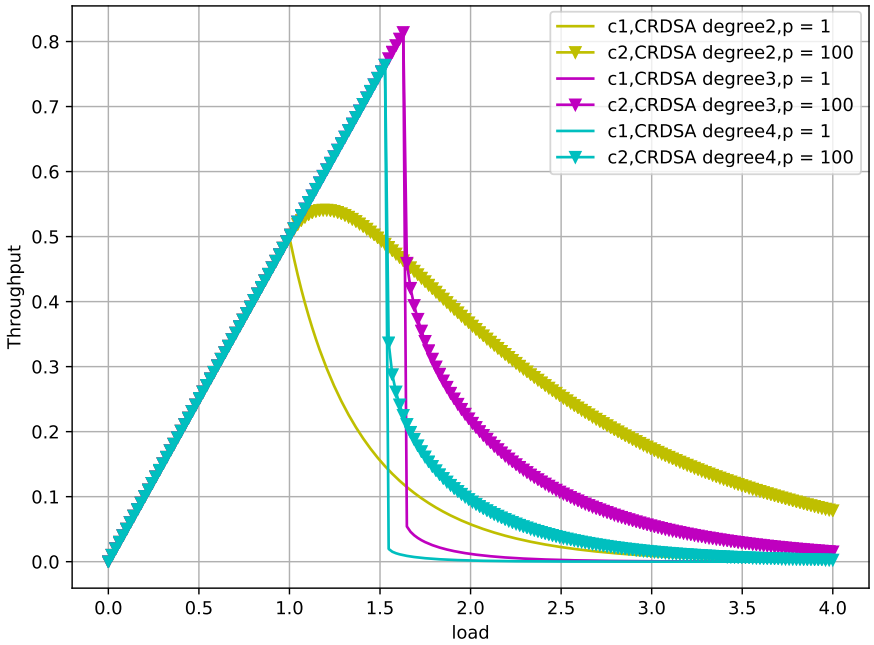
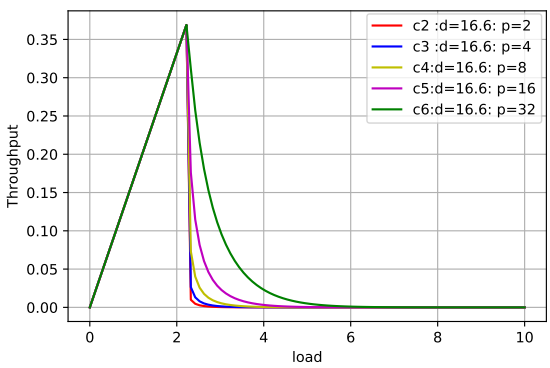


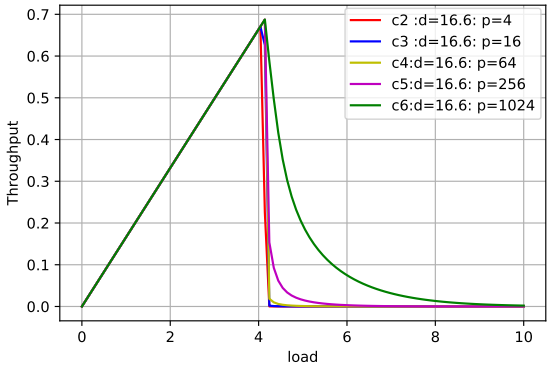
Figure 4.3: The throughput of 2 classes of CRDSA with 2, 3 and 4 repetitions with different powers and similar densities

Fig. 4.3 shows the performance of two classes with CRDSA with 2, 3, and 4 repetitions respectively. The goal of this comparison is to confirm that IRSA can perform better than CRDSA, as in the classical case, as was shown in [1, 6], and confirm this even in the case of classes. With the power of classes set to $P_{c_1} = 1, P_{c_2} = 100$, the best degree for CRDSA appears to be 3 as in CRDSA without capture. For that value, IRSA indeed still performs better than CRDSA which achieves a maximum throughput equal to 0.814. The cascading effect for CRDSA degree 3 and 4 as for IRSA can be also estimated.

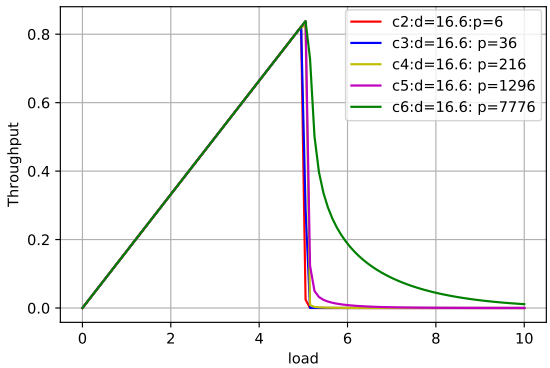
To understand more the impact of the power difference between the classes on the system performance, we study the case of 6 classes with a geometrical difference in power and for different SIR levels, as in Fig. 4.4. One of the key factors that affect system throughput is user differentiation. As before, when the classes have more difference in power, the obtained throughput can be higher. This is related to the fact that the created interference from one class to another is more tolerable in the SIC receiver as seen in Fig. 4.4a, Fig. 4.4b, Fig. 4.4c and Fig. 4.4d (for clarity, the results of only 5 classes out of 6 are plotted). Interestingly, for a geometric factor of 6 (see Fig. 4.4c), the total throughput with 6 classes is almost 5 ($> 0.8 \times 6$). This implies that it is almost as if there were 6 cascading decoding,



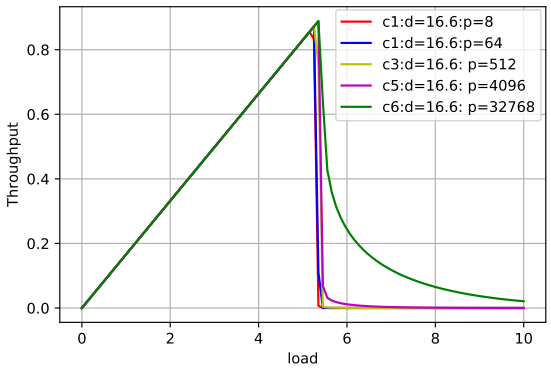
(a) 6 classes with a geometrical difference of 2 in power and SIR = 1



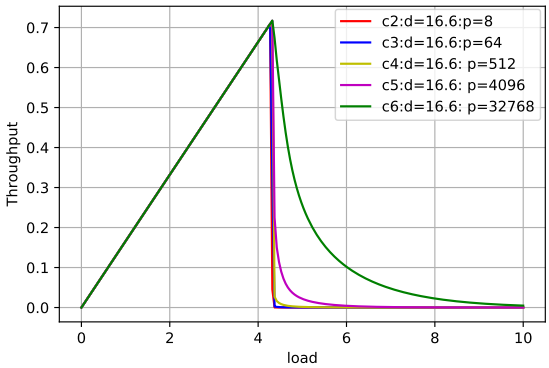
(b) 6 classes with a geometrical difference of 4 in power and SIR = 1



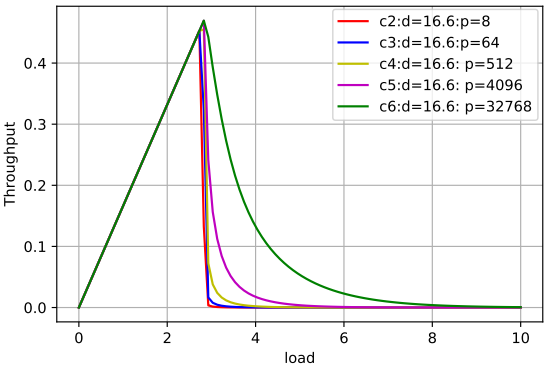
(c) 6 classes with a geometrical difference of 6 in power and SIR = 1



(d) 6 classes with a geometrical difference of 8 in power and SIR = 1



(e) 6 classes with a geometrical difference of 8 in power and SIR = 2



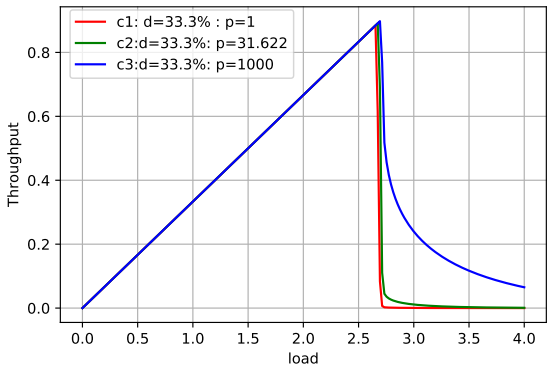
(f) 6 classes with a geometrical difference of 8 in power and SIR = 4

Figure 4.4: The throughput of IRSA with 6 classes and geometrical difference in power and equal densities and SIR = 1

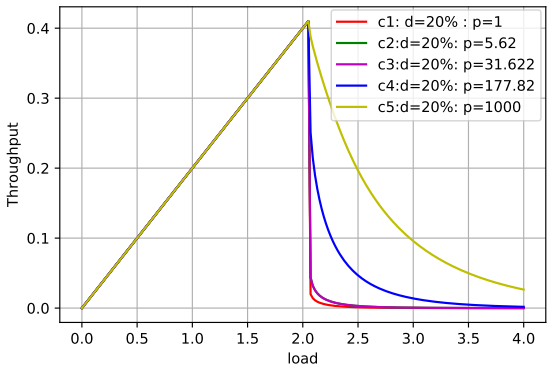
with more than 80% successfully used slots for each class. For smaller geometric factors, the total throughput falls dramatically less, and adding more intermediate classes seems to sometimes decrease performances.

Another important key factor is the receiver sensitivity (i.e, the SIR threshold T_h, that we will refer to as simply "the SIR"). Fig. 4.4d shows a throughput of $T_{max} = 0.966$, in a case where SIR = 1, while the throughput degrades almost to the half $T_{max} = 0.469$ when the SIR is 4 times greater (Fig. 4.4f). This can be seen easily by considering a packet from the class 3 and one of its interferes in both cases. In case of SIR = 1, the collision between 8 packets from class 2 and one packet from class 3 can be still decoded since $SIR = \frac{512}{8\times64} = 1$, while in case of SIR = 4, the packet of class 3 can be decoded in case of interference with one or two packets at maximum from class 2.

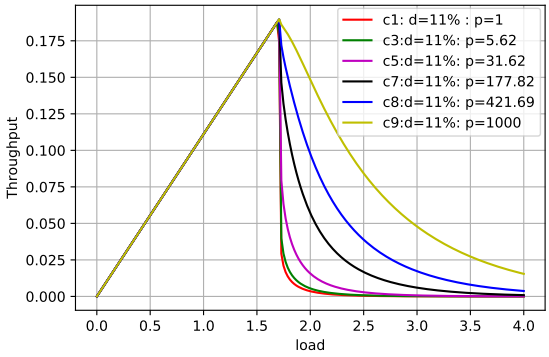
Fig. 4.5 shows the importance of the number of classes on the achieved throughput and the associated gain. The performance degrades notably when there are more classes in the network, as in Fig. 4.5c due to the increase in the interference between the classes. The total throughput is around 2.7, 2.0, and, 1.6 respectively. It appears as if the intermediate classes were blocking the cascading effect. Thus, the number of classes should be chosen after taking into account different parameters, including the received power range and the needed SIR.



(a) 3 classes in a power range (0,1000)

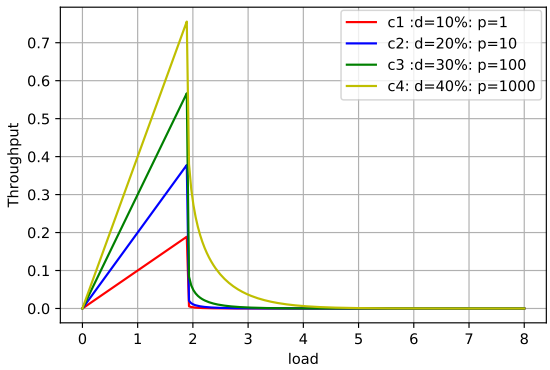


(b) 5 classes in a power range (0,1000)

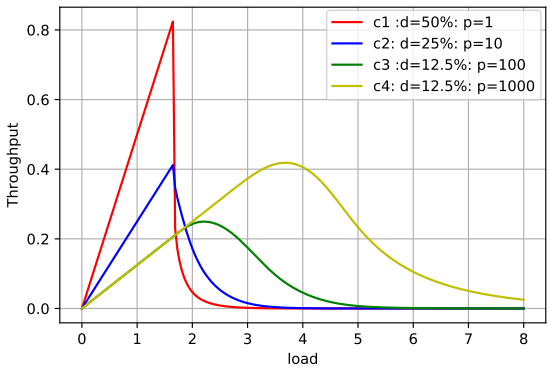


(c) 9 classes in a power range (0,1000)

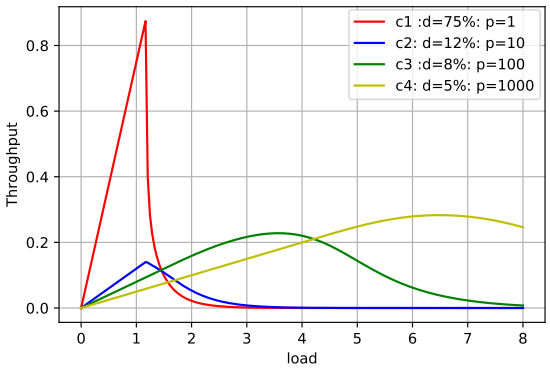
Figure 4.5: The effect of increasing the number of classes in the same power range



(a) 4 classes with different densities-10,20,30,40



(b) 4 classes with different densities-50,25,12.5,12.5



(c) 4 classes with different densities-75,12,8,5

Figure 4.6: The effect of different densities of the classes

Another critical factor is the users' densities in each class. Fig. 4.6 shows different scenarios with different users' densities. Recalling that the decoding process starts from the highest power class, Fig. 4.6a illustrates what occurs when the highest power class has also the highest density. The effect of the class density is not clear in this case: even though the power ratio between classes is 10, lower power classes are often blocking high power one (as in Fig. 4.2).

The impact of class density is clearer in Fig. 4.6b and Fig. 4.6c, where the highest power class (in yellow) is more affected by the interference from the other classes (87.5% and 95% of the users are interfering with class 1). As shown in both figures, the decoding of the highest power class continues correctly (linearly) up to a certain load, and then it degrades slowly, affected by the high interference from other high-density classes. Two observations can be made: in those cases, for high values of the load, the lower power classes unnecessarily "jam" the channel, because their packet success rate goes to 0; but on the other hand, when the degradation of the throughput of higher classes is not reached, the system is interesting as it introduces a form of priority between users.

4.6 Conclusion

In this chapter, we formally introduced Multi-Power IRSA (MP-IRSA), as a random access method, when the replicas of different users are transmitted/received with different powers. We introduced a new density evolution variant based on grouping users into classes: it allows analyzing the performance of MP-IRSA. Multi-power IRSA proves to be a better choice than CRDSA and classical IRSA and confirms the benefits of the capture effect at the receiver well beyond the IRSA limit of 1 [packet/slot]. The impact of different system factors on the achieved throughput and the associated load was extensively studied. First, the power difference between the classes and the number of classes plays a huge role in the decoding process, and we observe the best results when the power difference is large: decoding is cascading (one class after the other), instead of cycling (switching between different classes). The receiver sensitivity and the density of each class are other important factors that we studied.

Another key factor that we did not explore is the repetition degree distribution: we used the same soliton distribution for all classes. Finding methods to optimize a common degree distribution for all classes or multiple degree distributions for multiple classes, and studying its impact, is a possible future work. Overall, diversity in power is the second method studied in this thesis for the performance improvement of IRSA, after K-MPR. Notice that K-IRSA can provide an upper bound of MP-IRSA, by considering how many packets from different classes can be possibly decoded on the same slot, by capture and intra-slot SIC alone. In that sense, the two chapters are complementary.

Chapter 5

Design of IRSA with Interference Cancellation Errors

5.1 Introduction

In the previous chapters, as in many IRSA research works, some idealized transmission models have been used, without necessarily taking into account various errors, such as transmission errors or other kinds of errors that could affect the achieved throughput. In particular, IRSA has been widely studied with a common assumption is that interference cancellation can always be applied perfectly. In common IRSA literature, a transmitted packet is denoted a *singleton packet* when there are no packets left from other users that were transmitted in the same slot, possibly after having performed some interference cancellations. Additionally, a singleton packet is considered to be successfully decoded with probability 1. In practical communication systems and signal processing under noise and imperfection, the probability of interference cancellation error is in general non-zero. Channel estimation errors, caused possibly by Doppler shift or noise on the pilot signals, can lead to dramatic degradation of the system performance. These estimation errors cause an imperfect cancellation of the signal which in turn accumulate as undesirable energy, that causes decoding errors and degrades the performance.

In this chapter, we study the design of IRSA protocol, accounting for a non-zero probability of error due to imperfect interference cancellation (IC). This is one of the sources of errors that can be taken into account, and it complements the existing research work on IRSA studying transmission errors [38]. We will consider that the probability of correctly retrieving a packet decreases after every interference cancellation iteration applied to the packets in a slot. Moreover, since all packets of a user can be lost due to interference can-

cellation failure, there is a non-zero probability that a user's information is completely lost at the end of the iterative process. For this reason, similarly to [38], a user recovery ratio α will be introduced and used as a new parameter for the system in our analysis. The load of the system will be optimized concerning the desired user recovery ratio. We analyze the convergence of density evolution, and we searched for the user degree distribution that maximizes the channel load. A new parameter is introduced to model the packet loss rate of the system, which is non-zero due to potential IC errors. We investigate the trade-off between optimal load and packet loss rate, which sheds light on new optimal distributions that outperform the known ones in presence of IC errors. Finally, we show that our asymptotic analytical results are consistent with simulations obtained on a finite number of slots.

5.2 System Overview and Related Work

In this section, we will study the impact of SIC errors and optimize the design of IRSA under this more realistic and general model. We will also shed the light on the related work of IRSA with decoding errors, and then we will formulate our main problem to optimize the performance of IRSA while taking into account the SIC errors.

5.2.1 Successive Interference Cancellation Model

The succession of interference subtractions may lead to the wrong retrieval of a packet. A wrongly retrieved packet would result in a packet loss and cannot be used to subtract interference for other remaining packets on the slot.

Let ℓ be the number of packets collided in a slot. We denote by w_ℓ the probability that a packet can be correctly retrieved after SIC (i.e., by eliminating the other $\ell - 1$ packets collided in the slot). We consider the case of an exponential error due to SIC:

$$w_\ell \triangleq \gamma^{\ell-1}, \tag{5.1}$$

where γ is defined as the *interference cancellation efficiency* (or *SIC efficiency*) for our model. γ is the probability to correctly cancel the interference of one packet on one slot, and this probability is independent of the other packets in the slot.

Note that $\gamma = 1$ would lead to the classical model of SIC without error studied in [68], which is a special case of Eq. (5.1). [1, Appendix A] proposes a short generalization of the density evolution for IRSA for the case where the ℓ^{th} packet is retrieved with an arbitrary probability w_ℓ: when applied to the special case of IRSA, our later equations can also be

derived from [1, Eq. (9)]. These extensions of density evolution that consider errors were not exploited in [1], neither in an optimization problem nor through numerical results.

The above loss model Eq. (5.1) is an idealized model of the SIC process. More accurate modeling of the physical layer inter-slot SIC process has been described in [1]. Specifically, before any SIC operation, the signal parameters (amplitude, phase, frequency offset) might need to be estimated. Estimation errors could happen, and they lead to incorrect signal subtraction (cancellation). After each SIC operation, residual energy is present due to the fact that the imperfection of the cancellation process. The residual energy would accumulate as the number of SIC operations on the same slot increases. Estimation errors and residual energy due to imperfect cancellation could lead to significant SIC failure probability. [1, Appendix B] and [68, Appendix A] observe that under their respective physical layer models and assumptions, the packet error rate is 10^{-2} and 10^{-3} respectively after $\ell - 1 = 7$ SIC operations. This has been a reasonable justification for the assumption of perfect SIC in some literature. However, for instance, operating at lower E_b/N_0 (see [1, Fig. 9] or [68, Fig. 10]), or maybe other reasons (could be having a larger frame which may need more SIC operations), would lead to an increase of those probabilities.

Our simplified model in Eq. (5.1), is essential, a model where each SIC operation has the same probability $1 - \gamma$ to be the breaking point after which the residual packets can no longer be recovered. Some reasons could be: the accumulation of random residual energy errors, or an infrequent large error in the estimation of the parameters of transmission (phase, amplitude, carrier frequency offset) due to a random event. In other words, $1 - \gamma$ is the probability that a packet cannot be correctly subtracted in a slot when SIC is applied. As a consequence, the probability to retrieve a packet on a certain slot, through canceling the interference from the other $\ell - 1$ colliding packets, is given by $\gamma^{\ell-1}$. This explains the above expression of the exponential error in Eq. (5.1). This model also corresponds to an approximation, where γ would be selected to be the average SIC efficiency observed from a real SIC model.

In practice, γ is not expected to be constant with ℓ. For instance, considering the data points of the curve from [1, Fig. 9] for $E_b/N_0 = 1.5$ dB (resp. $E_b/N_0 = 1.8$ dB), one can observe a variation of the estimated value[1] of $1 - \gamma$ of more than 40% (taking one value of E_b/N_0 on the curve, and looking at the values of *PER* for various numbers of

[1]By denoting PER_j the Packet Error Rate (PER) after j SIC operations, and writing $\text{PER}_j = 1 - \gamma_j^j(1 - \text{PER}_0)$, one can compute an estimate of γ_j from the data point for $j+1$ bursts in the curve of [1, Fig. 9]; with PER_0 being the PER for the AWGN channel without SIC. For $E_b/N_0 = 1.5$ dB (resp. $E_b/N_0 = 1.8$ dB), one obtains the estimates: $\gamma_1 = 1-0.9\times10^{-2}, \gamma_3 = 1-1.1\times10^{-2}, \gamma_7 = 1-0.8\times10^{-2}$ (resp. $\gamma_1 = 1-1.2\times10^{-4}, \gamma_3 = 1 - 1.7\times10^{-4}, \gamma_7 = 1 - 1.6\times10^{-4}$).

colliding packets). This is an idealized physical layer simulation, and one might expect greater variations with real hardware. On the other hand, notice that an increase of γ after numerous SICs might also be mitigated by the fact that most packets are recovered after a limited number of SICs (typically less than 5). This is the effect of SIC errors in the later iterations of the decoding process, which also needs to be assessed.

Some other prior works have accurately studied the impact of errors in IRSA decoding. For instance, the impact of imperfect CSI with PDMA/IRSA [69], or equivalently estimation errors with MARSALA/CRDSA [70, 71]. Although sophisticated parameter estimation methods are used, and mathematical expressions are obtained, they do not easily translate into a simple model of SIC failure probability. This is due to that they have to be iterated in a manner that parallels the IRSA/CRDSA decoding process, and the results are obtained by simulations. Alternately, some insights could be gained from surveys, for instance, one survey of SIC for OFDM [72] and one survey of Power-Domain NOMA [73] which relies also on SIC. However, for the richer NOMA literature, the latter noted that "Although there exist works that analytically study the SIC error propagation in basic MIMO systems, there is no prominent research that provides a mathematical understanding of the effect of imperfect SIC on NOMA schemes." We reiterate their conclusion that this would constitute an interesting research direction, and in this chapter, we focus on our idealized SIC failure model in Eq. (5.1) as a first approximation that allows us to still capture the effects of SIC failures.

5.2.2 Performance Metrics and Objectives

Our goal is to maximize the number of users in the channel while minimizing the amount of packet loss. Note that we use the same notations as in Section 2.4. To assess the efficiency of the different parameters, several performance metrics are introduced. $G \triangleq \frac{M}{N}$ is the load of the system, in other words, the number of users per slot, which is to be maximized. Let S be the number of decoded users at the end of the iterative process, where $S \leq M$. Then, $T \triangleq \frac{S}{N}$ is called the throughput of the system. T is always less than or equal to G and $T = G$ if and only if all users have been correctly decoded. Finally, we have the packet loss rate (PLR) of the system:

$$PLR \triangleq 1 - \frac{S}{M}, \tag{5.2}$$

which is the proportion of non-decoded users at the end of the iterative process, that we aim to minimize. It is clear that we can also write $PLR = 1 - \frac{T}{G}$.

Since all packets of a user are may be lost due to SIC failure, there is a non-zero prob-

ability that a user's information is completely lost at the end of the iterative process. This has been observed in the literature studying IRSA or CSA in presence of channel losses, see [26, 38, 39] for instance. Thus, unlike IRSA or CSA without losses, there no longer exists a proper *load threshold* G^* below which all users' data blocks are retrieved with probability converging to one. To remedy for that, [26] defines the *asymptotic decoding threshold*, and equivalently [38] defines the *user recovery ratio*, that both help to redefine the idea of load threshold.

Following [38], the user recovery ratio α is also introduced in our system model, as a new parameter to be addressed. α is defined as the target ratio of decoded users at the end of the process, or in other terms, the wanted (minimum) probability that a user's data block is decoded successfully. Clearly, $1-\alpha$ is then the ratio of non-decoded users, i.e., the PLR. Then, we define a load threshold G^*_α, which is the maximum load such that a maximum ratio of $1-\alpha$ users is non-decoded. Mathematically, we define:

$$G^*_\alpha(\Lambda,\gamma) \triangleq \max\left\{G : 1 - \frac{T(\Lambda,G)}{G} \leq 1-\alpha\right\} \tag{5.3}$$

where γ is fixed. Λ is the main parameter to optimize in order to achieve the highest G^*_α, and IRSA optimization consists in solving the following optimization problem:

Problem 1 *Given $\alpha \in (0,1)$, $\gamma \in (0,1)$, find a degree distribution Λ^* that maximizes $G^*_\alpha(\Lambda,\gamma)$, that is:*

$$\Lambda^* = \arg\max_{\Lambda} G^*_\alpha(\Lambda,\gamma) \tag{5.4}$$

corresponding to [38, Eq. (4) and Eq. (5)].

In [26], the scheme of IRSA with captures has also been studied and a similar optimization problem to Eq. (5.3) has been solved to obtain the optimal repetition strategies that maximize the throughput given a target packet loss rate (PLR). In our work, we consider the interference cancellation failures that limit the number of maximum retrieved users and then, we extend it to derive the equivalent density evolution with a numerical approximation for finite frames. Furthermore, we provide a complete performance evaluation for given values of parameters such as the probability of SIC error supported by comprehensive simulations.

Notice that the system described above will be analytically studied when the frame size is infinite, i.e. $N \to \infty$, in the first place. We will then compare the derived analytic results to simulation results (in Section 5.5) which are obtained in a finite setting, where $N = 1000$.

5.3 Density Evolution Analysis

Note that the terminology and equations used in this section are inline with the basic density evolution equations in Section 2.4.3. We reformulate the following density evolution equations to fit the described scenario of IRSA with errors:

- Burst-perspective degree distribution $\Lambda(x) \triangleq \sum_\ell \Lambda_\ell x^\ell$
- Slot-perspective degree distribution $\Psi(x) \triangleq \sum_\ell \Psi_\ell x^\ell$
- Burst-perspective edge distribution : $\lambda(x) \triangleq \sum_\ell \lambda_\ell x^{\ell-1} = \frac{\Lambda'(x)}{\Lambda'(1)}$
- Slot-perspective edge distribution $\rho(x) \triangleq \sum_\ell \rho_\ell x^{\ell-1} = \frac{\Psi'(x)}{\Psi'(1)}$
- The rate of the distribution Λ: $R \triangleq \frac{1}{\sum_\ell \Lambda_\ell \ell} = \frac{1}{\Lambda'(1)}$
- We take a random edge, and we look at one of its connected bursts, the probability that this randomly chosen edge cannot be retrieved at the i-th iteration of the iterative process by applying SIC on its slot: p_i (see Fig. 2.3).
- We take a random edge, and we look at one of its connected slots, the probability that this randomly chosen edge cannot be decoded at the i-th iteration of the iterative process for the corresponding user: q_i.

Using Eq. (2.4), we see that $\sum_\ell \frac{\lambda_\ell}{\ell} = \frac{\sum_\ell \Lambda_\ell}{\sum_\ell \Lambda_\ell \ell} = \frac{1}{\Lambda'(1)}$, and we get from the edge perspective:

$$R = \sum_{\ell=2}^{L} \frac{\lambda_\ell}{\ell}, \tag{5.5}$$

which shows that R is linear in λ, that will be helpful for the formulation of a linear optimization problem. Note that L is the degree of the polynomial Λ, and that the index ℓ starts with the value $\ell = 2$, up to the value $\ell = L$, as in [1]. Finally, we define the average number of packets per slot as:

$$\nu \triangleq G\Lambda'(1) \triangleq \frac{G}{R} \tag{5.6}$$

In error-free IRSA, the goal is to maximize G such that one still has $\lim_{i\to\infty} p_i = 0$, and as a result $\lim_{i\to\infty} q_i = 0$ [1]. From the definition of p, a convergence towards 0 means all packets have been decoded with probability 1. As common in the IRSA and CSA literature (see for instance [2, 24, 26, 38, 68, 74]) and Section 3.2.3, we reuse some already known reasoning on the density evolution of p_i and q_i in similar contexts, but adapt them due to the changes in the settings, and obtain new density evolution equations (such as Eq. (5.11)). For this model of IRSA with decoding errors, the study of the convergence is similar but more complicated since we cannot guarantee the decoding of all users due to γ, as indicated in Section 5.2.2. Indeed, the definition of p is skewed by the error on SIC that makes the convergence towards zero impossible: there is a non-zero probability to fail to

retrieve all packets after SIC. We update the two equations from Eq. (2.9), also referred to as *extrinsic information transfer* (EXIT) functions, also in [68,75] which help to model the evolution of $\{p_i\}_{i\geq 0}$ and $\{q_i\}_{i\geq 0}$, and combine them to study the monotonicity of $\{p_i\}_{i\geq 0}$ and its convergence.

To retrieve a packet by applying SIC in a slot, all the other colliding packets must have been retrieved with SIC correctly carried out. Thus, by fixing ℓ, the number of packets colliding in a slot, and instead of Eq. (2.6), the probability to retrieve the last packet with SIC is:

$$1 - p_i = w_\ell (1 - q_i)^{\ell - 1} \tag{5.7}$$

as shown in [1, Eqn. (8)].

In our case, with the specific w_ℓ that we defined in Eq. (5.1), we have: $1 - p_i = (\gamma(1 - q_i))^{\ell-1}$, and we can obtain new closed-form expressions by taking the average over ℓ:

$$1 - p_i = \sum_\ell \rho_\ell (\gamma(1 - q_i))^{\ell-1} = \rho(\gamma(1 - q_i)) \tag{5.8}$$

that replaces Eq. (2.8).

The slot degree distribution $\psi(x)$ corresponds to the distribution of the number of users per slot when there are M users, each of them, transmitting an average of $\frac{\nu}{M}$ packets in one given slot. [2] It is the Binomial distribution $B(M, \frac{\nu}{M})$, hence: $\psi_\ell = \binom{M}{\ell}(\frac{\nu}{M})^\ell (1 - \frac{\nu}{M})^{M-\ell}$. Following a reasoning from the IRSA literature,[3] proposed in [1]:

$$\begin{aligned}\psi(x) &= \sum_{\ell=0}^{\ell=M} \psi_\ell x^\ell = \sum_{\ell=0}^{\ell=M} \binom{M}{\ell} \left(\frac{\nu}{M}\right)^\ell \left(1 - \frac{\nu}{M}\right)^{M-\ell} x^\ell \\ &= (1 - \frac{\nu}{M}(1 - x))^M\end{aligned}$$

[4]. When $M \to \infty$, while ν is fixed:

$$\lim_{M\to\infty} \psi(x) = \lim_{M\to\infty} (1 - \frac{\nu}{M}(1 - x))^M = e^{-\nu(1-x)}$$

Coincidentally, $\lim_{M\to\infty} \psi'(x) = \nu e^{-\nu(1-x)}$. Thus, we can deduce the following property for $\rho(x)$ using Eq. (2.4):

$$\lim_{M\to\infty} \rho(x) = \lim_{M\to\infty} \frac{\psi'(x)}{\psi'(1)} = \frac{\nu e^{-\nu(1-x)}}{\nu e^0} = e^{-\nu(1-x)}$$

[2] ν is the average number of replicas per slot from all users. It is divided by M, the number of *users*, to see how many replicas one user transmits per slot. We expect it is $\ll 1$

[3] It is actually an application of the classical Poisson approximation, i.e., the Poisson distribution with parameter $\mu = \alpha\beta$ can be used as an approximation of the binomial distribution $B(\alpha, \beta)$ for large α and small β. In our case, $\alpha = M, \beta = \frac{n}{M}$ and $\mu = n$.

[4] Note that: $(x + y)^n = \sum_{k=0}^n \binom{n}{k} x^{n-k} y^k$

This yields the approximation: $\rho(x) \approx e^{-\nu(1-x)}$, where $\nu = \frac{G}{R}$, as in [1, Eqn. (5)]. Note that this formula becomes exact when M tends to infinity.

Using Eq. (5.8), our first EXIT equation can be written as:

$$p_i = f_{\text{SIC},s}(q_i) \text{ with } f_{\text{SIC},s}(q) = 1 - e^{-\nu(1-\gamma+\gamma q)} \tag{5.9}$$

and $f_{\text{SIC},s}(q)$ replaces $f_s(q)$ in Eq. (2.9) and Eq. (2.10).

The second EXIT equation is unchanged and, as in Eq. (2.9) and Eq. (2.10), we have

$$q_i = f_b(p_{i-1}) = \lambda(p_{i-1}). \tag{5.10}$$

We will see its usefulness later. Again, the EXIT function Eq. (5.10) is the same as in error-free IRSA, whereas the EXIT function Eq. (5.9) is different: it takes into account the risk of erroneous SIC. We combine Eq. (5.9) and, Eq. (5.10) which yields our following new recurrence relation:

$$p_i = 1 - e^{-\nu(1-\gamma+\gamma\lambda(p_{i-1}))} \tag{5.11}$$

As usual, the convergence condition of the sequence is: $p_{i+1} < p_i, \forall i$. However, since there is a non-zero probability of failing to recover every edge due to SIC error, it is impossible to have $p_i \to 0$ for $\gamma < 1$. Thus, we define an error limit $p_{\min}$, as the expected limit of the sequence $\{p_i\}_{i\geq 0}$. Note that $p_{\min}$ is related to the average probability of failing to decode a packet in a slot at the end of the process[5]. The iterative process converges if and only if:

$$p > 1 - e^{-\nu(1-\gamma+\gamma\lambda(p))}, \ \forall\, p \in (p_{\min}, 1]$$

The convergence equation can be re-written as:

$$\lambda(p) < -\frac{1}{\gamma}\frac{R}{G}\ln(1-p) + \frac{\gamma-1}{\gamma}, \ \forall\, p \in (p_{\min}, 1) \tag{5.12}$$

The above inequality Eq. (5.12) is linear in λ since R is linear in λ, according to Eq. (5.5). As a consequence, it can be solved using standard linear optimization solvers. Notice that when $p \to 0$, Eq. (5.12) will lead to $-\frac{1-\gamma}{\gamma} > 0$ which is strictly impossible for $\gamma < 1$. This confirms that for fixed G and Λ, there is an interval for p close to zero where the inequality cannot be satisfied. This observation justifies mathematically that the limit of the sequence $\{p_i\}_{i\geq 0}$ is strictly greater than zero. It can also be noted that in the error-free case, the inequality is: $\lambda(p) < -\frac{R}{G}\ln(1-p), \forall p \in (0,1)$, which is a special case of our problem with errors: the optimal load of the system has to be lower in our configuration.

[5]We use the notation $p_{\min}$ in this chapter to emphasize that a minimal error limit is expected due to IC errors; later, we will use p_∞, e.g in Eq. (7.1).

An initial idea would be to fix the user recovery ratio α and find the corresponding optimal load G^*_α and code probability distribution, satisfying Eq. (5.3) and Eq. (5.4). Note that α and $p_{\min}$ are both related to the packet loss rate PLR. In this asymptotic setting, and recalling that α is defined as the target ratio of decoded users at the end of the process, there exists a direct mapping between α and $p_{\min}$ which is:

$$PLR = 1 - \alpha = \Lambda(p_{\min}) \tag{5.13}$$

The probability that a packet is not decoded after a certain number of iterations is given by the probability that all the edges connected to the corresponding user node are unrevealed. If a node has ℓ connections, such probability is p^ℓ. By taking the average on all possible node degrees: $PLR = \sum_\ell \Lambda_\ell p^\ell = \Lambda(p)$.

If we were to solve the optimization problem for a given α, we would need to reverse the function, λ, resulting in non-linear constraints. For this reason, we study the optimization problem, which consists in finding the optimal $G^*_{p_{\min}}$ and Λ for a fixed and given $p_{\min}$ as a parameter. Since $p_{\min}$ and the PLR are directly related by an increasing function Λ, we show, in the next section, how the PLR can be computed from $p_{\min}$. In Section 5.5, we will confirm by simulations that this relation is also valid for finite frames.

5.3.1 Linear Programming Formulation

We derive a linear optimization problem from the convergence equation Eq. (5.12), which is formulated as follows:

$$\begin{aligned}
&\underset{\lambda_i}{\text{Maximize}} && G && (\mathcal{P}_3)\\
&\text{subject to:} && C1 : \forall i \in [2, ..., L],\ 0 \le \lambda_i \le 1,\\
& && C2 : \sum_{i=2}^{L} \lambda_i = 1,\\
& && C3 : G\lambda(p) < -\frac{1}{\gamma} R \ln(1-p) + G\frac{\gamma - 1}{\gamma},\\
& && \quad \forall\, p \in\ (p_{\min}, 1)\,.
\end{aligned}$$

It is a refinement of the optimization problem $(\mathcal{P}_2)$ for 1-MPR described in the Section 3.3.4. The objective is to maximize the system load G. Constraints $C1$ and $C2$ are mandatory to make sure λ is a probability distribution, while $C3$ comes from Eq. (5.12). To limit ourselves to a finite number of inequalities, we use a technique introduced for low-density parity-check codes (LDPC) [76] which consists in taking a finite set of values of p, by discretizing the interval $(p_{\min}, 1)$ into steps of size s.

Since R is linear in λ, the feasibility problem (i.e., checking whether $C1$, $C2$ and $C3$ are feasible) is linear in λ. Increasing G leads to an increase in the left-hand term of $C3$, and there is a maximum load threshold $G^*_{p_{\min}}$ such that the problem is not feasible for $G > G^*_{p_{\min}}$.

We proceed by bisection on G: for each value of G, we check whether the problem is feasible and increase or decrease G accordingly. The pseudocode of the bisection algorithm is illustrated in the Algorithm 1 of the Section 3.3.4. We follow the same steps as in the Algorithm 1 except that we replace the problem ($\mathcal{P}_2$) with the problem ($\mathcal{P}_3$), that is mostly, the constraint C3. Through this procedure, the objective converges to the value of the maximum threshold $G^*_{p_{\min}}$. The solver returns an optimal λ, which is then mapped to the corresponding degree probability distribution Λ thanks to Eq. (2.4).

A common approach found in the literature to solve load maximization problems is to use the differential evolution heuristic. Differential Evolution (DE) is a population-based metaheuristic search algorithm that optimizes a problem by iteratively improving a candidate solution based on an evolutionary process. However, this method is not guaranteed to find the optimal distributions. Indeed, we see that Table 3.2 contains distributions (e.g., for $K = 1$ and degree $n = 8$ and $n = 16$) slightly outperforming the ones found by differential evolution in [1, Table I]. Another advantage of our linear solver is that the precision of the solution can be controlled by the number of bisection steps and the discretization granularity s.

5.4 Design of IRSA with SIC errors

5.4.1 Suggested Solutions of the Optimization Problem ($\mathcal{P}_3$)

We apply Python CVXPY linear solver [77] on the optimization problem ($\mathcal{P}_3$), and study the impact parameters to optimize the design of IRSA. The error tolerance at termination of the bisection search ϵ is set to, 0.001 while the discretization step s is set to 0.02.

The SIC efficiency γ is a parameter imposed by the system. We explore the impact of its influence in Fig. 5.1. We selected the reference value $\gamma = 0.99$ (and we are varying γ around this value, e.g. from 0.8 to 0.999): it is approximately one order of magnitude higher than the mentioned results in [1, Appendix B] and [68, Appendix A], whose physical layer simulations report a fairly low decoding error rate (10^{-2} and 10^{-3} after 7 SIC operations). It corresponds to one use case of IRSA for high-reliability communications (as in URLLC [78]). Considering only the impact of the imperfect channel state information (CSI), from [69] for instance, we know that the reliability of the SIC process can be improved to approach the performance of perfect CSI. This can be done by using a dedi-

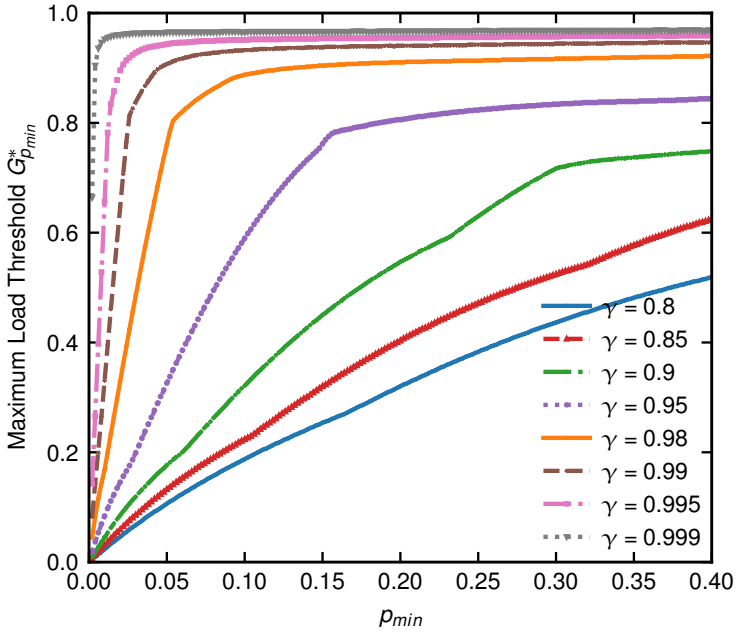


Figure 5.1: Influence of parameter γ: Maximum load threshold $G^*_{p_{\min}}$ versus $p_{\min}$ for IRSA unconstrained R and different γ.

cated pilots-based channel estimation with OFDM at the expense of having to operate at higher SNR.

The main parameter to study and to optimize is the error limit, $p_{\min}$, which is related to the *PLR* by Eq. (5.13).

Our first result is in Fig. 5.1. We plot the maximum load threshold G^* as a function of the parameter $p_{\min}$ for different values of the parameter γ under study, and fix all the other parameters to preset values: $\epsilon = 0.001$, $s = 0.02$ and $L = 15$.

Due to the relation between $p_{\min}$ and the *PLR*, the x-axis representing $p_{\min}$ can be seen as an increasing function of the packet loss rate. In Fig. 5.1, we study the influence of this SIC efficiency on the system performance. The curves have similar shapes, an increase in $p_{\min}$ leads to an increase in the maximum load $G^*_{p_{\min}}$ since the problem is less constrained. The curves also all have an inflection point, the load increases significantly below this point and saturates around 0.9 after, for higher γ. If we decrease γ, the maximal load threshold decreases and is reached for a higher $p_{\min}$.

In Fig. 5.1, $p_{\min}$ is the exact variable used when solving the optimization problem, and one can observe directly the impact of $p_{\min}$ on the performance of the system. For reference, we plot the relation between $p_{\min}$ and the PLR of the optimal solution for that $p_{\min}$ in Fig. 5.2.

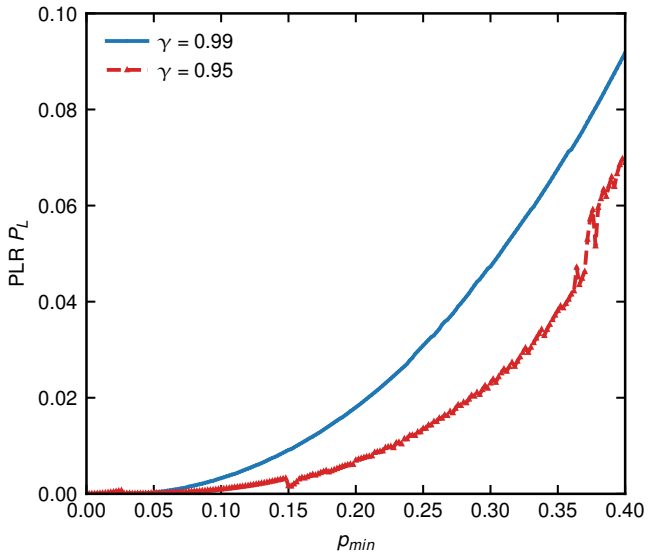


Figure 5.2: PLR of the solutions as a function of $p_{\min}$

5.5 Simulation Results with Optimized Distributions

In the previous section, the optimization problem helps us find various distributions $\Lambda_{p_{\min}}$ that provide different trade-offs between load threshold and error. The results are obtained considering density evolution is valid asymptotically when the number of users and slots tends to infinity. Now we want to confirm our results in a finite setting, by computer simulations. Although the *PLRs* found analytically are low, we observe orders of magnitude differences between them, and we will see that they are much more important and cannot be neglected when N is finite. For the following observations, we fix the number of users to $N = 1000$ and study the *PLR* as a function of the load of the system, averaged over 1000 simulations and set $\gamma = 0.99$.

Fig. 5.3 is the plot of the *PLR* as a function of the load in the case of IRSA for several distributions: $\Lambda_{0.02}$, $\Lambda_{0.05}$ and $\Lambda_{0.1}$ for $p_{\min} = 0.02$, 0.05 and 0.1, respectively, and also the optimal error-free distribution. The distributions Λ with high $p_{\min}$ perform better above a certain load threshold, corresponding to the optimal load of the distribution. For instance, the load threshold of $\Lambda_{0.1}$ is 0.85, and it is the load where this distribution starts to perform better than the others. However, when the load is below the threshold, the distributions for a lower $p_{\min}$ achieve a lower PLR P_L and have better performance. This confirms that the distributions are optimal only for a specific load: other distributions may be preferable when the load of the system varies. The optimal loads are also lower in the finite setting: around 0.85 for a tolerated *PLR* of 10^{-2} instead of 0.9. Finally, we can see that the optimal error-free distribution performs poorly: it is the best distribution for $G > 0.9$ but the *PLR*

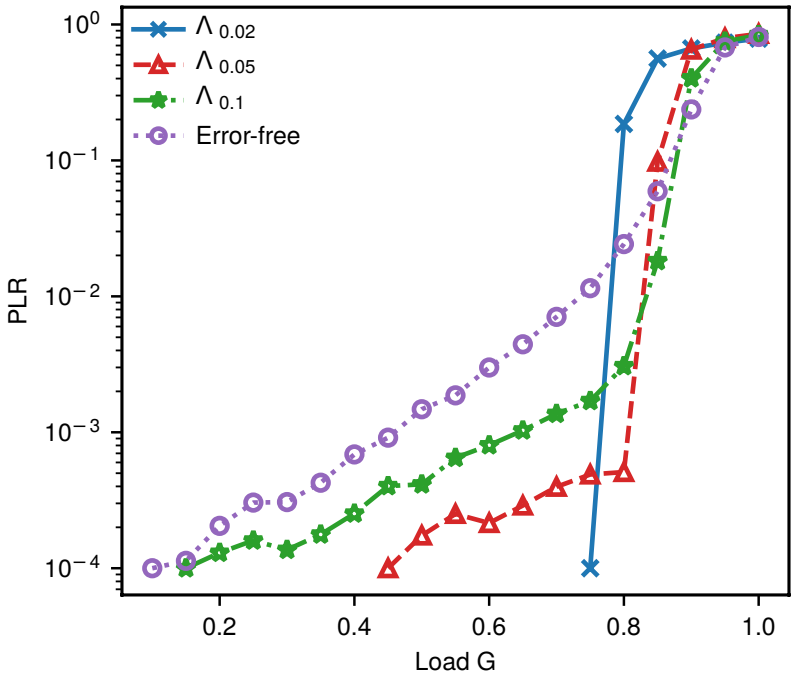


Figure 5.3: *PLR* versus load G for IRSA, $\gamma = 0.99$, $N = 1000$ and different distributions Λ.

is already above 10^{-1}. For a load below 0.9, the distributions $\Lambda_{p_{\min}}$ perform better. The relation between $p_{\min}$ and the *PLR* is also confirmed by this figure.

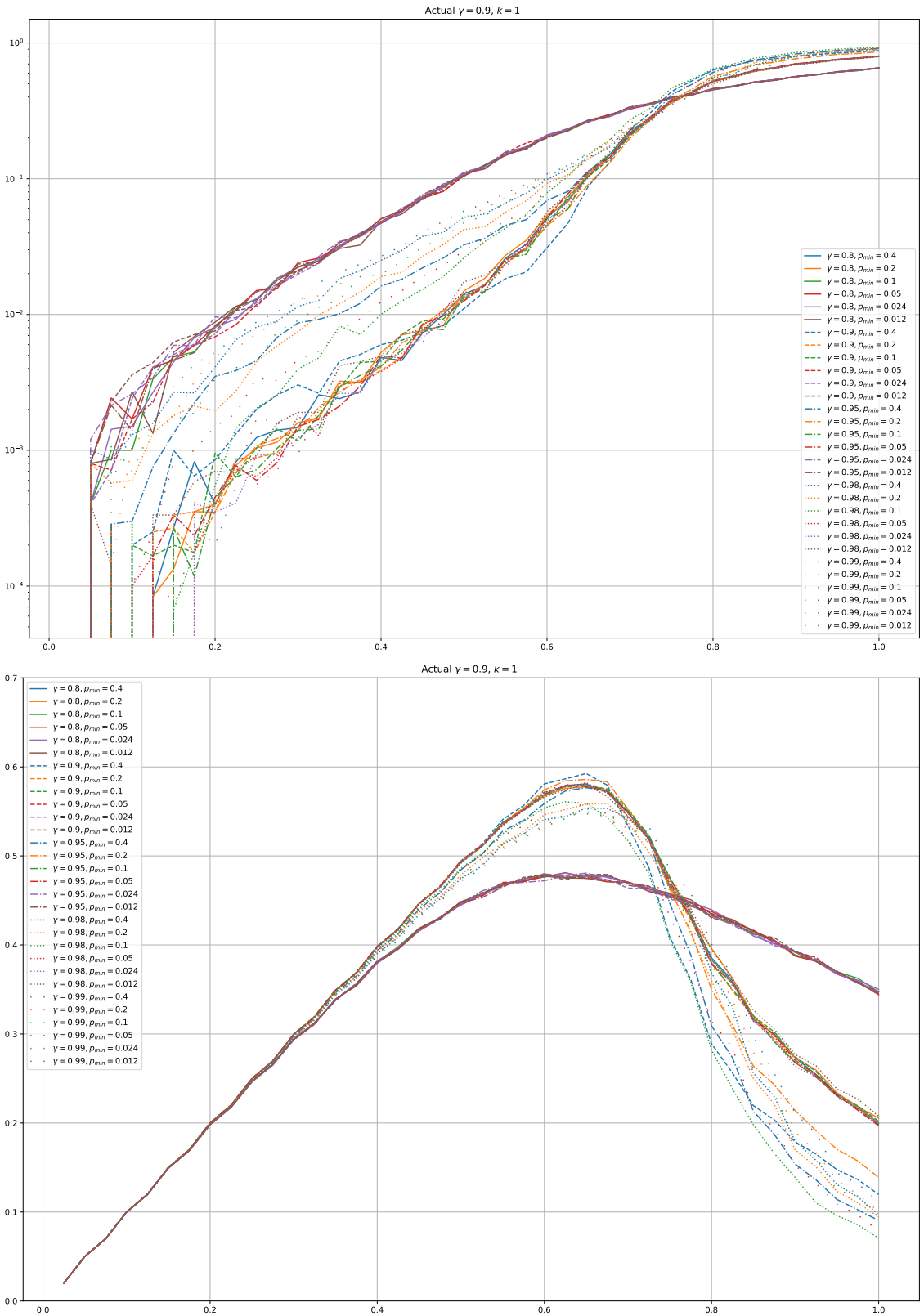


Figure 5.4: Systematic study of simulations for low $\gamma = 0.9$, with distributions designed with various γ and $p_{\min}$. The *PLR* is on top, and the throughput is on bottom.

To explore further the previous idea, a more systematic study of the behavior of the optimized distributions was done, when γ is set to a relatively low value $\gamma = 0.9$ and with smaller frame size, that is with $N = 100$. Different distributions are computed and used in the simulations. On purpose, they are not necessarily designed for the $\gamma = 0.9$ that is actually used in the simulations. By extensively varying $p_{\min}$ and γ, the intent is to have an arguably representative subset of all the optimal solutions that can be found by the problem ($\mathcal{P}_3$). The results are represented in Fig. 5.4. From the results of the throughput, one can identify roughly two sets of distributions: some with relatively good performance, and some with clearly lower performance. Naturally, the best distributions are the ones designed for the $\gamma = 0.9$, but as can be guessed, optimizing with a too low γ (i.e. $\gamma = 0.8$) or designing for a too ambitious $p_{\min}$ (low values such as $p_{\min} = 0.012$) are two reasons that equally result ultimately in lower performance.

For readability, Fig. 5.5 extracts only a few distributions from the large computed subset, and considers more values of the actual γ: $\gamma = 0.8, 0.9, 0.95$ and 0.99. We see that, of course, distributions designed with the proper γ outperform the others. For γ closer to 1, the effect of mismatch in the design of γ is less severe. Finally, the maximum throughput obtained by the different distributions, for the different γ are complementary results from the Fig. 5.1: we observe a maximum throughput between 0.5 (for $\gamma = 0.8$) and 0.7 (for $\gamma = 0.99$)

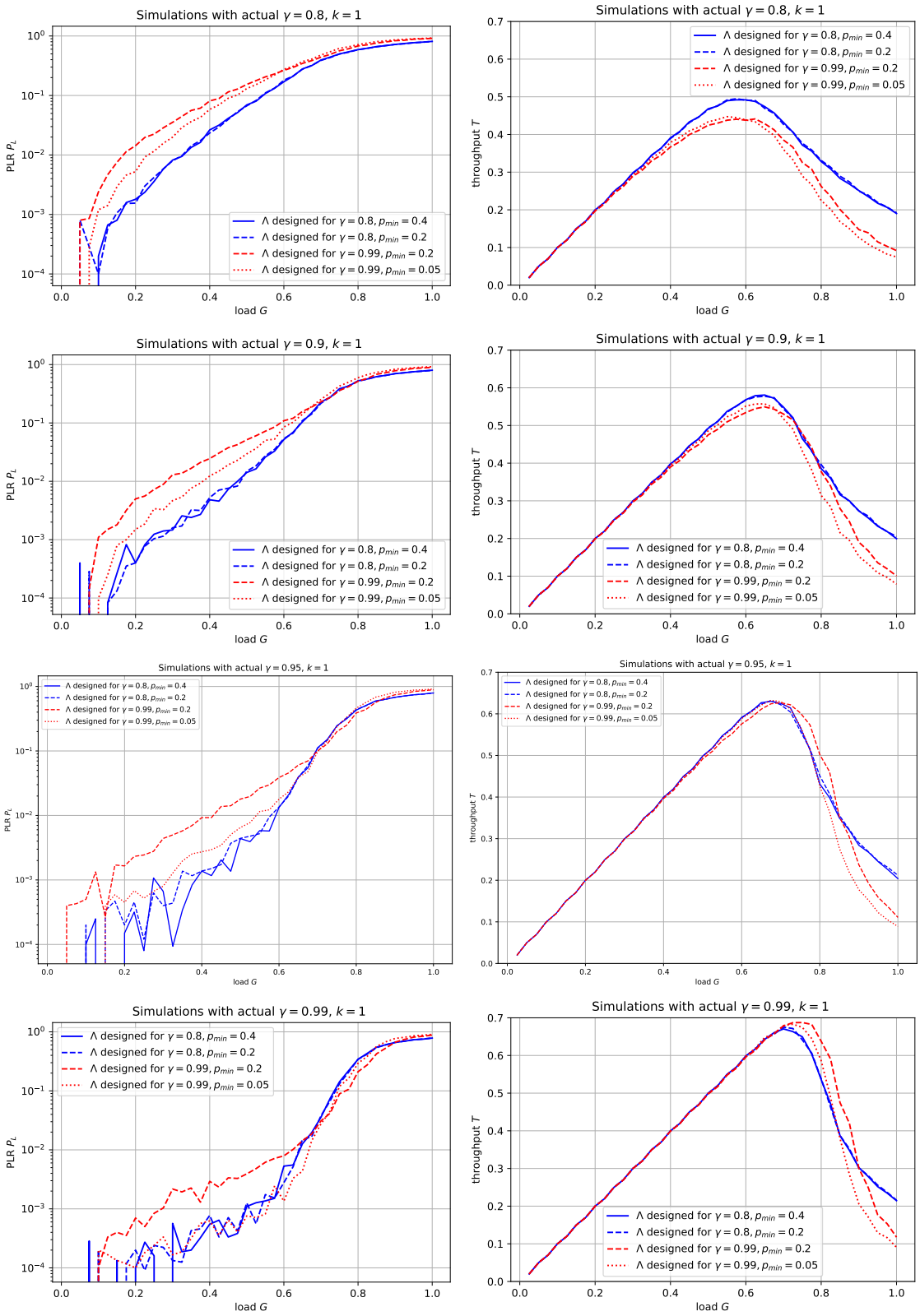


Figure 5.5: Influence of parameter γ: simulations with low γ, with distributions designed with various γ and $p_{\min}$.

5.6 Conclusion

In this chapter, we studied a novel extension of Irregular Repetition Slotted Aloha (IRSA), taking into account realistic effects, namely errors due to an imperfect SIC process. With these new assumptions and new modeling, we proposed a method to obtain the optimal distributions (in terms of maximum load threshold, or equivalently, throughput) for given values of parameters such as the probability of SIC error. This is accomplished by first deriving new density evolution equations, then formulating an optimization problem, that can be solved efficiently through different techniques that we proposed, including recourse to underlying linear optimization problems.

Then, starting from a preset system configuration, through numerically obtained distributions, we introduced a parameter $p_{\min}$, which allows us to extract new distributions $\Lambda_{p_{\min}}$, that each of them is optimal for a certain load G and a certain packet loss rate PLR. We studied the impact of the SIC efficiency γ on the maximum load threshold and the packet loss rate PLR. Choosing a distribution with a high load threshold is at the expense of having a high PLR as well. These PLRs are dramatically increased if SIC error is taken into account.

Chapter 6

A Regret Minimization Approach to Irregular Repetition Slotted Aloha

6.1 Introduction

During the last decade, there has been a trend to use Machine Learning (ML) applications for wireless networks [79]. Machine learning has succeeded in getting academic and industrial attention as a promising collection of methods that allows devices to be able to learn, automate and optimize without human intervention. Recently, promising approaches of ML have been proposed to enable intelligent features to 5G. Indeed, the promise is that ML approaches allow to automatically learn the system parameters, predict future scenarios, and adapt to dynamic environments [80]. Machine learning has been proposed as an efficient tool to allow nodes to learn how to coordinate their transmissions such that the network throughput is maximized. ML approaches are classified into three main categories, Supervised Learning (SL), Unsupervised Learning (UL), and Reinforcement Learning (RL). Other secondary machine learning approaches such as Markov models, Heuristics, Controllers can also be found in use with 5G applications. In [80], the applications of the tools of each category of ML in wireless networks, specifically in cellular networks, have been surveyed.

Reinforcement learning (RL) is a method by which a learning agent learns from the consequences of its actions in a real or simulated environment, by means of rewards and penalizing. RL has been widely applied to ALOHA [81–83] and slotted ALOHA [84] mainly to avoid collisions, improve the throughput and increase energy efficiency. RL methods have been also applied to mitigate collisions in NOMA systems and improve the system throughput [85, 86].

Motivated by the importance of RL as a promising solution to improve transmission strategies for MAC layer protocols, in this chapter, we present a reinforcement learning approach for optimizing the performance of IRSA protocol. The main objective of this work is to address the IRSA access scheme according to a centralized approach. Users are grouped in classes of different priorities, where users of one class share the same degree distribution. In this centralized approach, global optimization of degree distributions is computed offline on a central controller, namely the base station. This centralized optimization problem is typically non-convex. We adopt one specific variant of a method related to reinforcement learning, Regret Minimization, to learn the protocol parameters. We explain why it is selected, how to apply it to our problem with centralized learning, and finally, we provide both simulation results and insights into the learning process. The results obtained show the excellent performance of IRSA when it is optimized with Regret Minimization.

6.2 Related Work

In the literature, several research directions have addressed topics that are related to modern random access protocols. Naturally, there also exists an extensive literature on random access protocols themselves that date back over several decades. In this section, we focus on modern random access, and on research studies that applied various reinforcement learning techniques. We identify the following related topics where machine learning techniques have been used: cognitive networks with spectrum sensing, classic random access protocols in IoT networks, and finally, more specific machine learning approaches to protocols of the IRSA family itself, or NOMA-based protocols. In the following, we describe some of the related articles that have covered these topics.

In cognitive networks, Dynamic Spectrum Access is a wireless network paradigm where the users exploit their knowledge of the environment to successfully access a shared medium and maximize their throughput. The problem of dynamic spectrum access for wireless networks has been recently explored with machine learning techniques in [87] and [88]. It is shown that the problems of joint user association and spectrum access are typically combinatorial and non-convex, and require near-complete and accurate information to obtain the optimal strategy.

According to [89], developing efficient learning approaches to optimize medium access has been the center of attention of many research works. In particular, Deep Q-Learning (DQN) provides promising solutions for the Dynamic Spectrum Access problem, specif-

ically, for IoT networks. In [90], a novel distributed dynamic spectrum access algorithm based on deep multi-user reinforcement learning (DRL) has been proposed. The users transmit over shared channels using a random access protocol. Time is slotted, but no SIC is used in the receiver to resolve collisions. In the proposed approach, every user maps their current state (the history of selected actions and past network state observations) to a certain action (use of a shared orthogonal channel) based on a trained deep-Q network. Their objective is to maximize a certain network utility. The used utility functions are the user sum rate, and it is also competitive in the sense that each user aims to maximize its rate. The proposed algorithm enables the user to learn good policies in an online, distributed manner.

The authors of [91] address the problem of collisions and idle time of random access protocols by designing a fully distributed IoT protocol to improve the device access to the shared medium. The proposed online learning scheme is based on designing optimized dictionaries of transmission patterns to avoid collisions between users. The dictionary contains a subset of the possible binary vectors of a length equal to the total number of slots in the frame. The goal is to select an optimized set of transmission patterns from the dictionary, where the dictionary is common to all users and the probabilities to use the transmission patterns. The scheme provides some URLLC guarantees for IoT applications that require the same time latency, energy efficiency, and low coordination overhead.

Dynamic multi-channel access was also considered in [92], where the user selects a channel, at each time slot, from multiple correlated channels. Each user can observe the state of the chosen channel only at a given time slot, which means that the current state of the system is not fully observable, hence the problem is modeled as a Partially Observable Markov Decision Process (POMDP). The study aims to design an adaptive DQN framework that can adapt to time variations and maximize the long-term expected reward for each user.

Another work direction for designing efficient IoT protocols is to enhance existing MAC protocols so that they fit the new requirements of IoT networks. Optimizing the performance of MAC protocols has been addressed in many research studies over the last forty years. Some of these studies have introduced machine learning techniques to variants of the ALOHA protocol family. A novel Q-learning based on Informed Receiving Protocol has been introduced in [93]. ALOHA-QIR provides some intelligence to the nodes to access the slots that have a lower probability of collision. The nodes keep hoping to different slots to learn the optimum ones. In this ALOHA variant, the nodes keep listening during the hopping while the receiver is informed by the preferred slots of each node by sending

"ping packets", so the receiver can turn them off when needed. The classical Q-learning algorithm with a simple reward design (±1) is used to learn the optimum slots to select. The proposed approach helps to achieve over twice the maximum throughput of Slotted ALOHA.

In [94], the IRSA protocol is optimized using online learning. By considering the base station as the decision-maker, the performance of IRSA is optimized by maximizing a utility function that reflects the number of decoded packets. The problem of optimal resource allocation (slots allocation) is formalized as a Multi-Armed-Bandit (MAB) problem. The authors use the Bayesian UCB algorithm to solve the MAB problem and compare it with other commonly used methods. The degree distribution is also optimized by fixing the degrees and optimizing the probabilities of selecting them (e.g., of the form $\Lambda_2, \Lambda_3, \Lambda_8$). In [95], finite length IRSA is also optimized through a Q-learning approach.

In the next section, we present the IRSA model with a reinforcement learning approach to optimize its performance.

6.3 System Model and Assumptions

6.3.1 System Description

We consider IRSA as an access protocol for users (devices) sharing a communication channel to a single base station. The access time of the channel is divided into slots of equal duration. The duration of the slot is equal to the time needed to transmit a packet (including propagation delays, etc.). In this system, however, we assume that a *frameless IRSA* is used (as in [74]), as opposed to the framed, classical version of the IRSA protocols (as the classical model that we have presented in section 2.4). In framed IRSA, there is a frame of a predefined length, where each user randomly selects slots, and at the end of which the decoding is performed. In frameless IRSA, there is a very large set of slots, potentially infinite. In our model, for practical reasons, this set of slots always has a fixed size of M slots, and it is called a contention round. We divide the contention round into virtual frames, where each virtual frame size is much shorter than the contention round (in our simulation it is 30% of the contention round). When the user decides to send a packet, the user is associated with a virtual frame. The active user sends the replicas of its packet only during the virtual frame period. The goal of introducing the virtual frame is to facilitate the decoding process and rewards computations, which will be explained more precisely in the next section.

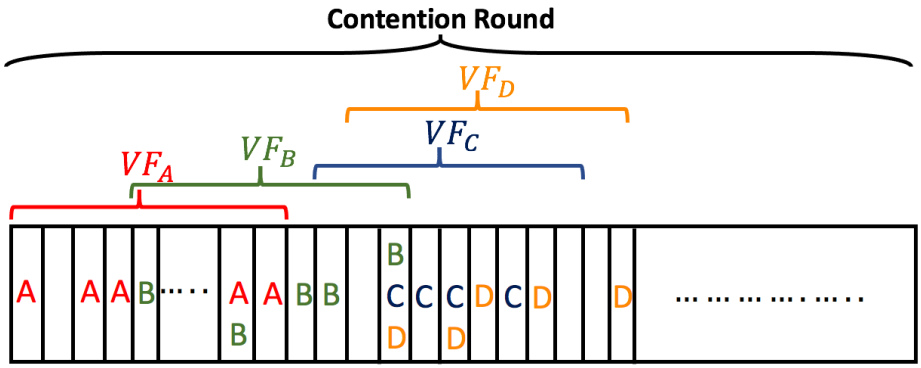


Figure 6.1: IRSA frameless structure

At each time slot, the number of active users is determined by a Poisson arrival rate μ (e.g. the number of active users on one slot is a random variable N_a with distribution $\Pr(N_a = k) = \frac{\mu^k e^{-\mu}}{\mu!}$). In our case, to be consistent with the literature and with the rest of the thesis, the arrival rate μ is also denoted network load G.[1] It is the average number of active users on one slot. Each active user selects a repetition degree to use from a set of multiple allowed degrees which are identical for all users. At the base station, SIC is used to resolve the collisions. Figure 6.1 shows the frameless IRSA structure with all active users and their associated virtual frames where they are allowed to send their packets. Virtual frames can overlap and the transmissions from different active users can cause collisions, as seen in the figure. Unlike the classical IRSA decoding, (explained in section 2.4.2), the base station performs online decoding by starting a decoding iteration after each received slot, instead of waiting for the whole frame to end and then starting the decoding process.

6.4 IRSA-RM: IRSA Based on Regret Minimization

6.4.1 Problem Formalization

We adopt the frameless IRSA structure (explained in section 6.3). For simplicity of presentation, we formulate the problem with two classes of users. The two classes have different access priorities. The users of the same class share the same degree distribution ($\Lambda_{i,c}$). The base station uses SIC to perform the slot-by-slot online decoding. Our objective is to find the best degree distribution for a known Poisson arrival rate $\mu = G$, that maximizes the weighted throughput of both classes. Formally, our problem could be written as an optimization problem derived from the problem ($\mathcal{P}_1$):

[1] Note that, in this chapter, we consider that the arrival rate of active users follows a Poisson distribution, while it is fixed in the rest of the thesis

$$\begin{aligned} \underset{(\Lambda_i)}{\text{maximize}} \quad & \alpha_{C_0} T_{C_0}(\Lambda_{C_0}) + \alpha_{C_1} T_{C_1}(\Lambda_{C_1}) \qquad (\mathcal{P}_4) \\ \text{subject to} \quad & 0 \leq \Lambda_{C_i,j} \leq 1 \ \forall j \in [0, 1, 2, .., D] \\ & \sum_{j=0}^{j=D} \Lambda_{C_i,j} = 1 \end{aligned}$$

Where: T_{C_i} is the throughput of the class C_i, which measures the average number of decoded packets of this class. $\alpha_{C_0}, \alpha_{C_1}$ are constant weights indicating the importance of the throughput for each class.
One can think of solving such a problem by using the DE tool since frameless IRSA transmission and decoding using SIC can still be represented by a bipartite graph [74]. Since the BS performs the decoding for each slot, a part of the bipartite graph will be available at any time, thus the density evolution equations will not necessarily represent the decoding state in the middle of the contention round. Because classical DE [1] is valid only asymptotically and the finite length analysis can be computationally expensive, we adopt another direction in this chapter, for this variant of IRSA.

In the next section, we propose a new learning framework for optimizing the transmission strategy of frameless IRSA. We consider a method of offline learning. We assume multiagent settings, and we apply the method of Regret Minimization, where each user wants to minimize its regret by taking better next decisions.

6.4.2 Reinforcement Learning Approaches and Regret Minimization

In this section, we use *Reinforcement Learning* (RL) to find good solutions of the problem described in ($\mathcal{P}_4$). A classic reference on reinforcement learning in general is [96]. As in many network problems, the decisions taken by one node, device, or one user can be modeled as a Markov Decision Process (MDP) [96, section 3]: each participant in the network is an *agent*, that makes decisions, denoted as *actions*, based on some current *state* from the environment. *Rewards* for each taken action are computed and are used to adjust the future choice of actions. Classical algorithms such as Q-Learning [96, section 6.5], Multi-Armed Bandits [96, section 2], and others, are well-known examples of such reinforcement learning algorithms.

Applying those to random access introduces several challenges: the first one is that there are several agents instead of just one (*Multi-Agent* Reinforcement Learning, MARL, see [96, section 15.10]); the second one is that, by definition of random access, each agent

only knows part of the network state, if only because it does not know the actions of other agents (Partially Observable Markov Decision Process, POMDP, see [96, section 17.3]).

Learning in a multiagent setting is indeed a complex task: the impact of the decision taken by one agent may depend on the decisions taken by other agents in the system. Thus, first, classical RL approaches for a single agent can create difficulties, such as non-stationarity and oscillations when applied to multiagent systems. Second, controlling multiple agents poses additional challenges compared to single-agent systems such as the definition of the collective goal of the agents, the heterogeneity of the agents, the ability to operate with many agents, and partial observability [97].

Frameless IRSA is such a multi-agent system, subject to partial observability. Numerous learning approaches have been proposed in the literature to handle POMDP, including Deep Reinforcement Learning (DRL) [97].

Many of the algorithms proposed in the literature lose their proof of convergence in a MARL setting, and there does not necessarily exist a general theory characterizing the cases under which every MARL algorithm is successful [98]. Their convergence or non-convergence dynamics is a topic of study by itself, with also strong links with game theory [98, 99]. Indeed, while applying Q-Learning to frameless IRSA, we experienced non-convergence, which led us to select an algorithm whose multi-agent dynamics have been well studied: Regret Minimization (RM) [100].

Regret Minimization is an algorithm where each agent maintains a set of *weights* for actions. Once normalized, the weights indicate the probability that the agent selects each action. At a given time, after the action selection by one agent according to weights, each such action i changes the environment state and has a corresponding reward which is provided by the environment. At the same given time, an optimal action could have been played by the agent instead, which would have resulted in an optimal reward r. The difference between the optimal reward r_{opt} and the actual reward r_i gives the *loss* of the agent at that given time: $\ell_i = r_{opt} - r_i$, which is a measure of *regret* for selecting action i. The Polynomial Weights algorithm is one of the Regret Minimization algorithms that assign weights for each action and uses the "loss" concept to update the weights after each play-

ing round [100]. Formally, it is as follows [100, page 13]:

$$
\begin{aligned}
&\text{Initially: } w_i^{(1)} = 1 \text{ and } p_i^{(1)} = \frac{1}{|X|} \text{ for } i \in X \\
&\text{At time } t-1\text{: an action } i \in X \text{ is selected according to } (p_j^{(t-1)})_{j \in X} \\
&\qquad\qquad \text{the reward of action } i \text{ is computed: } r_i^{(t-1)} \\
&\qquad\qquad \text{the potential reward of the best action is: } r_{opt}^{(t-1)} \\
&\qquad\qquad \text{the loss is computed as: } \ell_i^{(t-1)} = r_{opt}^{(t-1)} - r_i^{(t-1)} \\
&\text{Update for time } t\text{: } w_i^{(t)} = w_i^{(t-1)}(1 - \eta \ell_i^{(t-1)}) \\
&\qquad\qquad p_i^{(t)} = \frac{w_i^{(t)}}{\sum_{i \epsilon X} w_i^{(t)}}
\end{aligned}
\tag{6.1}
$$

with:
X: the set of possible actions.
$w_i^{(1)}$: the initial associated weight to the action i.
$w_i^{(t)}$: the associated weight to the action i at time t.
$p_i^{(1)}$: the initial probability of using an action i among the $|X|$ actions available.
$p_i^{(t)}$: the probability of using an action i at time t.
η: the learning parameter (akin to a learning rate).

The weights update is based on two main parameters: the learning parameter to control the speed of the weight changes, and the loss, which specifies the impact of the played action by computing how far the played action was from optimality. The weights of the actions are used to compute the probability of using each of the actions in the next playing round.

Notice that richer variants of Regret Minimization have been proposed, such as Counterfactual Regret Minimization (CFR) [101]; with IRSA, they might be suited for agents with richer interactions, for instance, agents taking decisions on the transmission of each replica (instead of selecting a degree once).

6.4.3 Applying of Regret Minimization to Frameless IRSA

Returning to our initial problem, we assume that the network consists of users competing in the same slotted wireless channel to transmit packets towards one base station using the frameless IRSA protocol. Users are grouped into classes of different priorities. The users of one class also share the same degree distribution. As mentioned above, each user has partial observability about the network, because it does not know on which slots the other agents are transmitting (nor about collisions). However, they have additional information:

an important assumption is that the base station is maintaining a discretized estimate $\bar{G}$ of the load $G = \mu$ (the agent Poisson arrival rate) in the system and broadcasting it to each agent.

Our objective is to maximize the total throughput of users, for each given network load G; where the throughput of each class is weighted by a different factor (so that some classes carry more weight, as a priority mechanism). We assume that each active agent, who has a packet to send, is associated with a virtual frame to send the packet and its replicas. It is interesting to look at the base station perspective: it observes singletons on some slots, collisions on some other slots, and performs SIC for each packet already decoded.

We adapt the Polynomial Weight RM algorithm detailed in Eq. (6.1) to solve our problem. To emphasize the learning aspect, here, the term "agent" will be used as an equivalent to "user". In order to map the problem features to RM, we have the following assumptions and system model:

• A centralized *offline* learning approach based on Regret Minimization is considered. Numerous simulations or *episodes* (as in Q-Learning) are run. Each episode corresponds to a long contention round. It is intended that after learning has finished, the weights could be used in an actual network, or in our case, are actually used as node degree distributions Λ in further simulations without learning (see Section 6.5).

• The base station is assumed to broadcast a discretized estimate $\bar{G}$ (with a finite number of possible values) of the actual load G: currently $\bar{G}$ is the measured average number of users per slot since the beginning of the contention round.

• The action of each agent is: selecting the number of repetitions (the degree) for their packet.

• As per Polynomial Weights RM, each of the agents maintains weight tables (denoted w) which are used to compute the probability of selecting each action, akin to a probability distribution Λ. We extend it: one different table of weights is used depending on some state. The agents consider the load estimate given by the base station as the environment state, and it is discretized to constitute a finite set of possible states. For each different discretized load estimate, the agent uses and updates a different set of weights $(w_i(\bar{G}))_{i \in X}$.

• Additionally, the agents of the same class are sharing the same weight tables (w table) in the learning process. Updates of the weights after each selected action are thus shared within agents of one class[2]. The goal of using the same w table for all the agents of the same class is to drive the agents inside one class to act cooperatively, and to work in a

[2]Note that then the learning also behaves as if one class would be one agent by itself. The algorithm and our implementation, also works with non-shared tables.

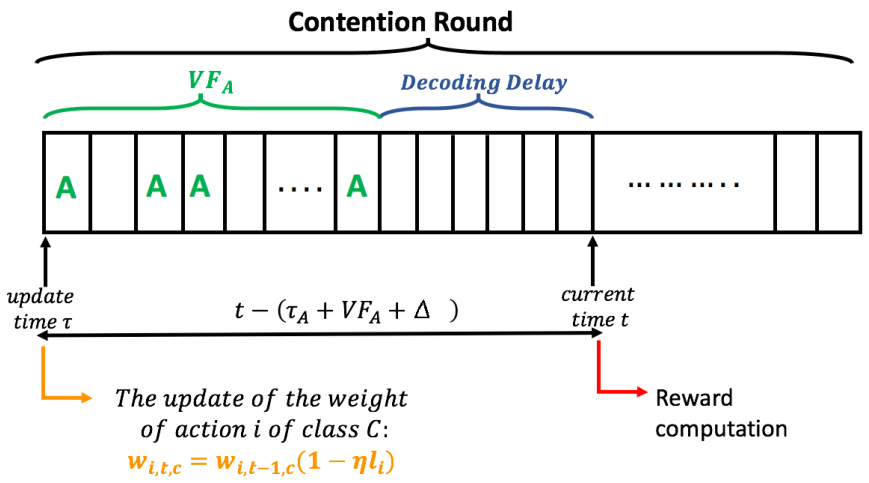


Figure 6.2: Reward computation and update delays

coordinated manner towards the collective goal. This may depart from usual assumptions in RM and evolutionary dynamics.

• At the moment an agent selects a degree, the results of this action are unknown until some time has elapsed (see Section. 6.4.4). Thus, an approach with delayed reward computation and updates similar to *n-step Sarsa* [96, Section 7.2] is used.

• The main challenge for applying the Polynomial Weight algorithm is to compute the loss. The loss computation is directly related to the rewards' calculation. As our goal is to optimize the joint throughput, we opt to directly link the rewards to the number of decoded agents in each class and set "reward = number of decoded agents". Defining an IRSA reward is otherwise difficult.[3]

• The number of decoded and non-decoded users is available at the global simulator level during our centralized learning process.

We further detail delayed updates and reward computation in the following sections.

6.4.4 Delayed Updates

Each agent sends within its virtual frame size and the base station decodes slot by slot. Therefore, the base station needs to wait, at least, for the end of the virtual frame to decide if one agent can be decoded or not. Hence, to accurately compute a reward, one delay needs to be introduced: this is illustrated in Fig. 6.2. As shown in the figure, agent A starts to be active at time τ and sends its packets using the action i, e.g. sending i replicas, ($i = 4$ in this case), within the associated virtual frame spanning the time from τ to $\tau + VF_A - 1$. Only at time $\tau + VF_A$, one can be certain that all replicas of A have been sent. But decoding

[3]Due to the credit assignment problem.

can be further delayed: during this virtual frame, other agents could become active and transmit in overlapping virtual frames (possibly shifted in time, see Fig. 6.1) and could induce collisions that need to be resolved in order to recover one of the replicas of A. But in turn, those other agents might collide with agents whose virtual frames are occurring even later, etc.[4] For practical purposes, an additional *decoding delay* denoted Δ has to be introduced after which the base station would consider the slots definitely non-decodable. As a result, in our simulator, we compute the rewards of an action selected at time τ, only at time $t = \tau + VF_A + \Delta$, and perform the RM weight update at that time. This is similar to n-step Sarsa [96, section 7.2], with $n = VF_A + \Delta$.

6.4.5 Reward and Loss Computation

We assume that there are two classes C_0 and C_1, and that the class C_0 always has a higher priority than the class C_1. This priority difference is introduced in the reward computation of each class. As the base station decodes the slots up to time t (see Fig. 6.2), it computes the number of decoded agents of each class up to the time t. The associated reward of an agent A that played an action i at the time τ is computed at the time t as follows:

$$\begin{aligned} r_i(A) &= P_{C_0,t} \qquad \text{if} \qquad A \in C_0 \\ r_i(A) &= \alpha P_{C_1,t} + (1-\alpha) P_{C_0,t} \qquad \text{if} \qquad A \in C_1 \end{aligned} \tag{6.2}$$

where:
$r_i(A)$: is the associated reward of action i from the agent A in the class C.
$P_{C,t}$: is the number of decoded packets of agents of class C up to time t.
α: is the parameter that weights the priority of classes.

From Eq. (6.2), notice that as α is smaller, the priority of class C_0 is higher. Notice also that the reward of each agent is computed based on the collective amount of decoded packets of all agents (of the same class), and hence they act cooperatively. This is in opposition to the selfish behavior of the agents if the reward was based exclusively on the individual performance of each agent.[5]

Now, more importantly, in IRSA, reward computation is difficult, because the decoding process is iterative: it is difficult to assert if an individual action is responsible for undecoded packets. A straightforward reward is used here: essentially the number of decoded packets (or a function of it). In other RL algorithms, this would not work, as the reward

[4] It is indeed possible to construct a frameless IRSA scenario where one user can be only decoded after an arbitrarily large delay.

[5] Note that, with respect to Problem ($\mathcal{P}_4$), this corresponds to: $\alpha_{C_1} = \alpha, \alpha_{C_0} = 1 - \alpha$

would grow linearly with time. In RM, however, only the loss (regret) is used for the updates, and it is the difference of reward between the best action and the taken action. In our case, the loss translates as the number of packets that the taken action had prevented from being decoded, which is exactly the meaningful information.

But then in addition to computing the actual reward using Eq. (6.2), corresponding to the played action i, it is necessary to compute the optimal reward that the agent could receive if it played the optimal action at the time τ. This has a cost and increases complexity. In the case of a single-agent system, with no delay in update computation, the optimal reward can also be computed at a time t by trying all possible actions at a playing time $t-1$ and considering the action that yields the maximum reward as the optimal action to take at the time $t-1$. If the action space is large, this process could already be costly.

However, it is more complicated in a multi-agent system where 1) reward computation is delayed (here: by necessity), 2) where other agents are also interacting in the environment in the interval between the action of one node and its associated reward computation. To handle this, in practice, we maintain one main simulation where each selected action by

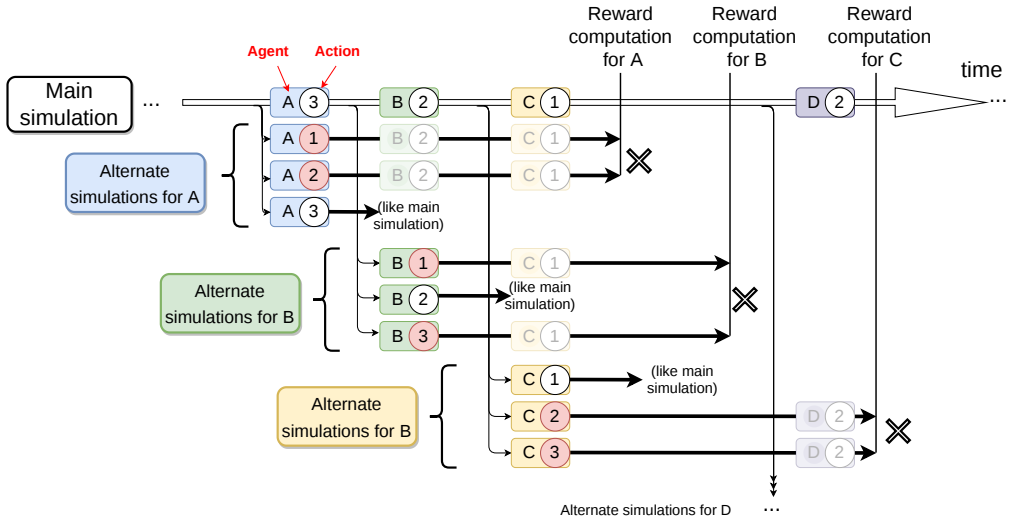


Figure 6.3: Alternate simulations for agents A, B, C, and D

one agent is performed. But we also maintain *alternate simulations* (equivalent to "alternate realities" in mundane terms), that differ from the main simulation only by one action of one agent. Each action of an agent indeed results in creating one new associated alternate simulation for each of its other possible actions (initialized as a copy of the main simulation). At the time of the reward computation for the agent, the reward is computed in each of its alternate simulations: since in its alternate simulations the only difference is the action of the agent (not those of other agents), the difference of reward between different

actions can be immediately ascribed to the actions themselves. Fig. 6.3 illustrates alternate simulations, in a scenario where 3 actions $1, 2, 3$ are possible, and where agent A selects action 3, agent B selects action 2, agent C selects action 1, and agent D selects action 2 in the main simulation. The alternate simulations correspond to simulations where one agent selects each of the 2 alternate actions.

Consider an agent A of class C that selected an action $i \in X$ at a time τ. Its optimal reward is computed as follows:

$$r_{opt}(A) = \max_{j \in X} r_j(A) \text{ with } X = \{0, 1, \ldots, D\} \tag{6.3}$$

And the loss of playing an action i, by an agent A at time τ is computed using the following equation:

$$\ell_i(A) = \frac{r_{opt}(A) - r_i(A)}{N} \tag{6.4}$$

where:
r_i, is the associated reward of action i, which is computed using Eq. (6.2) and the knowledge of the class C of node A.
N: is a normalizing factor, taken to be the total number of users in the system.

We summarize the design of our offline regret minimization-based learning algorithm: it is an adaptation of n-step Sarsa [96, section 7.2], where the Q table update is replaced by the weight update from the Polynomial Weights Regret Minimization Eq. (6.1), and where agents of the same class, share the same weights.

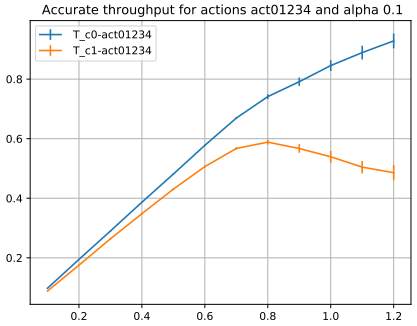
6.5 Numerical Results

In this section, the performance achieved by IRSA-RM as a random access MAC protocol is illustrated through simulations. There are two phases: first, the learning phase, whose objective is to obtain good degree distributions; second, the performance evaluation of these distributions, as is common in IRSA evaluation.

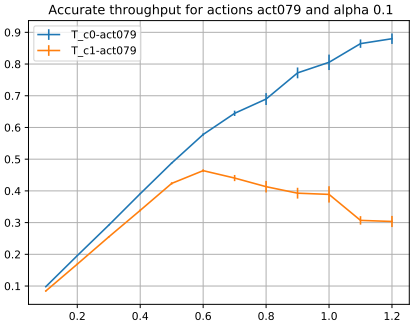
We developed our own simulator for IRSA and RM. For all results, a contention round of $M = 500$ slots is used, the virtual frame size is set to $VF = 150$ slots, while the decoding delay is $\Delta = 50$ slots. For all simulations, both classes have equal arrival rates. The maximum possible degree (action) is $D = 10$. Different cases of class priority are studied; the results are always obtained for two classes, and two different values of the priority

parameter: $\alpha = 0.1$ and $\alpha = 0.3$ for Eq. (6.2). In both cases, the class C_0 has a higher priority than the class C_1.

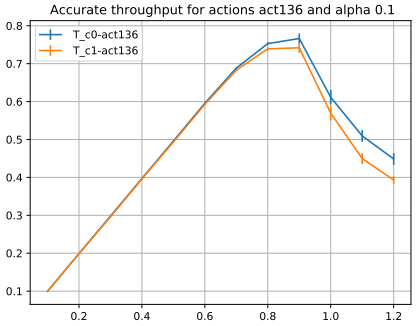
We start with the learning phase, whose objective is to find good degree distributions with respect to the described optimization problem in $(\mathcal{P}_4)$, interpreted through Eq. (6.2). In this phase, agents are restricted to one fixed subset X of actions from the set of all possible actions $X \subset \{0, 1, 2, \ldots, D\}$. Several such subsets are selected. For each action subset, several learning processes are run: the total user Poisson arrival rate $G = \mu$ is fixed during each of them, and one learning process is run for each G taken from 0.1 to 1.2 with step 0.1. At the end of each learning process, for each class C, the RM algorithm yields some weights $(w_{C,i}(\bar{G}))_{i\in X}$ from which probabilities of selecting actions are derived $(p_{C,i}(\bar{G}))_{i\in X}$ which are directly interpreted as lambda distributions (e.g. $\Lambda_i^{RM}(X, C, \bar{G}) \triangleq p_{C,i}(\bar{G})$). Each learning process is run for $E = 5000$ episodes, and the learning rate η is set to 0.04.



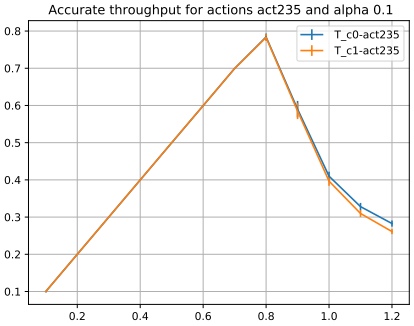
(a) Achieved scaled throughput of both classes, actions {0,1,2,3,4}



(b) Achieved scaled throughput of both classes, actions {0,7,9}



(c) Achieved scaled throughput of both classes, actions {1,3,6}



(d) Achieved scaled throughput of both classes, actions {2,3,5}

Figure 6.4: Scaled throughput comparison between different set of actions and an external distribution ($\Lambda_2 = 0.5, \Lambda_3 = 0.28, \Lambda_8 = 0.22$)

We then compute the performance when applying the obtained distributions. Our main metric is the throughput, e.g. how many decoded packets are recovered per slot. We evaluate the throughput for different loads: but for these simulations, the load does not represent a Poisson arrival rate, but an exact load $G = \frac{N}{M}$, e.g. there are $G \times M$ users exactly, as is common in IRSA performance evaluation. We represent throughput versus load in figures, as is done in [1, Fig. 5] for instance.

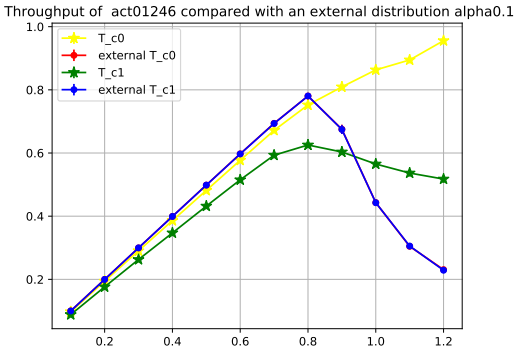
For each selected action set, the throughputs of each of the classes C_0 and C_1 are computed from an average of 300 simulations for a given load G. Note also that for a given load G, one uses the distribution $(\Lambda_i^{RM}(X, C, \bar{G}))_{i \in X}$ obtained in the learning phase by first selecting the $\bar{G} \in \{0.1, 0.2, 0.3 \ldots 1.2\}$ closer to G. The scaled throughputs are represented in Fig. 6.4. The scaling factor is 2, to account for the fact that the actual load of one class is $\frac{1}{2}G$, and to make it comparable to classical IRSA (without classes). Thus, the graph for a "perfect" random access protocol would be a line $y = x$ for $x \in [0, 1]$. The priority parameter α is set to 0.1: this means that the agents of class C_1 would trade 10 lost packets of class C_1 for one successfully decoded packet of class C_0.

We selected various action sets with different features: some with a continuous set of degrees $(0, 1, 2, 3, 4)$, some with high degrees, and some with a mix of both high and low degrees. The reported results of simulations are for actions that are representative of what had been observed in general. As the class C_1 tends to a strategy that weights 10 times more than the throughput of the class C_0, we expect it to choose the actions that limit the collisions with the packets of the class C_0, if possible, until around the throughput of class C_0 is somewhere 5× to 10× higher.

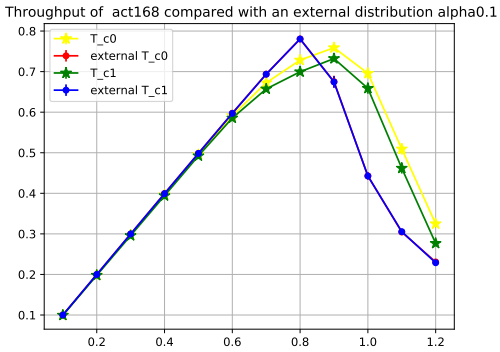
Fig. 6.4 reports the results for 4 different action sets. We are interested in assessing the quality of the priority mechanism introduced by having different distributions for the classes: it can be measured from the gap between the achievable throughput of the two classes. From the results (confirmed by others not presented here), we find that the first defining feature is the inclusion or not of action 0 in the action set: Fig. 6.4a and Fig. 6.4b represent results from two different subsets of actions that include action 0. We can see that the usual sharp decrease of throughput with IRSA around $G = 1$, does not occur for class C_0: only the throughput of class C_1 decreases at higher loads. The priority mechanism is thus working very well, as class C_1 leaves room for class C_0, indeed as its distribution is: $\Lambda^{RM}(\{0, 1, 2, 3, 4\}, 1, 1.2) = (\Lambda_0 = 0.594, \Lambda_1 = 0.088, \Lambda_2 = 0.086, \Lambda_3 = 0.101, \Lambda_4 = 0.130)$, with $\Lambda_0 = 0.594$, around 60% of its transmissions are suppressed at load $G = 1.2$.

In Fig. 6.4c and Fig. 6.4d, we used different actions sets, this time without action 0. We observe that this time, the class C_1 experiences the classical IRSA sharp decrease at a

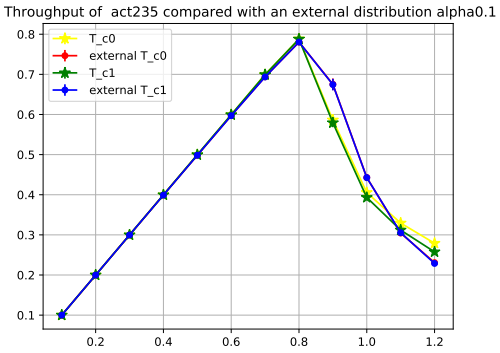
higher load. Introducing action=1 in the set, seems to slightly allow differentiation between classes: for action set $X = \{1, 3, 6\}$, $\Lambda_1(C_0) = 0.412$ and $\Lambda_1(C_1) = 0.590$, therefore a noticeable amount of packet transmission is just one single transmission (e.g. no repetition). Transmissions with such degree=1 colliding on the same slot cannot be retrieved by SIC, hence automatically resulting in lost packets (and lost slots). Therefore, at higher loads, the protocol has to find a balance between this phenomenon (wasting slots), and higher degree repetitions, that can benefit from SIC, but also risk blocking several slots, if undecoded. Action 1 appears safer, from the shown values of Λ_1.



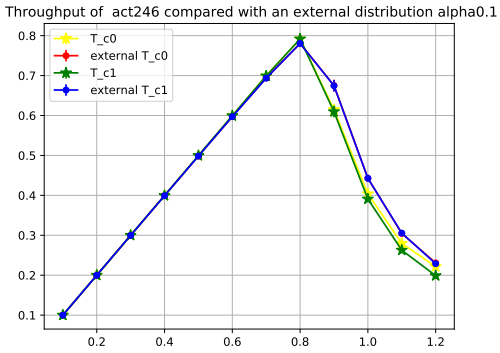
(a) Scaled throughput comparison between actions $\{0, 1, 2, 4, 6\}$ and an external distribution



(b) Scaled throughput comparison between actions $\{1, 6, 8\}$ and an external distribution



(c) Scaled throughput comparison between actions $\{2, 3, 5\}$ and an external distribution

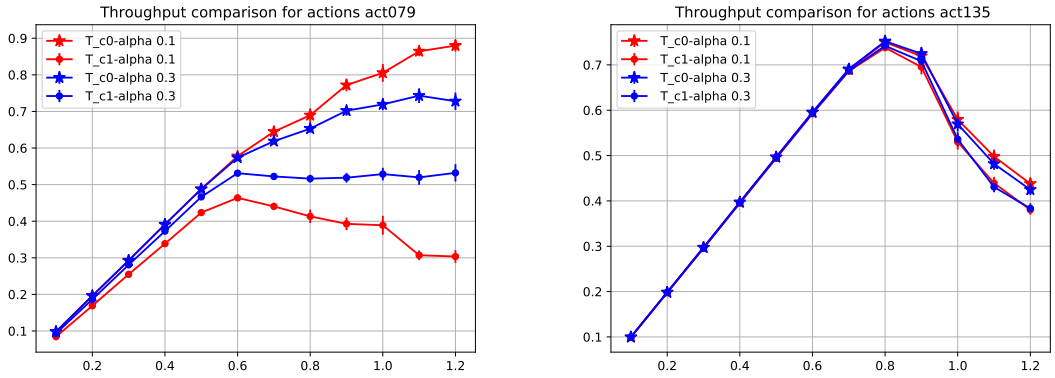


(d) Scaled throughput comparison between actions $\{2, 4, 6\}$ and an external distribution

Figure 6.5: Scaled throughput comparison between a different set of actions and an external distribution ($\Lambda_2 = 0.5, \Lambda_3 = 0.28, \Lambda_8 = 0.22$) from [1]

In [1], a framework for finding degree distributions was proposed for framed IRSA, using Density Evolution (for deterministic performance evaluation), and using Differential

Evolution (as a heuristic for finding a solution of $(\mathcal{P}_1)$): this method aims to find the distribution with the highest load threshold G^*, that is, the load up to which packet loss is vanishingly small when frame size increases towards infinity. These distributions are good comparison points, even though they are optimized for a different context. Fig. 6.5 shows the comparison between the achieved throughput by IRSA-RM with two classes and the achieved throughput by using the IRSA degree distribution $\Lambda_2 = 0.5, \Lambda_3 = 0.28, \Lambda_8 = 0.22$ from [1] (named there "$\Lambda_3(x)$") which we refer to as "external distribution". Fig. 6.5a shows a higher achieved throughput for the class C_0 using the learned set of actions $\{0, 1, 2, 4, 6\}$ with IRSA-RM compared to an external distribution. This is due to the priority mechanism: using the action 0, for a sizeable amount of time, the class C_1 leaves free slots for the class C_0. This same effect appears in Fig. 6.5b, thanks to degree $= 1$. In contrast, the achieved throughput for both classes in Fig. 6.5c and Fig. 6.5d is comparable to the throughput of the external distribution (always close, and for load $g > 0.8$, better as $\frac{1}{4}$ of the points, otherwise worse). Both IRSA-RM and the external distribution achieve the same maximum load of 0.8. This comparison proves that our learning algorithm operates very well in its objective of finding good distributions.

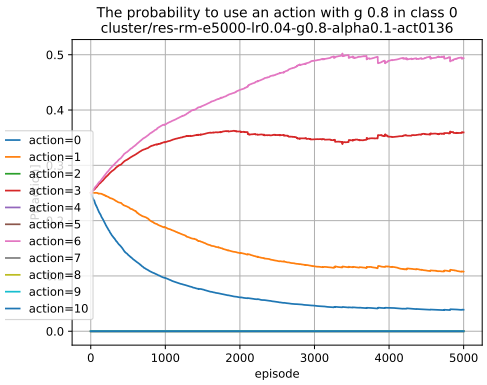


(a) Scaled throughput of actions $\{0, 7, 9\}$ for $\alpha =$ 0.1 and $\alpha = 0.3$

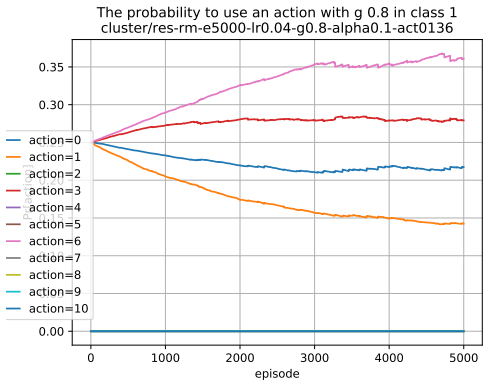
(b) Scaled throughput of actions $\{1, 3, 5\}$ for $\alpha =$ 0.1 and $\alpha = 0.3$

Figure 6.6: Scaled throughput comparison for different set of actions and different priority parameter values

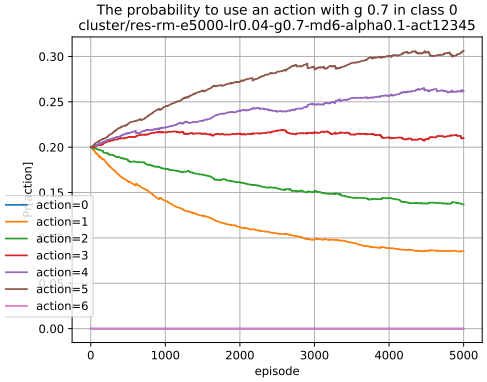
Next, we study the impact of the priority parameter α on the achieved throughput in both classes. In Fig. 6.6a, we compare the achieved throughput of both classes where $\alpha = 0.1$ and $\alpha = 0.3$ when using the action set $\{0, 7, 9\}$. As previously in Fig. 6.4a, with $\alpha = 0.1$, and action 0 in the action set, the priority mechanisms work well. The gap between the achievable throughput of both classes decreases (in blue) when α is increased to 0.3,



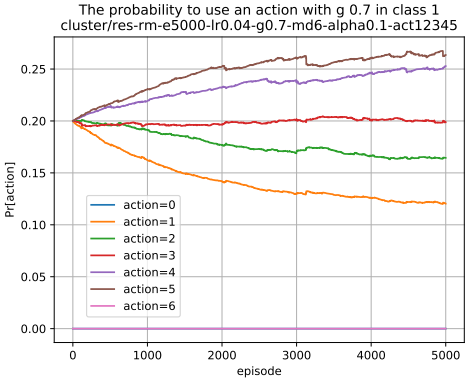
(a) The convergence of the probabilities of taking the actions $\{0, 1, 3, 6\}$ for class C_0



(b) The convergence of the probabilities of taking the actions $\{0, 1, 3, 6\}$ for class C_1



(c) The convergence of the probabilities of taking the actions $\{1, 2, 3, 4, 5\}$ for class C_0



(d) The convergence of the probabilities of taking the actions $\{1, 2, 3, 4, 5\}$ for class C_1

Figure 6.7: The convergence of the probabilities of taking the actions for both classes and for different sets of actions

as expected. Indeed, the coefficient $\Lambda_0(C_1, g = 1.2)$ decreases from 0.759 to 0.591 (hence action 0 is less used). When the action 0 is not available, as in Fig. 6.6b with action set $\{1, 3, 5\}$, the best option for the class C_1 to increase the throughput of the other class is to choose the action 1 (as $\Lambda_1(C_1) = 0.594$ for $G = 1.2, \alpha = 0.1$). As explained previously, the impact is still limited as shown by the small gap, and small difference when $\alpha = 0.3$ (and $\Lambda_1(C_1) = 0.538$ for $G = 1.2, \alpha = 0.3$). Indeed, the class C_1 has no other choice than to send at least one replica, which will always occupy some slot(s).

Finally, Fig. 6.7 reports the convergence of the RM learning process. The learning parameter was set to $\eta = 0.04$. Recall that the learning algorithm updates the weights of the actions $(w_i)_{i \in X}$ for each selected action after the proper update delay, and that these weights are used to compute the probabilities $(p_i)_{i \in X}$ of selecting each action according to Eq. (6.1). Again, these are equivalent to a degree distribution Λ. In Fig. 6.7a, for a network

load $G = 0.8$ and $\alpha = 0.1$, we show the evolution of the probabilities during the learning for the action set $\{0, 1, 3, 6\}$ at the end of each episode. The probabilities of selecting the smaller degrees 0 and 1 are dropping, while the probabilities to use the larger degrees 3 and 6 are rising. The changes stop around the episode, 3300 where the probabilities start to plateau (it is also true for class C_1). Disregarding action 0 (and to some extent, action 1) is the result of class C_0 attempting to maximize its throughput. On the other hand, probabilities of action 0 and 1 have the inverse behavior for class C_1; notice that because G is not so high, action 0 is still not the most selected. In Fig. 6.7c and Fig. 6.7d, we show the convergence of the probabilities for another set of actions without action 0 and for a network load $G = 0.7$ and $\alpha = 0.1$. The probabilities show a form of convergence around episodes $3100 - 3200$ for both classes. Notice that the learning rate could be a function of the episodes as in "1/(episode index)" for instance, but for practical purposes, our fixed learning parameter appears sufficient for our learning phase.

6.6 Conclusion

In this chapter, we studied the random access protocol, Irregular Repetition Slotted Aloha (IRSA) in its frameless version. We adapted a reinforcement learning approach based on Regret Minimization (RM) to optimize the transmission strategy of this protocol, and thus proposed the protocol IRSA-RM. RM is well suited to IRSA, as in both cases, one uses a set of probabilities of selecting a given number of repetitions Λ. The learning is performed offline: it learns the main protocol parameters (the user degree distribution) for a set of predefined network loads. After the learning phase, the parameters can be later used in a network: assuming that the estimate of the load is broadcast by the base station, each device will select the set of parameters that were learned with the closest load. We detailed precisely the mapping between our problem, optimizing IRSA, and the centralized learning approach with RM, including delayed updates, reward computation, alternate simulations, the introduction of priority classes, etc. Simulation results show a high level of performance of IRSA when it is optimized with Regret Minimization, and how IRSA-RM behaves for different types of actions (degrees) sets. Future work will include considering richer actions, maybe more sophisticated RM techniques, and applying Deep Reinforcement Learning techniques (see Chapter 8 and Chapter 9).

Chapter 7

A Game Theoretic Approach for IRSA

7.1 Introduction

In this chapter, we study the properties and the behavior of the IRSA random access when users might select different IRSA protocol parameters. This is modeled through a game theory approach, where users are competing. The main objective is to address the IRSA access scheme according to a competitive distributed approach. Users are grouped in classes, where users of one class share the same degree distribution, and each class has a fixed traffic load. The distributed approach occurs when the classes autonomously and selfishly set their degree probabilities to improve their effective throughput. This is naturally modeled as a non-cooperative game. The objective of this chapter is to study in detail this competition from a game-theoretic perspective. The main questions that we are addressing are the following:

- If users (classes) are freely setting their own IRSA parameters, is the performance of IRSA noticeably impacted?
- Can we prove that users' competition will not result in permanent oscillations?

We answer the questions by studying the existence of a Nash Equilibrium (NE), that we prove under some conditions. We also prove the convergence of the game towards this Nash Equilibrium. In addition, we provide illustrative numerical results, including the observed convergence speed and the price of anarchy. Notably, we show that unrestricted class competition results in no loss of efficiency in many cases, and in very low loss of efficiency in the worst cases (less than 2%).

Through extensive simulations, we assess the efficiency of the distributed approach, optimized through Game Theory. We also show that our IRSA game attains the Nash equilibrium via the "better reply" strategy, and quantify the cost of having users competing,

that is, the price of anarchy in comparison with a centralized approach. Our results imply that user competition does not fundamentally impact the performance of the IRSA protocol when the traffic load is fixed.

7.2 Related Work

As mentioned previously, there exists extensive literature on random access protocols themselves, dating back to over several decades. In this section, we focus on research studies that applied various game theoretic techniques to some random access protocols.

In a multiple access scheme, nodes can either cooperate or compete to achieve their objectives (e.g., optimal throughput, latency,...). Consequently, game theory has become a very useful mathematical tool to model and analyze multiple access schemes in wireless networks, and to obtain solutions for resource allocation, channel assignment, power control, and cooperation enforcement among the nodes. We can find two major game-theoretic approaches to model multiple access schemes: non-cooperative and cooperative game approaches [102]. In a non-cooperative game, the actions of the players are based on their payoff only. In a cooperative game, players establish an enforceable agreement in their group such that the game is between competing groups of players.

Game theory has been applied to random access for designing new random access protocols for future wireless applications. In [103], a general game-theoretic framework for designing contention-based medium access control has been presented. The behavior of selfish users who would want to transmit in every slot to get their packets successfully delivered to the receiver has also been addressed using game-theoretic approaches, as in [104] and [105]. Game theory has been also applied to study the random access scenarios with power control. A variant of ALOHA involving two transmission power levels has been presented in [106]. The authors have presented two non-cooperative optimization concepts: the Nash Equilibrium (NE) and the Evolutionary Stable Strategy. In [107], a multiple access game for ALOHA with (power-domain) NOMA has been formulated. The payoff function is based on an energy efficiency metric. The mixed-strategy Nash equilibrium has been derived for each user's transmission or access probability. It has been shown that the probability of transmissions can approach one as the reward of successful transmission increases. Therefore, the throughput does not approach zero, as in ALOHA, despite the packet collisions because the power levels of users are different thanks to NOMA.

The work in [108] exploits game-theoretic tools to study a non-cooperative IRSA game, and to our knowledge, this pioneering work [108] is also the only existing prior work to

study IRSA from a game theory perspective. The described scenario involves a system of selfish, uncoordinated users, where each user tries to maximize its successful decoding probability. The study aims to define an access cost that allows the degree distribution of users to be a Nash equilibrium. Thus, interestingly, the main idea is to change the cost function so that the existence of a Nash equilibrium is proven, and it would be in the best interest of users to follow it. However, no proof was provided that the best response algorithm can attain the Nash equilibria of the game. In addition, the fact that the operator does modify the users' utility function through pricing to enforce a predefined degree distribution may arguably seem a little artificial. Indeed, in any system, the network operator can enforce any particular user behavior by incurring a large penalty through cost and billing for deviating from the operator-dictated behavior. Then [108] still left open the question of the behavior of the IRSA protocol when users are competing and are left to their own devices, which is the topic that we are addressing.

7.3 System Model and Assumptions

7.3.1 System Description

In this chapter, we consider a system model, similarly to the one in [1, 108], and the one described in Section 2.4.3.

We still assume a *collision channel* model: two or more transmissions on the same slot result in a collision where no packet can be retrieved, whereas a single transmission (singleton) is always perfectly recovered. Fading effects are ignored. We assume that the SIC process is performed perfectly. We also consider that all packets arrive successfully at the receiver. It is possible to adopt more realistic assumptions on fading (see [26] and [109] for instance), on SIC errors due to residual energy after removing the signal from the slot (see Chapter 5), etc., and this would be a subject of future work. We observe that even with the simplifying assumptions, our game-theoretic analysis has merits, otherwise, it would easily become intractable.

7.3.2 Density Evolution

7.3.2.1 Notations for Density Evolution

We adopt the density evolution equations described in Section 2.4.3. Table 7.1 summarizes the basic DE equations used to model IRSA iterative decoding process. These equations

General IRSA Notations

Equation	Reference
$\Lambda(x) \triangleq \sum_\ell \Lambda_\ell x^\ell$	Burst node degree distribution in Eq. (2.1)
$\Psi(x) \triangleq \sum_\ell \Psi_\ell x^\ell$	Slot node degree distribution in Eq. (2.1)
$\lambda(x) \triangleq \sum_\ell \lambda_\ell x^{\ell-1}$	Polynomial representation of the burst edge degree distribution in Eq. (2.3)
$\rho(x) \triangleq \sum_\ell \rho_\ell x^{\ell-1}$	Polynomial representation of the slot edge degree distribution in Eq. (2.3)
$q = p^{\ell-1}$	Probability on the edge towards a slot node in Eq. (2.5)
$(1-p) = (1-q)^{\ell-1}$	Probability on the edge towards a burst node in Eq. (2.6)
$q_i = f_b(p_{i-1})$ and $p_i = f_s(q_i)$	In Eq. (2.9)
when $M \to \infty$: $f_s(q) \to F(\frac{G}{R}q)$ with $F(x) \triangleq 1 - e^{-x}$	Asympotic expression from Eq. (2.11)

Table 7.1: Density Evolution equations used for K-IRSA analysis

are going to be used later to analyze the convergence of the iterative decoding process through the density evolution equations (in particular Eq. (2.9)). Remember that the sequence $(p_i)_{i=0,\dots,\infty}$ characterizes how many of the edges in the graph correspond to undecoded users at each iteration i. Note that the sequence $(q_i)_{i=0,\dots,\infty}$ can equivalently be considered. Initial values are $q_0 = 1$, hence $p_0 = 1 - \rho(0)$. Alternately, picking $p_0 = 1$ just shifts the sequence by one, as it implies that $q_1 = \lambda(1) = 1$.

We will also study some properties of these iterated equations.

7.3.2.2 Convergence and Properties of Iterated Equations

To be able to finely analyze the performance of IRSA, we prove some properties of the density evolution iterations, some of which are known for LDPC-codes [61], but need to be transposed to IRSA:

- The sequence $(p_i)_{i\in\mathbb{N}}$ is decreasing:

Lemma 1 (Decreasing of the sequence $(p_i)_{i\in\mathbb{N}}$ with respect to iterations)
For all the iterations $i = 1, 2, 3, \dots$, $p_i = F(p_{i-1}; G, \Lambda)$ is a monotone, decreasing, sequence.

Proof: By induction: if for some $i \geq 1 : p_i \leq p_{i-1}$, then:

$$\begin{aligned} & p_i \leq p_{i-1} \\ \Rightarrow\ & F(p_i) = p_{i+1} \leq F(p_{i-1}) = p_i \\ & \text{(since F is monotone increasing with } p) \\ \Rightarrow\ & p_{i+1} \leq p_i \leq p_{i-1} \end{aligned}$$

Now consider $p_0 = 1$ and $p_1 = F(p_0; G, \Lambda)$. We have, $p_1 = 1 - e^{-G\Lambda'(p_0)} < 1$ thus $p_1 < p_0$, the induction hypothesis for $i = 1$. This proves the monotonicity and decreasing of the sequence $\{p_i\}_{i\geq 0}$.
Notice that the sequence is also lower bounded by, 0 and hence it must converge.

■

- The sequence $(p_i)_{i\in\mathbb{N}}$ converges towards a limit p_∞: Similarly to the proof in [61, Sect. 3.10] and specifically [61, Lemma 3.55] "Monotonicity with Respect to Iteration", we can now prove in Theorem 1 that the sequence $(p_i)_{i\in\mathbb{N}}$ is decreasing, and it converges towards a limit.

  Theorem 1 (Convergence of the sequence $(p_i)_{i\in\mathbb{N}}$) *The sequence $(p_i)_{i\in\mathbb{N}}$ is decreasing and converges towards a limit p_∞, that is the fixed point of the equation:*

$$p_\infty = F(p_\infty; G, \Lambda) \ \text{ with } F(x; G, \Lambda) = 1 - e^{-G\Lambda'(x)} \tag{7.1}$$

Proof: From Lemma 1, $(p_i)_{i\in\mathbb{N}}$ is a decreasing sequence. In addition, it is lower-bounded by 0, and $F(x; G, \Lambda)$ is continuous in x. Then the Theorem is a direct consequence of fixed-point theorems. ■

- The function F defined in Table 7.1 is a monotone function:

  Lemma 2 (Monotonicity (increasing) of $F(x; G, \Lambda)$ with respect to x)
  Let (Λ, λ) be the node and edge degree distribution pair. The function $F(x; G, \Lambda)$ defined in Eq. (2.11) is monotone, increasing with its argument x.

Proof: Consider some $x^* \in [0, 1]$ and $x \in [0, 1]$ with $x^* > x$:

$$\begin{aligned} & x^* > x \Rightarrow \lambda(x^*) > \lambda(x) \ \text{ (since } \lambda(x) \text{ is a polynomial with only positive coefficients)} \\ \Rightarrow\ & -G\Lambda'(1)\lambda(x^*) < -G\Lambda'(1)\lambda(x) \Rightarrow -e^{-G\Lambda'(1)\lambda(x^*)} > -e^{-G\Lambda'(1)\lambda(x)} \\ \Rightarrow\ & 1 - e^{-G\Lambda'(1)\lambda(x^*)} > 1 - e^{-G\Lambda'(1)\lambda(x)} \Rightarrow 1 - e^{-G\Lambda'(p^*)} > 1 - e^{-G\Lambda'(p)} \\ \Rightarrow\ & F(x^*; G, \Lambda) > F(x; G, \Lambda) \end{aligned}$$

and thus $F(x; G, \Lambda)$ is monotone increasing with x for any $x \in [0, 1]$.

■

As a conclusion, Lemma 2 proves that F is a monotone function, Lemma 1 proves that the sequence is decreasing. An important consequence is given by the Theorem 1 that proves that:

> *The sequence $(p_i)_{i\in\mathbb{N}}$ is decreasing and converges towards a limit p_∞, that is the fixed point of the equation:*
>
> $$p_\infty = F(p_\infty; G, \Lambda) \tag{7.2}$$
>
> *with F defined in Eq.* (2.11).

7.3.2.3 Performance Metrics from Density Evolution

As explained in previous section, the evolution of probabilities and their limit, the fixed point p_∞, gives an indication of the amount of decoded users. It is valid when the frame size M grows infinitely and can be a good approximation for a large finite frame size [1]. The two main performance metrics of density evolution that we use in this work are defined from p_∞ as follows:

(a). p_∞: the fixed point of Eq. (7.2) in Theorem 1. It is a function of Λ and G. It can be mapped to the probability of a packet loss at the end of the decoding process. In more detail, as indicated in [1], p_∞ is the probability that an edge in the bipartite graph of Fig. 2.3 is unrevealed at the end of the decoding process.

(b). The Packet Loss Rate (PLR): since the probability that a packet is not decoded after a certain number of decoding iterations is given by the probability that all its replicas are unrevealed, if the user node has ℓ replicas, then the probability that a packet of this user node is lost at the final iteration is $p_\infty^{\ell-1}$. By averaging on all the possible degrees, we have:

$$PLR(p_\infty) = \Lambda(p_\infty) \tag{7.3}$$

See [1, footnote 13].

(c). The throughput: the expression of the effective throughput (i.e the goodput) is the load multiplied by the PLR:

$$T(\Lambda) \triangleq G \cdot (1 - \Lambda(p_\infty)) \tag{7.4}$$

It measures the average number of decoded packets per slot. With a collision model, it verifies $T \leq 1$, and without losses $T = G$.

Notice that because we assume that the load G is constant (e.g. consider IoT applications where nodes are sending packets with fixed average rate), the throughput is essentially a proxy for the PLR.

7.3.2.4 Performance Metrics with Several Classes

We extend the results of the previous section to a system that has two different classes C_0 and C_1 and a fixed global load G. Each user is assumed to belong to exactly one class, and all users of one class c are assumed to use the same degree distribution $(\Lambda_{c,i})_{i\geq 2}$. Λ_{C_0} and Λ_{C_1} are the degree distributions of the classes, C_0 and C_1, respectively. We introduce $\alpha \in [0, 1]$, a parameter that indicates the proportion of the users that are in class C_0 ($\alpha = \frac{1}{2}$ implies that users are split equally among both classes).

We define $\Lambda_{avg}(x)$, the average degree distribution of both classes, as:

$$\Lambda_{avg}(x) = \alpha\Lambda_{c_0}(x) + (1-\alpha)\Lambda_{c_1}(x) \tag{7.5}$$

The system behaves as if there was only one class of users, with load G, and degree distribution Λ_{avg}.

Then p_∞ is obtained as the fixed point of $p_\infty = F(p_\infty; G, \Lambda_{avg})$. The throughput of the two classes is different and is obtained respectively as:

$$\begin{aligned} T_{C_0}(\Lambda_{C_0}, \Lambda_{avg}) &= \Lambda_{C_0}(p_\infty) \\ T_{C_1}(\Lambda_{C_1}, \Lambda_{avg}) &= \Lambda_{C_1}(p_\infty) \end{aligned} \tag{7.6}$$

These results can be generalized to more than one class.

7.3.3 Optimal Formulation for IRSA

Before addressing the non-cooperative version with competing users, we first recall that the optimization problem for a centralized IRSA has been explained in ($\mathcal{P}_1$) (Section 2.4.4). These preliminaries provide a performance baseline for the non-cooperative setting, and also illustrate the optimization of IRSA in general.

7.3.3.1 Optimal Formulation for a One-Class IRSA Scenario

In the classical IRSA system of one class of users, the aim is to find the best degree distribution that maximizes the system throughput at a given network load G. In classical IRSA, the degree distribution also starts from degree 2, i.e. $\Lambda_0 = \Lambda_1 = 0$. We describe the optimization problem as follows:

$$\begin{aligned} \underset{(\Lambda_i)}{\text{maximize}} \quad & T(\Lambda) \qquad (\mathcal{P}_5) \\ \text{subject to} \quad & 0 \leq \Lambda_i \leq 1 \ \forall i \in \{2, 3, \ldots, D\} \\ & \sum_{i=2}^{i=D} \Lambda_i = 1 \end{aligned}$$

where Λ is the degree distribution shared between all users, and T is the system throughput, which measures the normalized number of decoded packets per slot. When using density evolution, and assuming asymptotically large frame size, T is given by Eq. (7.4), which implies computing $p_\infty(\Lambda, G)$ from Eq. (7.2).

7.3.3.2 Optimal Formulation for a Two-Class IRSA Scenario

We extend the optimal formulation of the previous section, to a system that has two different classes C_0 and C_1 and a fixed global load G. Using the performance metrics for multiple classes from Section 7.3.2.4, we can rewrite the optimization problem $(\mathcal{P}_5)$ for a class selfishly optimizing its own throughput as

$$\begin{aligned} \underset{(\Lambda_{c,i})}{\text{maximize}} \quad & T_c(\Lambda_c, \Lambda_{avg}) \qquad (\mathcal{P}_6) \\ \text{subject to} \quad & 0 \leq \Lambda_{c,i} \leq 1 \ \forall i \in \{2, \ldots, D\} \\ & \sum_{i=2}^{i=D} \Lambda_{c,i} = 1 \end{aligned}$$

The described problem in $(\mathcal{P}_6)$ can be solved by using the expression of the throughput in Eq. (7.6).

7.4 The Class-based IRSA Non-Cooperative Game

In this section, we study the autonomous behavior of competing classes of users that engage in a non-cooperative strategic IRSA game. Non-cooperative game theory models the interactions between players competing for common resources. Here, the classes of users are the decision-makers or players of the game that seek selfishly to maximize their own throughput. We define a multi-player game $\mathcal{G}$ between n classes of users. Each class is assumed to make its decisions without knowing the decisions of other classes.

7.4.1 Formalization of the Non-Cooperative IRSA Game

The IRSA non-cooperative game $\mathcal{G} = \langle K, S, T \rangle$ can be described as follows:

- A finite set of classes $K = (C_0, ..., C_{n-1})$.
- For each class $c \in K$, the space of pure strategies S_c is formed by the Cartesian product of each set of pure strategies $S_c = S_{c,2} \times ... \times S_{c,L}$, where L is the maximal degree.
- An action of each player (i.e. class) consists in selecting one distribution Λ_c for a set of frames, where $\Lambda_c \in S_c$ represents a user degree distribution $\Lambda_c = \{\Lambda_{c,2}, ...\Lambda_{c,L}\}$. As $\Lambda_{c,k}$ is the probability of repeating a packet k times for a user in class c, then $\Lambda_{c,k} \in [0, 1]$, for $k \in \{2, .., L\}$.

  Furthermore, as previously, the general degree distribution (strategy) of one class c can be written in polynomial form as follows:

$$\Lambda_c(x) = \sum_{\ell=2}^{L} \Lambda_{c,\ell} x^{\ell} = \Lambda_{c,2} x^2 + \Lambda_{c,3} x^3 + .. + \Lambda_{c,L} x^L \tag{7.7}$$

- A *strategy profile* $\Lambda = (\Lambda_0, ..., \Lambda_{n-1})$ specifies the strategies of all players and $S = S_0 \times ... \times S_{n-1}$ is the set of all strategies.
- A set of utility functions, $T = (T_0(\Lambda), T_1(\Lambda), ..., T_{n-1}(\Lambda))$ where each utility function quantifies players' utility for a given strategy profile Λ, where $T_c : S_c \to \mathbb{R}$ measures the preference of the strategy $\Lambda_c \in S_c$ played by a player c.

  Each player (class) intends to select the strategy (the degree distribution) that maximizes their utility function. The utility function of a player $c \in K$ that plays strategy $\Lambda_c \in S_c$ is the *Throughput* of this player, defined as:

$$T_c(\Lambda_c) = G \cdot (1 - PLR_c(\Lambda_c)) \tag{7.8}$$

  where PLR_c is the packet loss rate of class c users, and G is the total network load. When G is constant, maximizing the throughput is equivalent to minimizing the Packet Loss Rate (PLR), as in Eq. (7.6). This corresponds to an optimization problem as in Section 7.3.3.2 written with an arbitrary number of classes.

7.4.2 Two-Classes IRSA game

We consider two classes of users. The two major components of the devised two-players IRSA game are as follows:

- A finite set of players, denoted by K, $c \in K$ is a class of users. For clarity, we coin the two classes as class C_0 and class C_1 respectively, i.e. $K = \{C_0, C_1\}$.
- Accordingly, the strategy space of class C_0 (resp. class C_1) is denoted by S_{C_0} (resp. S_{C_1}).

The total network load is defined as $G = \frac{M_{C_0}+M_{C_1}}{N}$, where N is the number of slots in the frame and M_{C_0} (resp. M_{C_1}) is the number of users in class C_0 (resp. the number of users in class C_1). We also define $\alpha = \frac{M_{C_0}}{M}$ as the proportion of class C_0 users over the total number of users M (hence $(1-\alpha)$ is the proportion of C_1 users) as in Section 7.3.2.4.

Definition 1 (Restricted IRSA game) *We define a restricted IRSA game where only two coefficients in the degree distribution are non-zero for all the strategy sets of the two players: we denote respectively by $\ell > m > 1$ with $m \neq \ell$, the two non-zero degrees:*

$$\Lambda_{C_0}(x) = \Lambda_{C_0,m} \cdot x^m + \Lambda_{C_0,\ell} \cdot x^\ell = \Lambda_{C_0,m} \cdot (x^m - x^\ell) + x^\ell \tag{7.9}$$

where the last equality comes from the fact that $\Lambda_{C_0,m} + \Lambda_{C_0,\ell} = 1$. We note for convenience $s_c = \Lambda_{c,m}$, accordingly:

$$\Lambda_c(x) = s_c x^m + (1 - s_c)x^\ell \tag{7.10}$$

$$\text{and } T_c(s_c) = G \cdot (1 - PLR_c(s_c)) \tag{7.11}$$

where ℓ and m are the non-zero constant integers representing the repetition degrees of the game with $\ell > m$.

($\mathcal{P}_1$) in Section 2.4.4 explains the main purpose behind IRSA optimization. Any enhancement in this protocol could be translated into an optimization problem that maximizes a given system metric (load, achieved throughput, ...) or minimizes the impact of a given drawback (delay, PLR, ...). In the IRSA game, instead of maximizing the total throughput as in ($\mathcal{P}_5$), each player aims to maximize their own throughput function in Eq. (7.11), for a given network load.

7.4.3 Nash Equilibrium of the IRSA Game

In this section, we establish the needed proof to show that the user degree distribution $\Lambda(x)$ could be a Nash equilibrium (NE) for the 2-classes strategic IRSA game under some conditions. We first use the classic definition of the Nash equilibrium as the point where no player will gain by changing their strategy unilaterally. In our case, it corresponds to:

Definition 2 (Nash equilibrium for IRSA strategic game) *A Nash equilibrium (NE) for the 2-classes IRSA strategic game is a strategy (degree distribution), $\Lambda^* = (\Lambda_c^*, \Lambda_{-c}^*)$ such that:*

$$T_c(\Lambda_c^*, \Lambda_{-c}^*) \geq T_c(\tilde{\Lambda_c}, \Lambda_{-c}^*) \qquad \forall \tilde{\Lambda_c} \in S_c \tag{7.12}$$

Where Λ_c^* is the strategy of class $c \in \{C_0, C_1\}$ and Λ_{-c}^* is the strategy of the other class(es).

NE describes the operating point which is stable in terms of local efficiency for all players. Therefore, in our IRSA game, we seek to study the existence of pure NEs and how to attain them.

Consider the 2-classes IRSA game, defined as a static strategic non-cooperative game with complete information and a finite set of players (2 players). There exist several ways to establish the existence of a Nash equilibrium, one of which is the Debreu-Fan-Glicksberg theorem (see for instance the overview in [110]): with our notations, (a) if $\forall c \in \{C_0, C_1\}$, S_c is a compact and convex set, (b) the utility function $T_c(\Lambda_c)$ is a continuous function in the profile of strategies S and (c) it is quasi-concave in the set of a player c own strategies S_c, then the game has at least one pure NE, according to this Debreu-Fan-Glicksberg theorem.

We actually prove this supposition in the following Theorem 2:

Theorem 2 (Existence of NE for the 2-classes restricted IRSA game for large G) *The two-players restricted IRSA game admits a Nash equilibrium when $G \to \infty$. In other words, there exists a load limit $G_1 > 0$, such that for any load $G \geq G_1$, the game always admits at least one pure Nash equilibrium.*

Proof:

For our proof of the NE existence, we split the proof into three Lemmas, described later, that establish the previous conditions (a), (b), and (c). Lemma 3 is provided to prove that the set of strategies of our 2-classes IRSA game is compact and convex. Lemma 4 proves that the utility function defined in Eq. (7.11) is continuous in the profile of strategies S under some conditions. And finally, Lemma 5 shows that the utility function of any player c is quasi-concave in their own strategy, with $c \in \{C_0, C_1\}$, for all G greater than a fixed limit G_0. As a consequence, the conditions of the Debreu-Glicksberg-Fan (1952) theorem are satisfied, which proves the existence of a Nash equilibrium. ■

7.4.4 Analysis of the Utility Function

In this section, we present the three necessary lemmas to prove Theorem 2.

Lemma 3 *Let S_c be the set of strategies of the player $c \in \{C_0, C_1\}$, where each strategy is a degree distribution $\Lambda_c(x)$ written as:*

$$S_c \triangleq \left\{ \Lambda_c(x) \;\middle|\; \Lambda_c(x) = s_c x^m + (1 - s_c) x^\ell, \forall s_c \in [0, 1] \right\}$$

Then the set S_c is compact and convex.

Proof: All properties are obtained from the fact that S_c is a segment of the plane $\mathbb{R}^2$ (and for non-restricted IRSA games, it would be the intersection of a hypercube and a hyperplane

in $\mathbb{R}^m$). It is easy to show that ***the set is compact:***

1. The set S_c is closed and

2. The set S_c is bounded and

3. The set is convex:

(a) $\forall s_i, s_j \in S_c$*:*

$0 \leq s_i \leq 1, \Rightarrow 0 \leq \theta s_i \leq \theta$

$0 \leq s_j \leq 1, \Rightarrow 0 \leq (1-\theta)\, s_j \leq (1-\theta)$

$\Rightarrow 0 \leq \theta s_i + (1-\theta)\, s_j \leq 1$*, each component of the new vector will be in* $[0, 1]$.

(b) $\forall s_i, s_j \in S_c$*:*

$s_i + (1 - s_i) = 1 \Rightarrow \theta s_i + \theta(1 - s_i) = \theta$

$s_j + (1 - s_j) = 1 \Rightarrow (1-\theta)s_j + (1-\theta)(1 - s_j) = 1 - \theta$

By combining both:

$\theta s_i + \theta(1 - s_i) + (1-\theta)s_j + (1-\theta)(1 - s_j) = 1$*, the sum of all components of the new vector is equal to one.*

From (a) and (b), we conclude that the set S_c is convex.

■

The second and third conditions to have a pure NE are relative to the utility function described in Eq. (7.11). Alongside Lemma 3, we need to show that the utility function for any class $c \in \{C_0, C_1\}$ is continuous in any strategy (proven in Corollary 1) and quasi-concave in its own strategy $s_c \in S_c$ (proven in Lemma 5).

As the player utility is a function of their throughput, we use the results and definitions from density evolution from Sections 7.3.2.3 and 7.3.2.4, to examine some properties of the utility function. The value of the fixed point p_∞ depends on the strategies of both players $s_0 = \Lambda_{C_0}, s_1 = \Lambda_{C_1}$ and also on the load G, and hence can be written as $p_\infty(s_0, s_1, G)$.

In the following Lemma, we prove that, except for some set of points $\mathcal{S}(s_0, s_1, G)$, p_∞ is continuous in the strategy of one player s_0 (actually the stronger result is that p_∞ is continuously differentiable). This lemma is a first step to prove that the utility function is also continuous in s_0.

Lemma 4 (Continuity and differentiability of $p_\infty(s_0)$) *Considering that s_1 and G are fixed, $p_\infty(s_0)$ (which becomes then a function of one variable s_0), is a continuously differentiable function except on a set of points coined $\mathcal{S}(s_0, s_1, G)$. This set of points $\mathcal{S}(s_0, s_1, G) \subset [0, 1]$ corresponds to the points where some expression $g(s_0, p_\infty(s_0))$ equates to zero.*

Proof: We consider here that s_1 and G are kept constant, and p_∞ (also denoted p in this proof for simplicity) is then a function of a single variable (the strategy of one player) s_0,

as well as all the quantities considered in this proof. We make use of the implicit function theorem, [111, Th. 1.3.1] on $p_\infty(s_0)$ which is defined by the fixed point equation Eq. (7.2).

The main expression of the implicit function theorem is intuitively obtained by considering p_∞ as a solution of the implicit equation $H(p) = 0$. This is equivalent to the fixed point equation Eq. (7.2) when introducing the notation $H(p) = F(p; G, \Lambda) - p$, and then computing the total derivative of (i.e. chain rule for) the equation $H(p) = 0$. Remember that here the only variable is s_0, and p depends on s_0. Differentiating both sides of $H(p) = 0$ with respect to s_0 gives the well-known expression, e.g. [111, Eq. (1.13)]:

$$\frac{dH}{ds_0} = \frac{\partial H}{\partial s_0}\frac{\partial s_0}{\partial s_0} + \frac{\partial H}{\partial p}\frac{\partial p}{\partial s_0} \tag{7.13}$$

Thus:

$$\frac{dH}{ds_0} = 0 \Rightarrow \frac{\partial p}{\partial s_0} = -\frac{\frac{\partial H}{\partial s_0}}{\frac{\partial H}{\partial p}} \tag{7.14}$$

The derivative of p exists only for s_0 for which the denominator is not zero, so that the solution $p(s_0)$ is continuously differentiable in a neighborhood of s_0.
Now, using the definition $H(p) \triangleq 1 - e^{-G\Lambda'(p)} - p$, we compute:

$$\frac{\partial H}{\partial s_0} = G\alpha(mp^{m-1} - \ell p^{\ell-1})e^{-G\Lambda'(p)}$$

considering that:

$$\Lambda(p) = \alpha(s_0 p^m + (1 - s_0)p^\ell) + (1 - \alpha)(s_1 p^m + (1 - s_1)p^\ell)$$

And since $H(p) = 0$, hence $1 - p = e^{-G\Lambda'(p)}$, we obtain:

$$\frac{\partial H}{\partial s_0} = G\alpha(1 - p)(mp^{m-1} - \ell p^{\ell-1}) \tag{7.15}$$

Taking the derivative of the function $H(p)$ with respect to p:

$$\begin{aligned}\frac{\partial H}{\partial p} = G(1 - p) \cdot \alpha\left[(m(m - 1)s_0 p^{m-2} + \ell(\ell - 1)(1 - s_0)p^{\ell-2})\right] \\ +G(1 - p) \cdot (1 - \alpha)\left[(m(m - 1)s_1 p^{m-2} + \ell(\ell - 1)(1 - s_1)p^{\ell-2})\right] - 1\end{aligned} \tag{7.16}$$

Solving equation (7.16) = 0 helps to find the roots where the derivative of $H(\Lambda, p)$ is zero, and for these roots, the derivative of p defined in Eq. (7.14) does not exist. In other words, this gives us the set:
$\mathcal{S}(s_0, s_1, G) = \{s_0 \in [0, 1] \mid Eq.\ (7.16)\text{=}0 \text{ is verified}\}$.

In addition, let us consider an interval $[a, b]$, where $\mathcal{S}(s_0, s_1, G) \cap [a, b] = \emptyset$. For any $x \in [a, b]$, then p_∞ is continuously differentiable in an open set around x [111, Th. 1.3.1]. Thus, p_∞ must be continuously differentiable in the whole interval $[a, b]$. ■

Corollary 1 (Continuity and differentiability of the utility function) *Considering that s_1 and G are fixed, the utility function defined in Eq.* (7.8) *could be written as $T_{C_0}(s_0) = G(1 - (s_0 p_\infty^m + (1 - s_0)p_\infty^\ell))$. The utility function (as a function of one variable s_0) is a continuously differentiable function except on some set of points $\mathcal{S}(s_0, s_1, G)$, as it is a composition of continuous functions.*

Lemma 5 (Monotonicity of the utility function) *Considering that s_1 and G are fixed, Let $T_{C_0}(s_0) = G(1 - \Lambda_{C_0}(s_0)) = G(1 - (s_0 p_\infty^m + (1 - s_0)p_\infty^\ell))$, be the utility function of class C_0, playing a strategy $s_0 \in S_{C_0}$ where p_∞ is a fixed point given by the solution of the Eq.* (7.2). *When $G \to \infty$, T_{C_0} becomes a monotone function (increasing) with s_0.*

Proof: Differentiating $T_{C_0}(s_0)$ with respect to s_0 yields:

$$\frac{dT_{C_0}}{ds_0} = -G\left[p_\infty^m - p_\infty^\ell + \frac{\partial p_\infty}{\partial s_0}\left(s_0 m p_\infty^{m-1} + (1 - s_0)\ell p_\infty^{\ell-1}\right)\right]$$

The expression $\frac{\partial p_\infty}{\partial s_0}$ appearing here can be obtained from Lemma 4, using Eq. (7.15), Eq. (7.16) and for any $s_0 \notin \mathcal{S}(s_0, s_1, G)$, it is:

$$\frac{\partial p_\infty}{\partial s_0} = -G\alpha(1 - p_\infty)(m p_\infty^{m-1} - \ell p_\infty^{\ell-1}) \Bigg/ \Big\{G(1 - p_\infty) \cdot \alpha\left[(m(m-1)s_0 p_\infty^{m-2} + \ell(\ell-1)(1 - s_0)p_\infty^{\ell-2})\right] + G(1 - p_\infty) \cdot (1 - \alpha)\left[(m(m-1)s_1 p_\infty^{m-2} + \ell(\ell-1)(1 - s_1)p_\infty^{\ell-2})\right] - 1\Big\}$$

When $G \to \infty$, we will prove that $p_\infty \to 1$; for convenience, we do a change of variable of p_∞ to ε, defined as $p_\infty \triangleq 1 - \varepsilon$. This allows us to study such limits. However, one technicality is that we need to prove bounds and limits independently of the values of s_0 and s_1. We introduce the following variants of the Bachmann-Landau notation (the family of Big O Notations), to be able to express this, as follows:

- Assuming $G \to \infty$:
- $f(x; s_0, s_1) = \tilde{o}(g(x; s_0, s_1))$ [f is dominated by g asympotically]
  $\implies \forall \eta > 0 \exists N$ such that $\forall x \geq N \forall (s_0, s_1) \in [0, 1]^2$:
  $|f(x; s_0, s_1)| \leq \eta |g(x; s_0, s_1)|$

- $(x; s_0, s_1) = \widetilde{\Theta}(g(x; s_0, s_1))$ [f is bounded both above and below by g asympotically]
  $\implies \exists k_1 > 0 \exists k_2 > 0 \exists N : \forall x \geq N, \forall (s_0, s_1) \in [0,1]^2 :$
  $k_1 g(x; s_0, s_1) \leq f(x; s_0, s_1) \leq k_2 g(x; s_0, s_1)$

Notice the only difference with classical Bachmann-Landau notation definitions $o(x)$ and $\Theta(x)$, are the "$\forall(s_0, s_1)$".

Our first preliminary proof is on the limit of $G\varepsilon$. For fixed $s_0 \in [0,1]$ and $s_1 \in [0,1]$, we have:

$$\lim_{G\to\infty} \varepsilon \overset{(a)}{=} \lim_{G\to\infty} G(1 - p_\infty) \overset{(b)}{=} \lim_{G\to\infty} G(1 - (1 - e^{-G\Lambda'(p_\infty)})) \overset{(c)}{=} \lim_{G\to\infty} Ge^{-G\Lambda'(p_\infty)} \overset{(d)}{=} 0$$

where (a) is by definition of ε, (b) is because p_∞ is the fixed point of Eq. (7.2), (c) is immediate, (d) is because $\Lambda'(x)$ is bounded by $\Lambda'(x) \leq \max(\ell, m) = \ell$ for any $x \in [0,1]$ and because $\lim_{G\to\infty} Ge^{-G\cdot K} = 0$ for any constant $K > 0$.
The key part in the previous reasoning (d), is the use of the bound: $\Lambda'(x) \leq \Lambda'(1) \leq \ell$, valid for any $x \in [0,1]$. This bound does not depend on s_0 and s_1, hence the same reasoning can be applied, to $\max_{s_0,s_1} p_\infty$, and thus this proves the following slightly more general result, independent of s_0 and s_1:

$$\lim_{G\to\infty} \left(\max_{(s_0,s_1)\in[0,1]^2} G\varepsilon \right) = 0 \tag{7.17}$$

$$\text{or with our notation: when } G \to \infty, G\varepsilon = \tilde{o}(1) \tag{7.18}$$

Notice that this is a stronger result that implies the following: when $G \to \infty$, $\epsilon = \tilde{o}(1)$ or in other terms, $p_\infty = 1 + \tilde{o}(1)$.

Armed with these definitions, and preliminary results, we can write $\frac{\partial T_{C_0}}{\partial s_0}$ with a Taylor expansion of $(1-\varepsilon)^\ell$ and $(1-\varepsilon)^m$:

$$\begin{aligned}\frac{\partial T_{C_0}}{\partial s_0} = G \Big\{ & -(1 - m\varepsilon) + (1 - \ell\varepsilon) + \tilde{o}(\varepsilon) \\ & + G\alpha\varepsilon(m - \ell) \cdot [s_0 m(1 - m\varepsilon + \varepsilon) + (1 - s_0)\ell(1 - \ell\varepsilon + \varepsilon) + \tilde{o}(\varepsilon)] \\ & \Big/ \Big(G\alpha \left[\left(m^2 - m\right) s_0 \varepsilon + \left(\ell^2 - \ell\right)(1 - s_0)\,\varepsilon + \tilde{o}(\varepsilon) \right] \\ & + G\,(1 - \alpha) \left[\left(m^2 - m\right) s_1 \varepsilon + \left(\ell^2 - \ell\right)(1 - s_1)\,\varepsilon + \tilde{o}(\varepsilon) \right] - 1 \Big) \Big\}\end{aligned}$$

Hence:

$$\frac{\partial T_{C_0}}{\partial s_0} = G\varepsilon(m - \ell + \tilde{o}(1)) \left[1 + \frac{G\alpha\,(s_0 m + (1 - s_0)\ell + \tilde{o}(1))}{G\varepsilon(D + \tilde{o}(1)) - 1} \right] + \tilde{o}(1) \tag{7.19}$$

$$\text{with } D = \alpha\left[\left(m^2 - m\right)s_0 + \left(\ell^2 - \ell\right)(1 - s_0)\right] + (1-\alpha)\left[\left(m^2 - m\right)s_1 + \left(\ell^2 - \ell\right)(1 - s_1)\right] \quad (7.20)$$

Now considering again the different parts of the expression Eq. (7.19) when $G \to \infty$:

- $G\varepsilon(D + \tilde{o}(1)) - 1 = -1 + \tilde{o}(1)$ because $G\varepsilon = \tilde{o}(1)$
- Therefore $\frac{G\alpha(s_0 m + (1-s_0)\ell)}{G\varepsilon D - 1} = -\widetilde{\Theta}(G)$
- Hence: $1 + \frac{G\alpha(s_0 m + (1-s_0)\ell)}{G\varepsilon D - 1} = -\widetilde{\Theta}(G)$
- Thus finally: $\frac{\partial T_{C_0}}{\partial s_0} = -G\varepsilon(m - \ell + \tilde{o}(1))\widetilde{\Theta}(G)$

We have the sign of every quantity involved in the product in the last expression, when $G \to \infty$: it is ultimately positive. Therefore, we conclude that:

$$\exists G_0 > 0 : \forall G > G_0, \forall s_0 \in [0,1], \forall s_1 \in [0,1], \frac{\partial T_{C_0}}{\partial s_0} > 0 \quad (7.21)$$

And this proves that there exists a $G_0 > 0$, such that the utility function is a monotone (increasing) function with s_0 for any $G \geq G_0$. ■

The three Lemmas of this section prove the Theorem 3.

7.4.5 Attaining Pure Nash Equilibrium

The Theorem 3 established the proof of the existence of (at least) one pure Nash equilibrium. In this section, we are interested in the dynamics of the game, and the convergence to a Nash equilibrium. We start by introducing the dynamics of the game by indicating how players adapt their strategies.

The *better reply* strategy of a player c is the one that improves their utility given other players' strategies. A better reply dynamics scheme consists of a sequence of rounds, where each class c chooses a better reply to the strategies of other classes in the previous round, but not necessarily the best one.[1] In the first round, the choice of each player is a better strategy based on their arbitrary belief about what the other players will choose. In some games, the sequence of strategies generated by better reply dynamics converges to a NE, regardless of the players' initial strategies. It is the case for our game $\mathcal{G} = \langle K = \{C_0, C_1\}, S, T\rangle$ as, according to [112], two-player generic quasi-concave games have the "Weak Finite Improvement Property" (WFIP).

[1] The strategy of choosing the best one, corresponds instead to the classical "*best response*" dynamics, for which the same hypotheses detailed later are not sufficient to establish convergence.

Property 1 (Convergence of Better Reply under WFIP [112]) *Consider a two-player game $\mathcal{G} = \langle K = \{C_0, C_1\}, S, T\rangle$, each player $c \in K$ has a one dimensional, compact, convex strategy set S_c, and a utility function T_c that is twice continuously differentiable in s_c, $\forall c \in \{C_0, C_1\}$ and quasi-concave with respect to $s_c \in S_c$.*
Then the game has the weak FIP, and this implies that from any action profile there is a finite sequence of single-player improvements leading to a Nash equilibrium [112].

We apply this property to our problem, which builds upon Theorem 2, by providing proof for convergence towards NE:

Theorem 3 (Convergence to Nash equilibrium for 2-classes restricted IRSA game for large G) *There exists a load limit $G_1 > 0$, such that for any load $G > G_1$, the two-classes restricted IRSA game will converge to a Nash equilibrium when iterating better reply dynamics.*

Proof: Under the assumption that $G \to \infty$, the existence of the NE is established by Theorem 2 for $G > G_1$ for some $G_1 > 0$. Also, according to Lemma 3, Lemma 5, and Corollary 1, our game is a generic two players quasi-concave game. And for $G > G_1$, T_c is actually not only continuously differentiable but, according to the analytic implicit function theorem [111, Section 6.1], infinitely differentiable. Hence, the conditions of Property 1 are satisfied, and therefore, following [112], better reply dynamics are guaranteed to converge to pure NEs for $G > G_1$. ■

7.4.6 Generalizations of the Two-Classes Restricted IRSA game

The existence of a Nash equilibrium and the convergence to the Nash equilibrium were established for the two-classes restricted IRSA games. One important question would be: how to generalize the results?

For the case of the n-classes restricted IRSA game (with $n \geq 3$): the performance of the system is obtained as previously, by computing a global average distribution Λ_{avg} (generalizing the Eq. (7.5)), and the global p_∞. When one class optimizes its utility, the system actually behaves as if all the other $n-1$ classes are equivalent to a single competing class, and the Theorem 2 can be generalized. On the other hand, the convergence to a Nash equilibrium is no longer established, as the result on the WFIP that we are using is specifically for a two-player game [112].

The second case is the general IRSA game as opposed to the restricted IRSA game of definition 1: here, players have multivariate utility functions, and results on the quasi-concavity are more difficult to establish. The issue is that the utility is computed from a

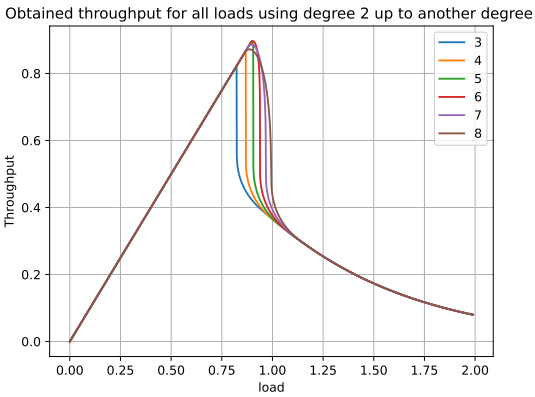
multivariate implicit function p_∞ (which is also not always guaranteed to be continuous). Hence, it is an open problem, but our numerical results at the end of the Section 7.5 show that the general IRSA game is behaving similarly to the restricted IRSA game.

7.5 Numerical Results

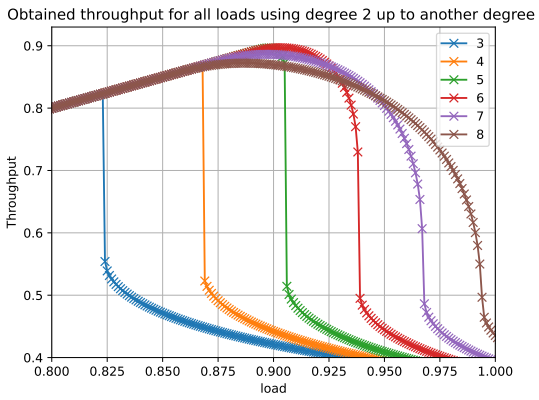
In the following, we assess the performance of the distributed IRSA game, described in Section 7.4, and present numerical verification that it attains Nash equilibrium via the better reply strategy. We assume a collision channel with fixed-size frames (sufficiently large so that the density evolution results are close to the actual performance). The users select the slots uniformly at random, however, they first select the number of packet repetitions according to their degree distributions. The performance is evaluated through density evolution. The network load is varied between $G = 0$ and $G = 2$.

In Fig. 7.1, we show the optimal throughput of a single class of users using the IRSA protocol with two coefficients for two degrees (m, ℓ), as in restricted IRSA (definition 1). In Fig. 7.1a, we consider one single class which always uses a degree $m = 2$ and another degree ℓ chosen from $\{3, 4, \dots 8\}$. We formulate the centralized optimization problem $(\mathcal{P}_5)$ and numerically compute the optimal throughput in Python with the `scipy` *differential evolution* algorithm. We compare the obtained optimal throughput for each pair of degrees. For each pair, there is a maximum value for the obtained throughput before it starts to drop rapidly, and then, it continues to decrease slowly with the increase of load. In fact, we focus on the part where this sudden decrease occurs. In Fig. 7.1b, we zoom in on this phenomenon, where we can see clearly the impact of the discontinuity of the utility function for some values of the load (the sudden decrease: at least a discontinuity for its derivative). We proved in Corollary 1 that the utility function (the throughput) is continuously differentiable except for some points. Thus, the highlighted decrease in Fig. 7.1b is certainly indirectly due to the discontinuity of the throughput in one of those points, itself due to a discontinuity of p_∞ in the same points.

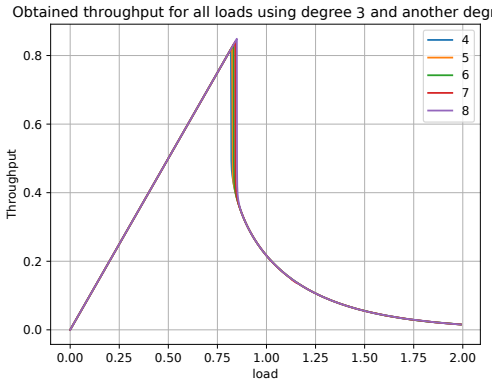
The same phenomenon happens in Fig. 7.1c, if the unique class uses degree $m = 3$ and another degree ℓ from $\{4, 5, \dots, 8\}$. We observe that the discontinuity occurs earlier, and the throughput at high loads is lower. The maximum achieved loads when using a degree 2 and another degree are higher than the maximum achieved loads when using a degree 3 and another degree. Therefore, it is generally better to pick degree 2 and another degree (higher than 3). For the rest of the results, we will focus on the distributions with $m = 2$. In Fig. 7.1d, we show clearly the discontinuity region of Fig. 7.1c where the throughput drops



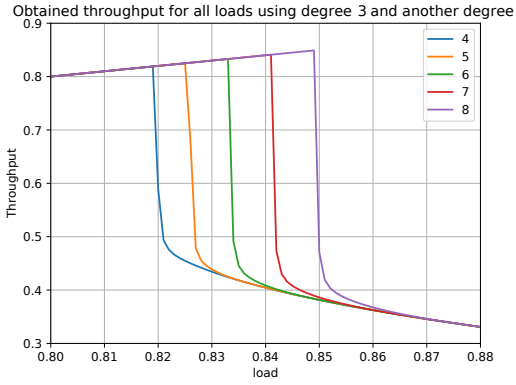
(a) The optimized throughput for one class using degree 2 and another degree for all loads



(b) A close sight of Fig. 7.1a



(c) The optimized throughput for one class using degree 3 and another degree for all loads

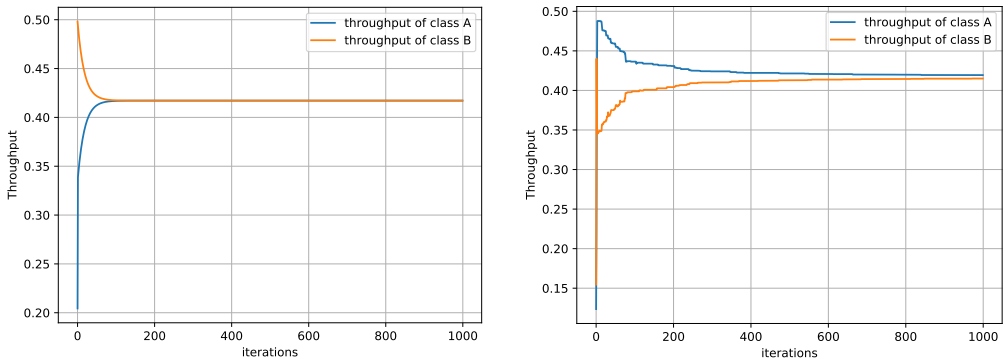


(d) A close sight of Fig. 7.1c

Figure 7.1: The optimized throughput for one class using two degrees for all loads

rapidly for some loads. The study of the optimal utility function helps to select the right pair of degrees for a given load.

Notice that for IRSA games for two-classes instead, and for instance if the selected degrees for both classes are $(m = 2, \ell = 6)$ for a load of $G = 0.93$ (see the red curve in Fig. 7.1b), many classical results from game theory cannot be directly applied owing to the fact that the utility function of the pair $(m = 2, \ell = 6)$ is discontinuous for that load as evidenced indirectly by Fig. 7.3. Indeed, the game no longer verifies the set of conditions in Property 1 because of the discontinuity of the utility function. In other words, the existence of a Nash equilibrium is no longer guaranteed. Note that the convergence of the two-classes IRSA game to a stable point was nevertheless always empirically observed (through best reply dynamics), as shown later. Precisely, the utility function of one class, described in Eq. (7.8), is a function of p_∞. In Lemma 4, we have studied the continuity and the differentiability of $p_\infty(s_0)$. The proof of Lemma 4 affirms that: for some points S (s_0, s_1, G),



(a) The achieved packet loss rate using Best Response strategy

(b) The achieved packet loss rate using Better Reply strategy

Figure 7.2: Comparison between the achieved packet loss rate of both players (classes), C_0 and C_1, when using Best Response and Better Reply strategies

the probability p_∞ might have a discontinuity (due to a potentially infinite derivative) in s_0. Alongside this proof, we went further in our simulations to study the behavior of p_∞ in the discontinuity region. For the same example, with a pair of degrees $(m = 2, \ell = 6)$ and a load of $G = 0.93$, we highlight in Fig. 7.3 the discontinuity of p_∞ in the strategy of the first player s_0 for several possible strategies of the second player s_1.

The important question is whether the restricted IRSA game can attain Nash equilibria. We have proven in Theorem 2 a sufficient condition for the convergence of better reply to NE, i.e.: $G \geq G_1$; where G_1 originates from Lemma 5 and verifies that $\forall G > G_1$, and for any s_0, s_1, T_{C_0} will be continuous, and its derivative is positive. For $(m = 2, \ell = 6)$, we numerically searched for G_1 for which the property is verified. We identify the smallest value of G_1: $G_1 \approx 1.295$. We also observed for $G \geq 0.938\ldots$, $\mathcal{S}(s_0, s_1, G) = \emptyset$, i.e. p_∞ and T_{C_0} are continuous. In the following, we experiment with all values of $G \in [0, 2]$, to explore whether convergence is still always experimentally observed. We explore the convergence towards Nash equilibrium points using the better reply algorithm, implemented in the following way: the two players (classes) select in turn a better reply based on the knowledge of the other player strategy. At each iteration, each player randomly chooses a number of distributions (100 in our case) and picks randomly one among the ones that give better or equal throughput compared to the selected distribution in the previous iteration. As shown in Fig. 7.2b, we test the better reply algorithm for a pair of degrees $(m = 2, \ell = 6)$ and a load of $G = 0.96$. The sequence of selected strategies by the better reply algorithm starts to converge towards the point $(T_{C_0} \approx 0.42, T_{C_1} \approx 0.42)$ after almost 850 rounds. Furthermore, we also used the best response algorithm for the same settings.

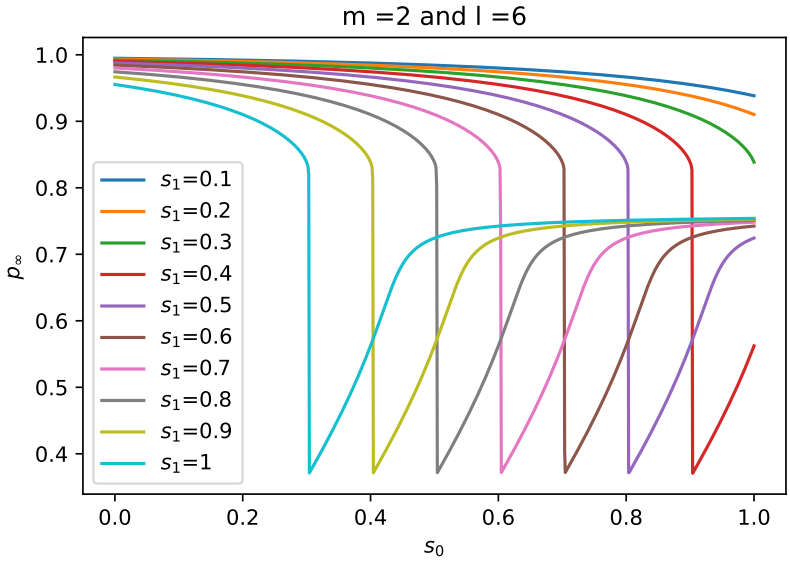
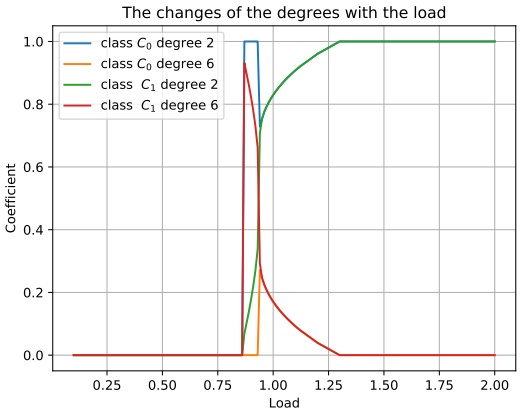
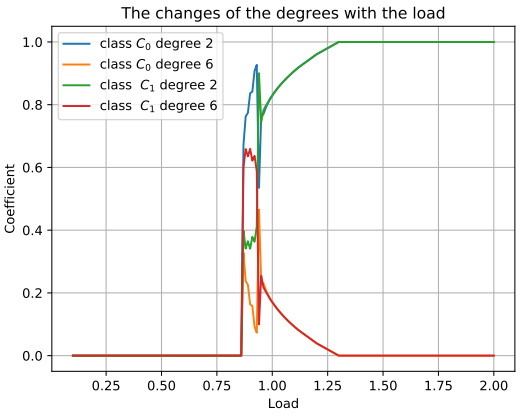


Figure 7.3: The variation of p_∞ as a function of the strategies of both players s_0 and s_1: $m = 2, l = 6$ and $G = 0.93$

In the best response dynamics, each player chooses the strategy that has the best outcome, knowing the other player's strategies. The best response algorithm converges after approximately 100 iterations towards the same equilibrium point. Fig. 7.2a shows distinctly that the best response algorithm converges much faster than the better reply algorithm, even though our convergence proof is for the better reply algorithm.



(a) The changes of the two players degrees as a function of load with the beststs response strategy



(b) The changes of the two players degrees as a function of load, with better reply strategy

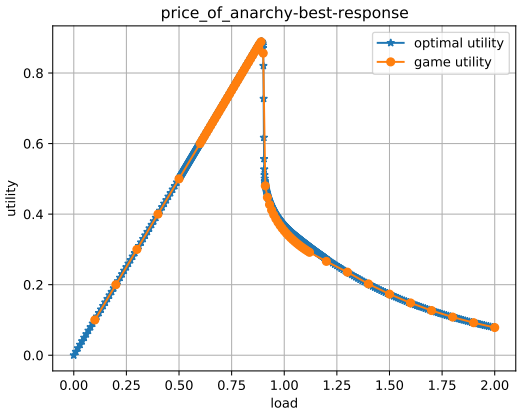
Figure 7.4: The changes of the two players strategies (degrees) as a function of load, with better reply and best response strategies

In addition to the comparison between the convergence of better reply and best response towards a Nash equilibrium, we show in Fig. 7.4 how the strategies change with the load: precisely we show the coefficients $s_0, (1 - s_0), s_1$, and $(1 - s_1)$ at the final iterations. For low loads, many possible strategies could be Nash equilibrium points. As the network is low-loaded the asymptotic PLR of 0 can be achieved, many degree distributions (strategies) with two degrees from $\{2, 3, \dots 8\}$ may achieve this asymptotic zero packet loss rate. As a consequence, these strategies can obtain a throughput equal to 0 for both players. Therefore, in both figures, Fig. 7.4a and Fig. 7.4b, we have replaced the selected strategies (coefficients) by zeros whenever we had zero packet loss rate: at convergence, the strategies are the random outcome of the iterative selection among a set of many possible strategies that lead to the same equilibrium point ($T_{C_0} = 0, T_{C_1} = 0$). Furthermore, Fig. 7.4a and Fig. 7.4b show how both players behave in the discontinuity region $G \in [0.86, 0.95]$. We observe that as the first player (class C_0) always chooses degree 2 (as $s_0 = 1$), the other player (class C_1) always chooses degree 6 (i.e. $s_1 = 0$). For higher loads $G > 0.95$, both players converge towards the same strategy and use degree 2 (i.e. $s_0 \to 1$ and $s_1 \to 1$). This is because when the network is highly loaded, repeating the packets creates more collisions and worsens the network load, hence the smallest degree is the best option for both classes.

Furthermore, we measure the inefficiency of the equilibrium for our restricted IRSA game in comparison with a centralized optimal approach obtained by solving ($\mathcal{P}_4$) for one class through computing the Price of Anarchy (PoA). In Fig. 7.5, we show how the selfish behavior of the two players (classes) is close to the optimal solution. In Fig. 7.5a, the discrepancy between the summed throughput (the sum of the throughput of both players) obtained by the IRSA game and the optimized throughput is very small. This discrepancy is only noticeable around and in the discontinuity region when the best response strategy is used. Fig. 7.5b shows the computed difference between the throughput of the optimal and the best response algorithm for the IRSA game. It appears that it is less than 2%. The same impact can be seen in Fig. 7.5c, where the better reply algorithm is used, and indeed the performance of both best response and better reply is nearly identical.

The reasoning is that, for low loads, many equilibrium points could lead to a zero packet loss rate and coincide with the optimal solutions. For high loads (more than 1.25), the players (classes) keep using the smallest degrees with a probability almost equal to 1 to reduce collisions. Otherwise, a slight change in the strategy of one player could lead to more collisions and this, in turn, makes all the players lose. For that reason, the only equilibrium points are the optimal points.

We display in Fig 7.6 the iterations where the better reply algorithm heads towards



(a) The Price of Anarchy for the best response strategy

(b) The Price of Anarchy (difference with optimal) for the best response strategy

(c) The Price of Anarchy (difference with optimal) for the better reply strategy

Figure 7.5: The Price of Anarchy for the best response and better reply strategies

convergence. It is clear that the better reply algorithm takes more iterations to converge in the discontinuity region, which also corresponds to the results shown in Fig 7.2b (average of 100 simulations per point).

In the theoretical study, we focused on restricted IRSA games. In Fig 7.7, we compare the price of anarchy between two types of games: the first game is the restricted IRSA game, where players have only two different degrees to choose from, and the second game is the general IRSA game, where the players have all possible degrees to choose from (up to a maximum degree). The goal of this comparison is to understand if both types of games lead to different results. We focus on the price of anarchy, as it is the efficiency loss caused by competition, which is one of our main questions. We observe that there is very little discrepancy between the price of anarchy obtained by both type of IRSA games (restricted

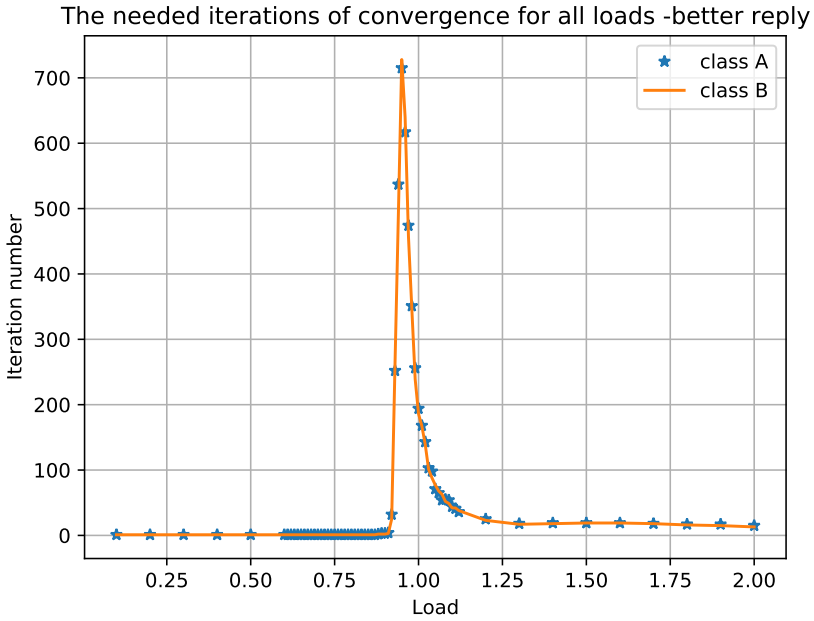


Figure 7.6: Convergence (number of iterations) with different loads

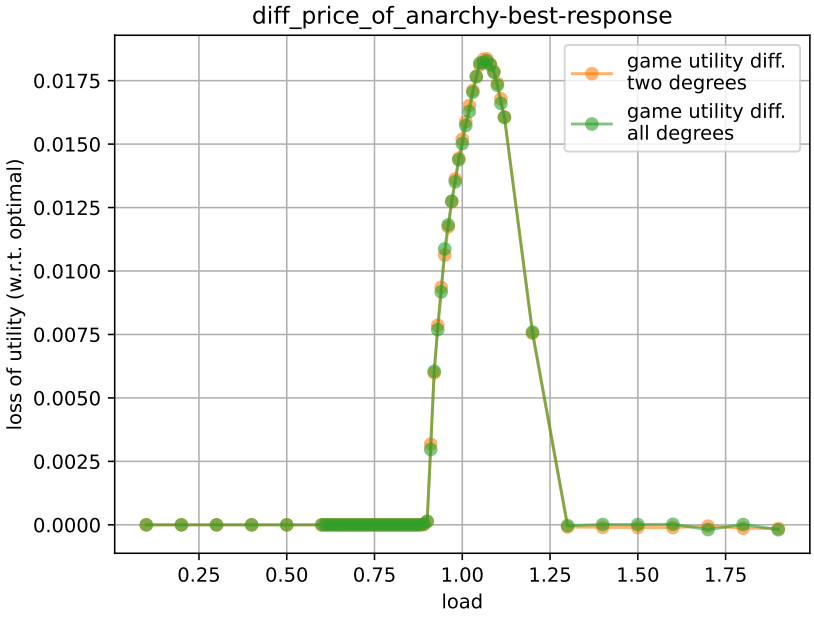


Figure 7.7: The comparison between the price of anarchy for the case of two degrees with the case of all possible degrees

or general). This illustrates empirically that the number of possible degrees in the game has little impact on the game outcomes in our scenarios, and that we might expect the general IRSA game to behave similarly to the restricted IRSA game.

7.6 Conclusion

In this section, we studied one of the modern random access protocols, Irregular Repetition Slotted Aloha (IRSA). We addressed the IRSA access scheme in a distributed fashion where users are grouped in competing classes, with users of the same class sharing the same degree distribution. The distributed approach is modeled as a non-cooperative game where the classes autonomously and selfishly set their degree probabilities to improve their own throughput. We gave proof for the existence of the Nash equilibria and how to attain them. We provided extensive numerical results that assess the notable improvement brought by the devised approaches and the small discrepancy of the distributed game-based approach in comparison with a centralized class-based IRSA approach.

Chapter 8

A Deep Learning Approach to IRSA

8.1 Introduction

In complex telecommunication systems such as IoT, where millions of devices compete for network resources, recent Machine Learning (ML) techniques have shown the capability of creating effective transmission strategies that no human could discover. ML techniques help the system to learn how to adapt to a fluctuating environment and to automatically tune the protocol parameters.

Recently, Deep Learning (DL) based approaches have been proposed for MAC layer protocols design. These DL-based solutions have shown their efficiency as they help to adapt the protocol features to the changing environment. On one side, given that efficient communication solutions are required for massive connectivity, exploiting DL techniques for optimizing transmission strategies looks promising. On the other side, the performance of IRSA degrades dramatically at high network loads, and in IRSA scenarios where short frame size is considered. Therefore, there is a need to handle such scenarios. Indeed, current optimization methods applied for IRSA have shown a limited improvement to the protocol performance in such scenarios.

The objective of this chapter is to use advanced machine learning techniques to optimize IRSA. Although, as indicated, the plain IRSA scheme can *asympotically* reach the optimal 1 [packet/slot] (Section 2.3). But yet the problem is not solved, and the ultimate challenges are (a) to ensure correct decoding just below the load $G = 1$, (b) to ensure correct decoding in the *non-asymptotic* case, e.g. small frame sizes, for which the performance can be much lower than 1 [packet/slot] (see [113, Fig 4.]), (c) to avoid the dramatic performance decrease right around the threshold load. In Chapter 6, we have applied a form of Reinforcement Learning to IRSA; however, the action was limited to the selection of

the number of the replica (the degree). In this chapter, we go further with more complex actions and with the use of Deep Learning. We present RC-IRSA, an IRSA approach with random codeword selection, where each codeword represents the transmission strategy of a user on the slots (i.e., defining precisely on which slots the user will transmit, instead of just deciding how many replicas will be transmitted). We mathematically formalize an optimization problem for the performance of RC-IRSA for short frame size. We are able to optimize it heuristically for very short frame sizes by using differential evolution. Then, we detail the main contribution of the chapter, which is an application of Deep Reinforcement Learning to optimize RC-IRSA: it results in the training of a Deep Neural Network model that defines exactly the slots on which the user will transmit. The main goal is not to improve the performance of IRSA, but to illustrate that the fine slot selection can be achieved through DRL (and to study its limits), and to use it as a framework for the next chapter. We compare the performance of our learning approach with the obtained performance of RC-IRSA through differential evolution and show that our proposed DRL approach indeed achieves very good performance for short frame size and approaches the optimal throughput values that we found by differential evolution. Our DRL approach works as a base for our proposed IRSA variant in the next chapter.

8.2 Related Work

Many research studies have addressed the problem of designing adaptive Medium Access Control (MAC) solutions for IoT networks. One research direction to optimize the users' transmission strategy is to deterministically select the slots on which the users will transmit. This can be represented essentially by a vector of 0 and 1 with one value for each slot. When considering all the users, this implies to design a *codebook* (a *dictionary*, or simply a *code*) of *access codes* (also called *sequences*, *protocol sequences* or *codewords*). The goal is to maximize the network capacity and avoid collisions. But alternately, codebooks can be used to simply identify active users.

As an introduction to the topic, we start with the classic work [114], which studies two types of superimposed codes: Zero-False Drop codes (ZFD) and Uniquely Decipherable codes (UD), and their applications in data communication. The paper shows several properties and construction methods over a wide range of parameter values. It also shows how a new class of codes, nonrandom binary superimposed codes, are constructed.

Their channel model is as follows: each node transmits a fixed zero-one sequence, and what is observed is another binary sequence of the same size, which is the *logical OR* of all

the simultaneously transmitted sequences (the terminology "*sum*" is used in [114] as they are referring to a *Boolean algebra*, but we will use the term "superposition").

For a given small positive integer m, a codebook whose codewords satisfy the following condition will be said to be uniquely decipherable of order m, abbreviated UD_m: every superposition of up to m different codewords is distinct from every other superposition of m or fewer codewords. A simple example of a list of 7-bits codewords from [114]

$$s_0 = \begin{pmatrix}1 & 1 & 0 & 0 & 0 & 0 & 0\end{pmatrix} \qquad s_1 = \begin{pmatrix}1 & 0 & 1 & 0 & 0 & 0 & 0\end{pmatrix}$$
$$s_2 = \begin{pmatrix}0 & 1 & 0 & 0 & 1 & 0 & 0\end{pmatrix} \qquad s_3 = \begin{pmatrix}0 & 0 & 1 & 1 & 0 & 0 & 0\end{pmatrix}$$
$$s_4 = \begin{pmatrix}0 & 0 & 0 & 1 & 1 & 0 & 0\end{pmatrix} \qquad s_5 = \begin{pmatrix}0 & 0 & 0 & 0 & 1 & 0 & 1\end{pmatrix}$$
$$s_6 = \begin{pmatrix}0 & 0 & 0 & 0 & 0 & 1 & 1\end{pmatrix} \qquad s_7 = \begin{pmatrix}0 & 0 & 1 & 0 & 0 & 1 & 0\end{pmatrix}$$

contains no duplicated codewords. In addition to that, when augmented with all $\binom{8}{2} = 28$ pairwise superposition of codewords, it still contains no duplicates. Thus, this set of eight codewords constitutes a UD_2 code. Observe for instance that the superposition $\begin{pmatrix}0 & 1 & 0 & 1 & 1 & 0 & 0\end{pmatrix}$ can uniquely be obtained from $s_2 \vee s_4$.

One usage of such code in communications is the following: such a UD_m codebook can be used, with a distinct codeword for each node. We can imagine a slotted frame (different from the IRSA frame), where nodes can transmit jamming signals (instead of packets) when there is a 1 in their sequence. At the base station (or any receiver), an energy detector can be used to identify in which slots jamming signals have been transmitted. If less than m nodes have been active and transmitted their code, then the receiver can uniquely retrieve the identity of the nodes.

A related line of work in [115] presents a grant-free random access scheme for short-packet communication on a collision channel without feedback, where user identities are conveyed through their activity patterns, i.e. their codewords. SIC is not used. Assuming that the population size is N and at most d devices can be active at any given time, the study shows that group testing codes can be used to design random access protocol sequences with minimal length. They consider the frame-synchronous and asynchronous cases, where each user has a corresponding codeword of length t. The receiver is assumed to know the user's codewords. Their approach leads to codes of length $t = \Theta(d \log N)$ for the asynchronous case. They minimize the complexity by minimizing the length of the codeword needed to decode d active devices at a time. Because SIC is not used, the scheme can also be used as a user identification scheme, e.g. if users transmit one bit, or if they use jamming signals in slots.

The previous codebooks were designed to perform user identification (an area where also *compressive sensing* has been proposed [116]). But it has also been used for packets transmission. The concept of using codebooks has been famously introduced in early work, along with the *collision channel without feedback* (and without frames) in [117]. Codebooks can be naturally used to specify on which slot users should transmit packets as in [117], where in addition, there is no frame boundary, hence each user starts transmitting at an arbitrary slot. [117, Eq. (15)] gives an example for two users who use the codebook:

$$s_0 = \begin{pmatrix} 1 & 0 & 1 & 0 \end{pmatrix} \qquad s_1 = \begin{pmatrix} 1 & 1 & 0 & 0 \end{pmatrix}.$$

SIC is not used. But as one can see, no matter on which slots the users will start transmitting, both packets will be recovered.

In the case of IRSA, it has been already proposed to construct a large codebook with one codeword for each user with *graph-defined* IRSA (G-IRSA) [118], based on the design of an LDPC code. This approach is not fully scalable when the proportion of active users decreases [119, Sec. IV.A]. This area is also linked to Code-Domain NOMA [120].

Recent work goes further and constructs capacity-achieving codebooks [121] that assume SIC. One of their example is for three users [121, SectionV.C]:

$$s_0 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \; s_1 = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \; s_2 = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Here, with SIC, all three users will get recovered, no matter in which slot they start. Additionally, instead of simply repeating the initial packet at the positions of the 1, some coding can be applied, such as MEBC-coding [117], and the same codebook can be used to transmit encoded packets, giving a global data rate of 1 packet per slot.

Back to the framed version of random access methods: one can define an M-Interference Cancelling code (or codebook), denoted M-IC code, of length n as a set of codewords such that: each active user has at least one successful transmission during every n consecutive time slots provided that the number of simultaneously active users is less than or equal to M after iterative decoding with SIC.

In this spirit, the authors of [119] have introduced the design of deterministic random access codes for the ultra-reliability region, targeting a packet loss rate less than 10^{-5}. They considered simple deterministic variations of CRDSA (D-CRDSA), where the number of repetitions is fixed (differing from IRSA, where the degree is drawn from a distribution), and compared them to the original random versions.

The study shows that larger codewords with more repetitions might maximize the user population and minimize the packet loss rate in their numerical experiments and at low

load. Notice that they use M-IC codes with more than M users, but show that they are still performing better than selecting random slots (up to a certain point). The work studies codes based on Steiner systems and discuss why prior codes based on LDPC designs [118] have drawbacks for URLLC scenarios. Designing M-IC codes that support a large user population is a hard problem. The paper also points out the limitations of any approach that is based on superimposed codes, for instance, the codes which satisfied the 3-IC condition could only support relatively small user populations as shown by the provided lower bounds on the supportable user population for 3-IC codes.

Finding M-IC codes, as constructing any kind of code in general, is not easy. Currently, there are some examples of constructions, but there exists no mathematical constructions nor automated ways to construct *all* such codes. To avoid the complexity of finding IC codes with the current tools or search algorithms, some research work used machine learning techniques. In [122], The authors apply a deep reinforcement learning (DRL) based algorithm to search for IC codes, using specific metrics and reward functions according to the underlying mathematical constraints. They model the construction process of an M-IC code with N codewords each of length n as an episodic symbol-filling game based on Markov Decision Processes (MDP). Each episode represents the construction of a codebook, C and it begins with the state where the codebook is empty, i.e. is constituted of empty codewords. Each step of the episode selects an action sequentially, to add ℓ bits to one codeword which is still shorter than the frame length. Each episode ends when the codebook is full, i.e. every codeword covers the frame length. At the end of each episode, C is evaluated by one metric $m(C)$ that counts the number of subsets of $m \leq M$ codes that cannot be fully decoded, and a reward $r(C)$ for this episode is linearly derived from $m(C)$. In particular, C is an M-IC code if and only if $m(C) = 0$. This MDP formulation is then used to search for codebook in a tree manner, with a Monte-Carlo Tree Search (MCTS), and a Deep Neural Network model is classically used to guide the search. The search results have indicated that the algorithm can efficiently discover IC codes, while the simulation results have shown that the discovered IC codes can produce significantly lower failure probability than random slot selection under the same latency requirements (again, even for several active users greater than M), and thus are more suitable for URLLC. The introduced codes in [122] are not necessarily optimal. A future research direction is to design a general construction for close-to-optimal IC-codes.

They also give an example of 4-IC code in [122, Section III.C]:

$$s_0 = \begin{pmatrix}0 & 0 & 0 & 0 & 1 & 0 & 0\end{pmatrix}, \quad s_1 = \begin{pmatrix}0 & 0 & 0 & 0 & 0 & 1 & 0\end{pmatrix}, \quad s_2 = \begin{pmatrix}1 & 0 & 1 & 0 & 0 & 0 & 0\end{pmatrix},$$
$$s_3 = \begin{pmatrix}0 & 0 & 0 & 1 & 0 & 1 & 0\end{pmatrix}, \quad s_4 = \begin{pmatrix}1 & 1 & 0 & 1 & 1 & 0 & 0\end{pmatrix}, \quad s_5 = \begin{pmatrix}0 & 0 & 1 & 1 & 0 & 0 & 0\end{pmatrix},$$
$$s_6 = \begin{pmatrix}0 & 0 & 0 & 0 & 1 & 0 & 1\end{pmatrix}, \quad s_7 = \begin{pmatrix}1 & 0 & 1 & 0 & 0 & 1 & 1\end{pmatrix}, \quad s_8 = \begin{pmatrix}0 & 1 & 0 & 0 & 1 & 0 & 1\end{pmatrix},$$
$$s_9 = \begin{pmatrix}0 & 1 & 1 & 0 & 0 & 0 & 0\end{pmatrix}, \quad s_{10} = \begin{pmatrix}1 & 0 & 0 & 0 & 0 & 0 & 0\end{pmatrix}.$$

Many other works have proposed codes for random access. One example is Learn2MAC [91] where the users have their own individual codebooks, and optimize the transmissions (maximize their individual utility) with online learning. More precisely, codewords are randomized for each user, but also a probability of selection is associated with each codeword: the codeword probability distribution is updated by the online learning algorithm. We also became recently aware of quite recent and interesting work combining random access and machine learning [123].

In the next section, we shed the light on related studies that consider IRSA with short frame length, and we introduce our prior work on applying a deep learning approach on short frame length IRSA.

8.2.1 Prior Work on Short-Frame Length IRSA

8.2.1.1 Short-Frame Length IRSA

In the literature, there is a lack of precise results for optimizing IRSA for smaller frame sizes. The main challenge to optimize IRSA with short frame size, which is the realistic scenario, is that there is not a simple mathematical approach to track the decoding process and compute the expected number of decoded users as with density evolution for asymptotically large frame lengths.

One of the most accurate performance estimates for IRSA with short frame length is possibly in [113], where a short-frame (SF) approximation for the packet loss rate of IRSA is particularly suitable for very short frames, up to 50 slots, has been introduced. The idea of this work is to write the packet loss rate as a function of all possible stopping sets in order to compute the exact value of the packet loss rate. It is then possible to write and solve numerically an optimization problem, as with density evolution. Even for small frames, the number of stopping sets would rapidly become intractable, yielding unmanageable complexity. To limit it, the number of considered stopping set in [113], has been limited to only the stopping sets that have e_m edges, where the parameter e_m has been introduced as an upper bound on the number of edges in the set.

One illustrative result is that short-frame length IRSA can be far from reaching a throughput near 1 [packet/slot] with near 0% PLR. The results in [113, Fig. 3 & Fig. 4] show unfavorable throughput/packet loss trade-off, despite that the simulations were done for a fixed number of users $M = 5$ in [113, Fig. 3] (PLR is 5% for $G = 0.5$) and for a constant load $G = 0.4$ in their [113, Fig. 4] (PLR is 4 %), to compare for instance to the results of our Fig. 3.8.

8.2.1.2 Our Prior Work: Deep-IRSA

One of the motivations for the work presented in this section is to improve IRSA performance, in particular when the size of the frames is finite, hence the assumption of "asymptotically infinite frame length" is far from being verified. To address this, our prior work in [124] has proposed Deep-IRSA, which uses a deep learning framework to optimize the performance of IRSA. We focused on the short frame length scenarios for different network parameters, with also the ability to enable two options: one which allows re-transmissions and another which has user priority classes (see the definition of priority classes in Section. 6.4.3). We applied Deep Reinforcement Learning techniques to solve this problem.

We first present how we applied DRL to IRSA with Deep-IRSA, as a predecessor to the more advanced variants, which will be presented in the next chapter. As in Chapter 6, an MDP model is adopted: the action of one user is the selection of a degree (after that, slots are still randomly selected), the reward is a global reward that simply counts the number of decoded users. In Deep-IRSA, additionally, there is some observed state. The main idea is that users can receive feedback from the BS to inform them about the number of collisions in the past frame and if their own transmissions succeeded. Then, each user adds an internal state to the received feedback: its class of service and whether it is a re-transmitter or not when any of these options is enabled. The feedback, together with the internal state, both constitute the observed state of one user. It is expected to influence the choice of user actions. Deep Reinforcement Learning with the Proximal Policy Optimization (PPO) [125] is used for training: a Deep Neural Network (DNN) model takes the state as input and outputs an action corresponding to a degree. PPO is a policy-based method, and the DNN model outputs a stochastic policy, e.g. the probability to select each degree. Without options, the DNN essentially outputs a degree distribution that is adjusted depending on the load (and its behavior becomes more complex when there are options). Note that the application of PPO to IRSA will be extensively explained in the next chapter.

We are interested in the obtained maximum performance in case of small frame size and a few numbers of users, $M = 50$. We varied the load and used Deep-IRSA as an optimiza-

tion method. For reference, we independently and numerically optimized distributions by the use of simulations. This becomes an obvious stochastic optimization problem: using the *Sample Average Approximation* (SAA) [126], we could just define an approximated objective function as the average of 100 simulations, that we can optimize with any method, and we opted for differential evolution. The results of the throughput obtained with both methods are reproduced in Fig. 8.1. The throughput we obtained for DRL shows a comparable performance against the differential evolution results. The Fig. 8.1 shows that IRSA with a short frame length is suffering from low throughput, e.g. a maximum of 0.65, whether it is optimized by differential evolution or DRL approach.

One way to enhance this work is to incorporate slot (or a group of slots) selection, which enables the DNN model to intelligently select slots instead of using random degree-then-slot selection. In effect, this would be equivalent to generating codewords.

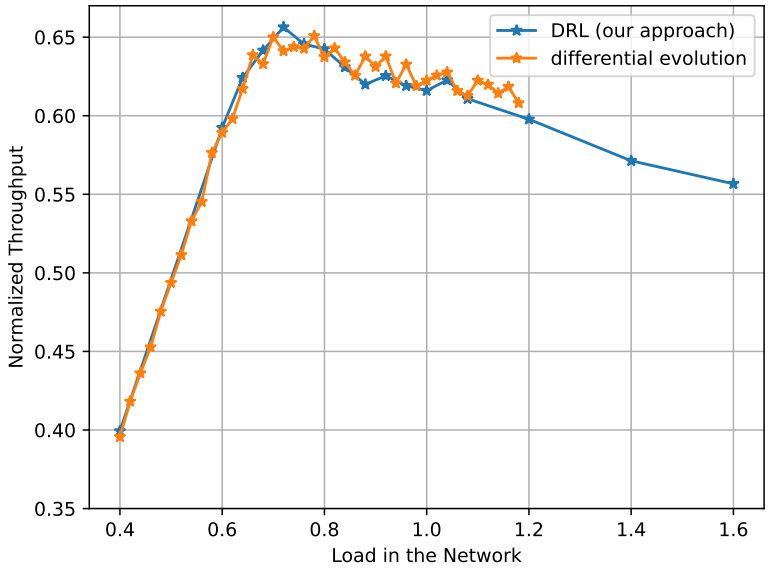


Figure 8.1: Throughput for (Relatively) Short Frame-Length, for Deep-IRSA, and for Differential Evolution with Simulations

8.3 RC-IRSA: IRSA with Random Codeword Selection

In this chapter, we are interested in the use of codebooks for the transmission phase of IRSA, as done for instance in G-IRSA [118] or [119]. However, unlike them, we do not intend to allocate deterministically a codeword to each user. We assume that the users are completely undifferentiated. In that case, it might be natural to have a codebook shared by all users (or by a subset of users), and to randomly select a codeword according to some distribution (as done in Learn2MAC [91] for instance). We denote this version of IRSA as

Random Codeword selection IRSA (RC-IRSA). Notice that this version is not expected to be more efficient than individual codeword assignment, but it will serve as a basis for the improved method in the next chapter.

In this section, we provide a general definition of RC-IRSA and the related optimization problem formalization to find optimal IRSA codebooks and codeword probability distributions. An IRSA codebook is a set of codewords, where each codeword represents the positions of the slots where the user sends their replicas. Let $\mathcal{S}$ denote a binary $\{0, 1\}$ scheduling code for deterministic access, which consists of ω codewords of length w, i.e., $\mathcal{S} \triangleq \{s_0, s_1, ..., s_{\omega-1}\}$. To transmit a packet, the user will use a codeword $s_i \triangleq (s_i[0], s_i[1], ..., s_i[w-1])$, for $i = 0, 1, 2, ..., \omega - 1$.

If a user uses the codeword s_i to transmit its packet in the frame, it transmits at the slot t if and only if $s_i[t] = 1$, and thus it repeats the same packet $\phi_i \triangleq \sum_{j=0}^{N-1} s_i[j]$ times in the frame, where N is the number of slots. Each codeword $s \in \mathcal{S}$ is associated with a probability of selection π_s. The codeword probability distribution $(\pi_s)_{s\in\mathcal{S}}$ replaces the degree probability distribution $(\Lambda_i)_{i=0...L}$ of IRSA.

In line with the OP problem described in $(\mathcal{P}_1)$ and in the classical IRSA system, the aim is to find the best codebook probability distribution that maximizes the system throughput at a given network load G.

Given a codebook $\mathcal{S} = \{s_0, s_1, ..., s_{\omega-1}\}$, with $s_i = (s_i[0], s_i[1], \ldots, s_i[w-1])$ for all $i \in \{0, 1, \ldots, \omega - 1\}$ and such that for all $t \in [0, w-1]$, $s_i[t] \in \{0, 1\}$. The optimization problem is:

$$\begin{aligned} \underset{(\pi_{s_i})}{\text{maximize}} \quad & T(\mathcal{S}, \pi_{s_0}, \ldots, \pi_{s_{\omega-1}}) \qquad (\mathcal{P}_7) \\ \text{subject to} \quad & 0 \leq \pi_{s_i} \leq 1, \ \ \forall i \in \{0, 1, ..., \omega - 1\} \\ & \sum_{i=0}^{\omega-1} \pi_{s_i} = 1 \end{aligned}$$

where T is the throughput. Note that, in this chapter, we express the throughput as "the average number of decoded users" instead of the definition that we used before. ω is the total number of codewords. Note that the throughput computation should be adjusted to take into account codewords probabilities.

We extend the optimal formulation of the optimization problem in $(\mathcal{P}_7)$, to a system that has a different but finite number of classes $K = (C_0, ..., C_{n-1})$, and a fixed global load G. For each class $c \in K$, there is a unique codebook. We can rewrite the optimization problem:

	Slot 0	Slot 1
A	1	0
B	1	1

	Slot 0	Slot 1
A	1	1
B	1	0

	Slot 0	Slot 1
A	1	0
B	0	1

	Slot 0	Slot 1
A	0	1
B	1	1

	Slot 0	Slot 1
A	1	1
B	0	1

	Slot 0	Slot 1
A	0	1
B	1	0

Figure 8.2: The transmission patterns that lead to the decoding of two users A and B sending on 2 slots

$$\begin{aligned} &\underset{(\pi_{s_{c,i}})}{\text{maximize}} && T_c(\mathcal{S}_c, \pi_{s_{c,0}}, \ldots, \pi_{s_{c,\omega-1}}) && (\mathcal{P}_8)\\ &\text{subject to} && 0 \leq \pi_{s_{c,i}} \leq 1, \quad \forall i \in \{0, 1, ..., \omega - 1\} \\ & && \sum_{i=0}^{\omega-1} \pi_{s_{c,i}} = 1 \end{aligned}$$

where $s_{c,i}$ is the codeword i of the codebook $\mathcal{S}_c$ and c is the class index and $c \in K$.

8.3.1 A Mathematical Analysis of RC-IRSA with 2 users and 2 slots

In this section, we introduce a scenario of RC-IRSA protocol in the simplest case: 2 users are competing for 2 slots. We present a mathematical analysis of this simple classical IRSA scenario. We suppose an IRSA scenario of 2 slots and 2 competing users A and B. Both users use the IRSA protocol with a maximum repetition degree of 2 and with a minimum repetition degree of 0. The set of possible codewords[1] is $\mathcal{S} = \{00, 01, 10, 11\}$, with the codeword probability distribution $(\pi_{00}, \pi_{01}, \pi_{10}, \pi_{11})$. The 1 in a codeword means that the user sends a packet on the slot, and the 0 means that the user does not send a packet on the slot. As a consequence, the number of all possible combinations of the codeword choices of both users is $2^4 = 16$. The six combinations that allow decoding both users are shown in Fig. 8.2. The probability that both users are decoded after sending on two slots is given as:

$$P_s = 2\pi_{01}\pi_{10} + 2\pi_{01}\pi_{11} + 2\pi_{10}\pi_{11} \tag{8.1}$$

[1]For the ease of presentation, here, we index the codewords with their binary representation (e.g. "01" for $(0, 1)$) instead of indexing them with integers.

In the case of 2 users, the number of decoded users is simply $2P_s$, hence we can equally maximize P_s. By using the method of Lagrange multipliers, we can find the probability to use each codeword such that we maximize the success probability:

$$\begin{aligned}
&\mathcal{L} = 2\pi_{01}\pi_{10} + 2\pi_{01}\pi_{11} + 2\pi_{10}\pi_{11} - \lambda\left(\pi_{00} + \pi_{01} + \pi_{10} + \pi_{11} - 1\right)\\
&\frac{\partial \mathcal{L}}{\partial \pi_{00}} = -\lambda = 0\\
&\frac{\partial \mathcal{L}}{\partial \pi_{01}} = 2\pi_{10} + 2\pi_{11} - \lambda = 0\\
&\frac{\partial \mathcal{L}}{\partial \pi_{10}} = 2\pi_{01} + 2\pi_{11} - \lambda = 0\\
&\frac{\partial \mathcal{L}}{\partial \pi_{11}} = 2\pi_{01} + 2\pi_{10} - \lambda = 0\\
&\frac{\partial \mathcal{L}}{\partial \lambda} = \pi_{00} + \pi_{01} + \pi_{10} + \pi_{11} - 1 = 0
\end{aligned} \tag{8.2}$$

Solving the set of equations in Eq.(8.2), and also checking the boundary conditions, gives the optimal probability to use each codeword as: $\{\pi_{01} = \pi_{10} = \pi_{11} = p, \pi_{00} = 1 - 3p\}$. Choosing $p = \frac{1}{3}$ will maximize the success probability:

$$P_s = 2 \cdot 3 \cdot \left(\frac{1}{3}\right)^2 = \frac{2}{3} \approx 0.667\ldots \tag{8.3}$$

and the throughput is given by:

$$T = 2 \cdot P_s = 2 \cdot \frac{2}{3} \approx 1.334 \quad \text{decoded users/frame}$$

8.3.2 A Mathematical Analysis of RC-IRSA With M Users and N Slots

Let us consider a system of M users, competing to send their packets on N slots using IRSA protocol. Each possible transmission of each of the M users on the N slots is represented by a vector of ones and zeros, considered as a possible codeword of length N. The ones refer to the transmission positions on the slots, and the zeros represent the slots where the user has not transmitted. Thus, the possible codewords that could be sent by a user, form a codebook $\mathcal{S} = \{s_0, s_1, ..., s_{\omega-1}\}$, which has: $\omega = 2^N$ codewords. The total number of possible codeword combinations that could be sent by the M users on the N slots is calculated as: $N_{com} = \omega^M$.

We define D as an M-dimensional array. Each element $d_{i_1,i_2,..,i_M} \in D$ represents the number of decoded users after using the codewords $s_{i_1}, s_{i_2}, ..., s_{i_M}$. In other words, each

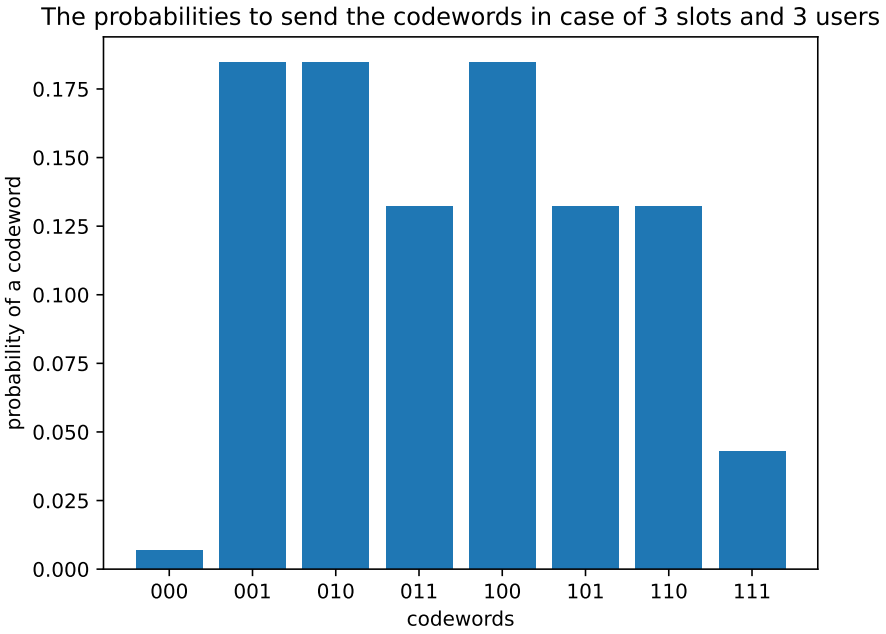


Figure 8.3: The probability to send each possible codeword in a system of 3 slots, 3 users

element of D is the number of decoded users after using one combination of the selected codewords by the M users in the system. Each codeword $s_i \in \mathcal{S}$ has a length N. The probability of decoding all users in the frame (the probability of success) is calculated as follows:

$$P_s = \sum_{i_1=0}^{\omega-1} \sum_{i_2=0}^{\omega-1} \sum_{i_3=0}^{\omega-1} ... \sum_{i_M=0}^{\omega-1} \pi_{i_1} \times \pi_{i_2} \times \pi_{i_3} \times ... \times \pi_{i_M} \times d_{i_1,i_2,i_3,..,i_M} \tag{8.4}$$

where π_i is the probability to send the codeword $s_i \in \mathcal{S}$.

Note that the number of possible codewords ω grows exponentially with the number of users M. Therefore, it becomes very difficult to mathematically evaluate the probability of success (described in Eq. 8.4) as the number of slots $N \geq 3$. Alternatively, the probability to use each codeword can be computed numerically using differential evolution.

A small illustrative example could be for $M = 3$ users sending on $N = 3$ slots. The number of possible codewords for one user is: $\omega = 2^3 = 8$ codewords. The total number of possible combinations is $N_{com} = 8^3 = 512$. Computing manually 512 values of the D array, and optimizing manually Eq. (8.4) is a complicated task. Thus, we use differential evolution to compute the probability to use each codeword such that the throughput is optimized.

Fig. 8.3 shows the probability to use each possible codeword after solving the optimization problem $\mathcal{P}_7$, We note from the figure that the probabilities to send one replica in one of the three slots are identical: $\pi_{100} = \pi_{010} = \pi_{001} = 0.1847$. The probabilities to send two replicas on two of the three slots are also identical: $\pi_{110} = \pi_{101} = \pi_{011} = 0.132$.

users \ slots	2	3	4	5	6
2	1.333332	1.714285	1.866634	1.935483	1.968247
3	0.969525	1.673334	2.288013	2.615522	2.791376
4	0.899124	1.440759	2.082369	2.789802	3.278239
5	0.864823	1.367171	1.922436	2.546374	–

Table 8.1: The obtained throughput of RC-IRSA via differential evolution for different number of users and slots

Thus, an optimized codebook of eight codewords has been generated, with the probability to use each codeword. Note that we can extract the coefficients of the optimal degree probability distribution from the generated codebook as: $\Lambda_0 = \pi_{000}, \Lambda_1 = \pi_{100} + \pi_{010} + \pi_{001}$, $\Lambda_2 = \pi_{110} + \pi_{101} + \pi_{011}, \Lambda_3 = \pi_{111}$, but the inverse is not possible in general. In other words, we cannot extract an optimal codebook from an optimal degree distribution. Note that the codeword degree distribution happens to be the same as a degree distribution for this example, but this is not expected to be true in other variants of RC-IRSA. The number of decoded users for this example is ≈ 1.673 decoded users/frame.

Table. 8.1 shows the obtained throughput by using differential evolution for different scenarios.

In this context, two questions arise: first, can we apply a DRL approach to find an optimal codebook that maximizes the throughput? The second question is whether extra knowledge about the users can help to synchronize the nodes, such that adding a sensing phase before the IRSA transmission may lead to optimizing the obtained codebook and maximizing the throughput. In the next section, we answer the first question, while we explore the other question in the next chapter.

8.3.3 Deep RC-IRSA: A Deep Learning analysis of IRSA with 2 users and 2 slots

In prior work, [124], we optimized IRSA using Deep Reinforcement Learning to obtain Deep-IRSA, where we learned the degree distribution. In this section, we use a deep learning approach to find the optimal codebook that maximizes the throughput in case of IRSA scenario of 2 users and 2 slots. We consider the same methodology to learn optimal codewords' probability distributions for IRSA instead of degree distributions. The application of DRL is done according to the following principles:

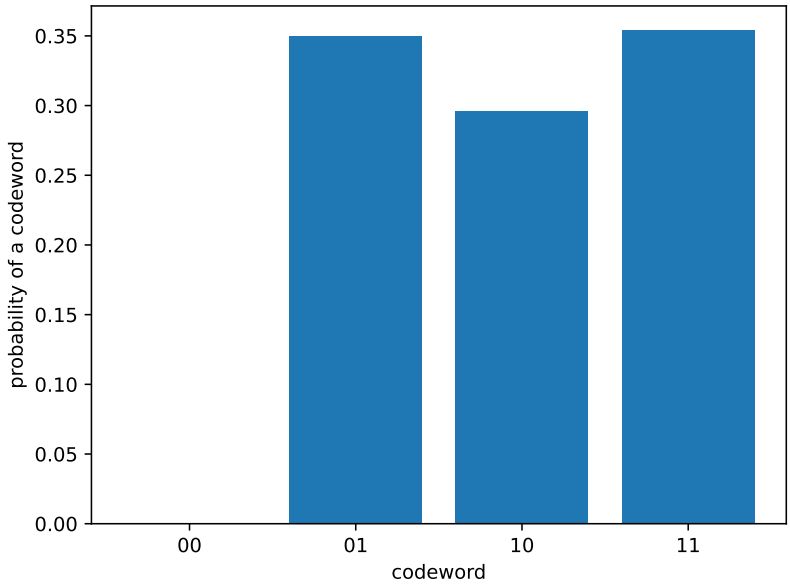


Figure 8.4: The probability to use each codeword in case of 2 users-2 slots system

slots / users	2	3	4	5	6
2	1.333117	1.701885	1.853383	1.927235	1.959373
3	0.967106	1.658842	2.271073	2.581683	2.754746
4	0.897425	1.436308	2.065276	2.724494	3.208501
5	0.857731	1.361922	1.908074	2.536552	3.160870

Table 8.2: The obtained throughput of RC-IRSA via the deep learning approach for different number of users and slots

- IRSA is recast in the DRL framework, it becomes a multiagent system, where each node is an agent.
- The action of each agent, is essentially the codeword selection $s_i \in \mathcal{S}$.
- At each episode, we generate a state, where we add a source of randomness which acts as a *source of entropy* to help the model to generate different codewords for the same actual state. Indeed, keeping the same state for both users in all the episodes will lead to choosing statically the same actions, as the agents always observe the same value, and the neural network model is deterministic. On the contrary, changing the state helps to make the actions change dynamically to attain the optimal values.

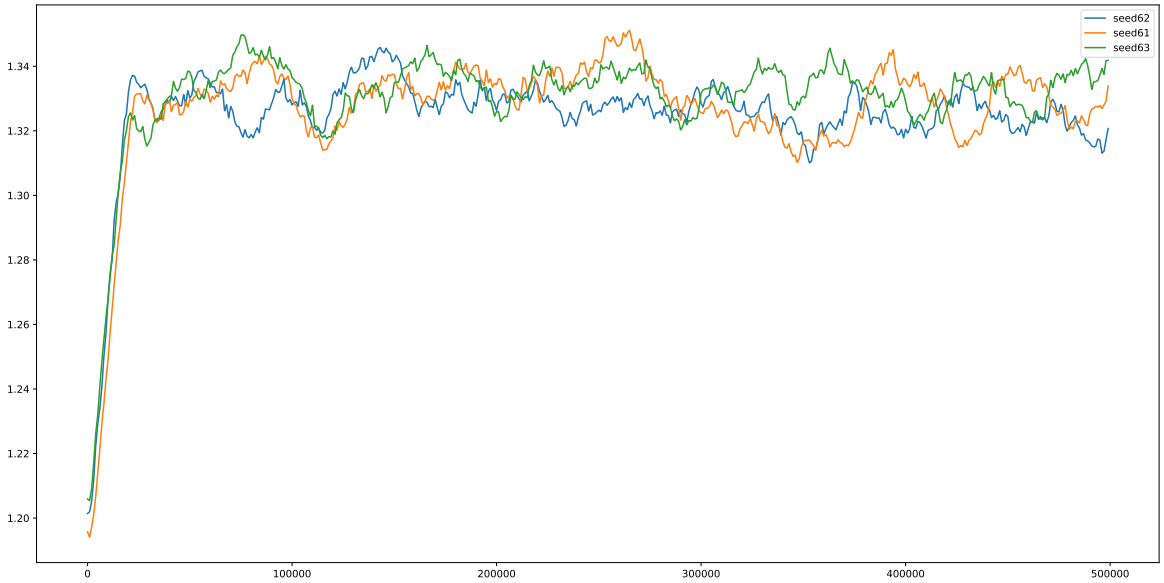


Figure 8.5: The convergence of IRSA throughput in case of 2 users-2 slots system

8.3.4 Deep RC-IRSA with M Users and N Slots

Fig. 8.4 shows the probability to use each codeword. Note that we are computing *empirical probabilities* observed from episodes after the convergence of our DRL approach (the last episodes). The DRL model does not give actions probabilities as an output, but it gives the selected slots. Each empirical probability to use an action (a codeword) has been obtained by counting how many times the same action was used after the model has converged (e.g. in the last episodes).

The resulting observed probabilities from the DL approach are:

$$\{\pi_{01} = 0.34, \pi_{10} = 0.3, \pi_{11} = 0.36, \pi_{00} = 0\},$$

while we obtained $\left\{\pi_{01} = \pi_{10} = \pi_{11} = \frac{1}{3}, \pi_{01} = 0\right\}$ in the theoretical approach.

Fig. 8.5 shows the convergence of the DRL approach towards the optimal throughput $T \approx 1.334$. Note that we used a smoothing filter that takes the arithmetic average of each value with its neighbor.[2] The size of the smoothing is $20,000$, which means that the average is taken on each successive $20,000$ values. The figure shows that our DRL approach converges after approximately $30,000$ learning episodes towards the optimal value that we found in the Section. 8.3.1.

Fig. 8.3.1 compares three different simulations for the same scenario, but with different seeds. The figure shows that the DRL with different seeds attains the optimal throughput.

[2]For more information about the smoothing function, see: https://docs.scipy.org/doc/scipy/reference/generated/scipy.ndimage.uniform_filter1d.html

In this section, we show more results of using Deep RC-IRSA to optimize the throughput in the case of the IRSA scenario of ≥ 2 users and ≥ 2 slots. Table. 8.2 shows the IRSA throughput when the DRL approach is used to optimize the performance. Comparing these values with the throughput values in Table. 8.1 that were obtained through differential evolution, confirms that our DRL approach works perfectly as it achieves the same optimal values obtained by differential evolution.

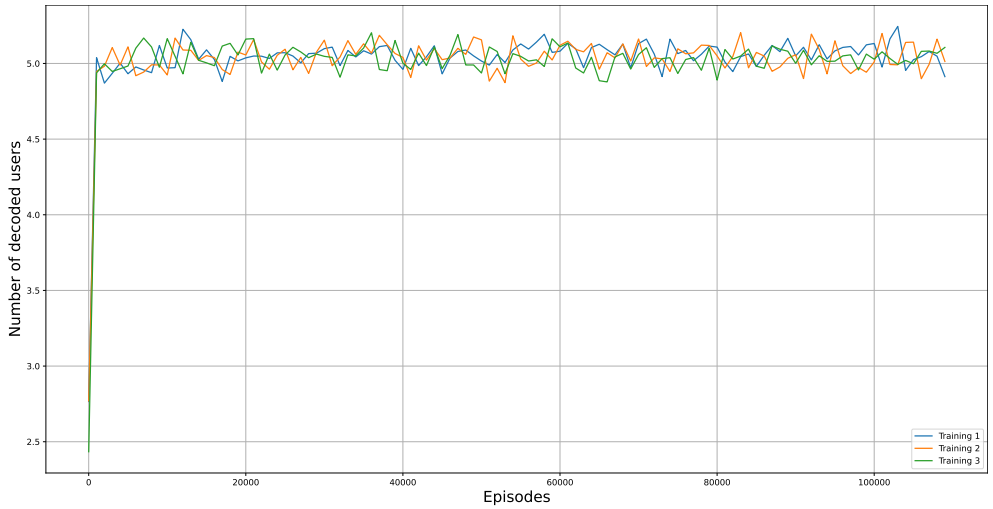
Fig. 8.6 shows the training of our DRL model for a different number of users and slots. We did 3 different trainings with three different seeds. In each case, we note that our DRL approach converges fast towards a value of the throughput of around 50%. In other words, the number of decoded users that we obtain with our DRL approach is always around half of the competing users. For instance, the obtained throughput in Fig. 8.6a, for a scenario of 10 users and 10 slots, is around 5. Note that, the optimal values of throughput obtained through differential evolution, for the scenarios where the number of users is equal to the number of slots, were always approximately 0.5 (see Table. 8.1). It becomes very complicated to mathematically compute the optimal throughput when the number of slots is ≥ 2. In addition, solving the described optimization problem in ($\mathcal{P}_7$) through differential evolution for more than ≥ 10 users becomes an extremely complicated task as the number of codewords combinations grows exponentially with the number of users.

Our DRL approach has proven to achieve a throughput around 50% for a scenario of ≤ 25 users competing for ≤ 25 slots. For Deep RC-IRSA with more than 25 users, our DRL approach does not seem to work correctly for two reasons: First, adding more slots will increase, exponentially, the action space of the neural network. The second reason is related to large stopping sets. As we do not have any restrictions on the taken actions after initialization and after the beginning of the training, the neural network could try some actions that include many 1 and a few zeros. This will lead to blocking many users in large stopping sets and ultimately, they will have almost always a zero throughput and leads to a sparse network problem. One solution to resolve such a problem is to force a penalty (negative reward), when the taken action is not favorable, and also increase the number of learning episodes.

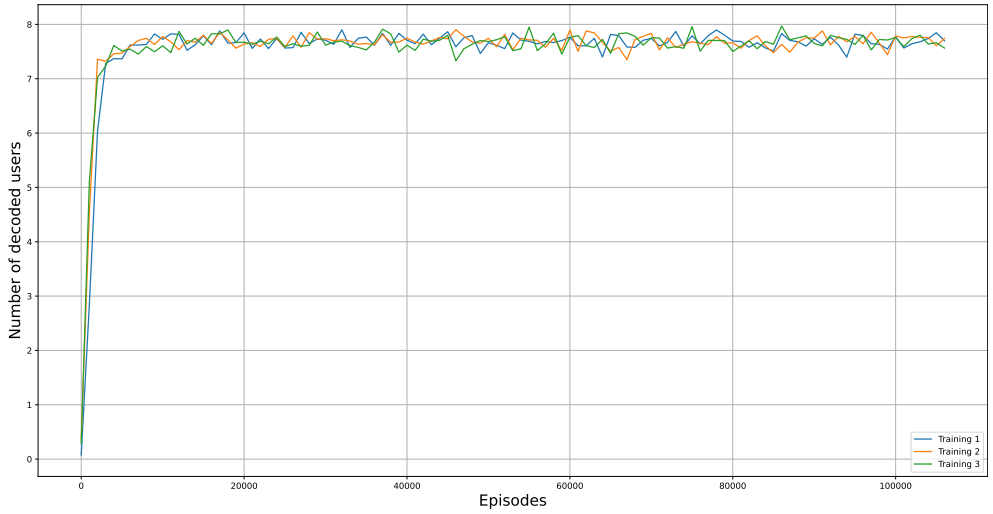
8.4 Conclusion

In this chapter, we aimed to optimize IRSA with small frame size scenarios. We first mathematically presented the problem of optimizing IRSA for small finite frame size and then, we optimized it through differential evolution. We applied Deep Reinforcement Learning

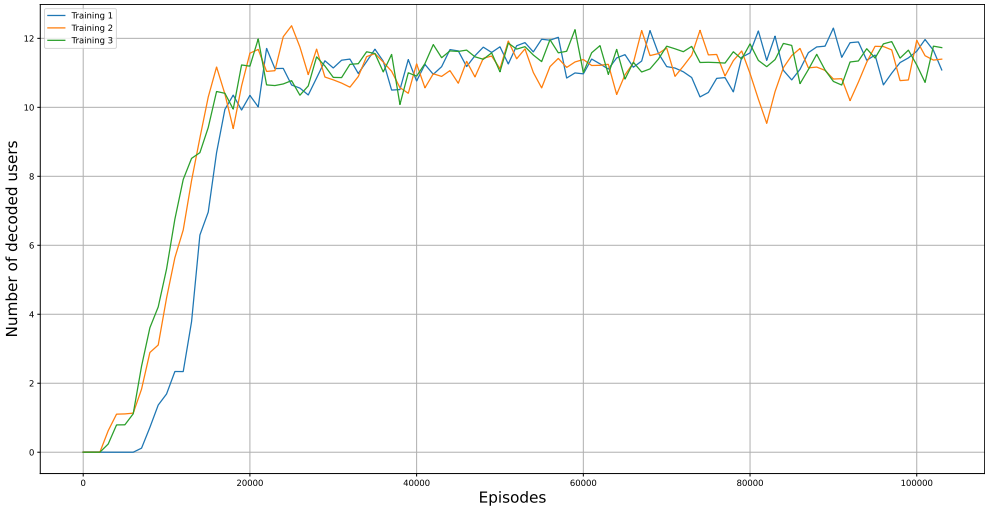
techniques to solve this problem. The results have shown that our DRL approach, Deep RC-IRSA, attains the optimal values that we found through differential evolution. More generally, our work provides a generic method to optimize IRSA and its different variants, and importantly it is able to do that by selecting the slots instead of just selecting the degree, which shed the light on potentially a much better way of optimization such protocol. Our proposed method proves that it is a promising alternative to known methods in the literature (differential evolution, density evolution, etc.), especially for some more complicated variants of IRSA as IRSA with power diversity or IRSA with sensing. The main question that we propose at the end of this chapter is, whether adding a sensing phase before IRSA transmission could be useful and costly affordable to synchronize the nodes and avoid collisions. In this case, can our DRL approach exploit sensing information and reach more than the optimal values we found for classical short-frame IRSA? This will be addressed in the next chapter.



(a) 10 users and 10 slots



(b) 15 users and 15 slots



(c) 25 users and 25 slots

Figure 8.6: The convergence of IRSA throughput for different number of users and slots

Chapter 9

A Deep Learning Approach to IRSA with Sensing

9.1 Introduction

One of the main roles of MAC protocols is to decide the transmission strategy of the connected nodes. Given the limited network resources and the variable environmental conditions, the design of efficient MAC protocols is one of the main challenges for current networks. The primary goal of ML techniques applied to random access is to exploit communication resources such that they provide the best possible connectivity for all demands. By definition of *random access*, randomness will inevitably, sometimes, provide situations where decoding is not possible. But we observe that if nodes had just a small amount of knowledge on how the other nodes intend to transmit, the performance could be improved. In this chapter, our main idea is to introduce some form of sensing between the nodes: active nodes not only transmit to the base station but can also sense the channel in order to get information about the presence and the activity of other transmitting nodes. This is in line with very common and popular methods from classical random access, such as Carrier Sense Multiple Access (CSMA) [127, Section 4.4], and its numerous generalizations [128]. A major question is now: how to design or adapt sensing to IRSA?

Motivated by a very recent work that exploits learning not just to learn the parameters, but to learn the entire methods of interactions [129] or even entire programs, our initial goal is to *learn to interact*, in the sense of *learning a sensing protocol* entirely through Deep Learning. Because of the high complexity of this task, in this chapter, we mostly detail initial steps towards these goals and provide a few, limited, but interesting results.

For that aim, we introduce Deep-Learning and Sensing-based IRSA (DS-IRSA), a new

variant of IRSA protocol which is based on sensing and that with optimizing through Deep Learning, with a derivative of Deep-RC-IRSA. With sensing, our transmission scheme is composed of two phases: a sensing phase, where active nodes send and attempt to sense short jamming signals with the intent of interacting with other nodes and potentially performing some (weak) form of coordination. Sensing is a generalization of carrier sense [128]. The second phase is as classical IRSA, but each node will choose an adapted transmission strategy to send its replicas, partly based on the information of the sensing phase. We use a Deep Reinforcement Learning approach based on the DRL method Proximal Policy Optimization (PPO) [125]. Our obtained results through learning show an excellent performance for short frame IRSA, compared to classical IRSA [1]. Deep-IRSA [124] and Deep-RC-IRSA (from the previous chapter), but our results are still for short and limited frame lengths.

9.2 System Model for Sensing-based IRSA (S-IRSA)

We introduce a version of IRSA with sensing (Sensing-based IRSA, S-IRSA), by extending the usual version of IRSA described in section 2.4. The main new modifications to the basic IRSA system model are listed below:

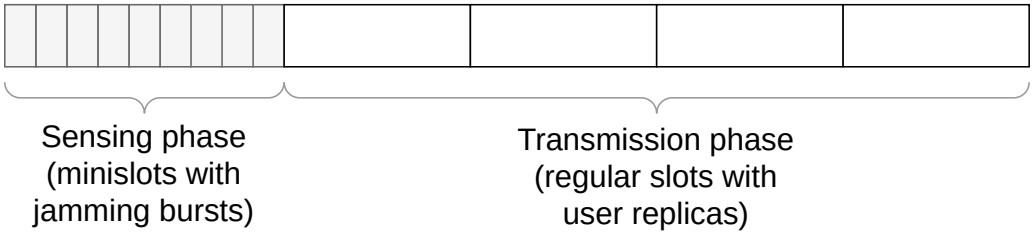


Figure 9.1: Extended frame: minislots of a sensing phase followed by regular slots of a transmission phase

- In classical IRSA and RC-IRSA, each active user decides on which slots it will transmit at the beginning of each frame: either directly with a list of slots (RC-IRSA), or indirectly by picking first a degree from a degree distribution and then picking several slots at random accordingly (classical IRSA). In our proposed approach, we extend the usual frame of IRSA by prefixing it with as phase of *minislots*, as represented in Fig. 9.1. We denote this phase as the *sensing phase*. The sensing phase is followed by the IRSA transmission phase where users will select their transmission strategies (i.e. the slots where each user will send its replicas).

- The goal of the sensing phase is that users collect information about the environment by only sensing the channel and optionally sending signals on the minislots. Then, they exploit this extra information to attempt to synchronize and avoid collisions.
- In the sensing phase, on each minislot, a user decides to be either active or stay inactive. If it is active on the minislot, it then sends a *jamming signal* for the duration of the minislot. We denote these transmissions as *jamming bursts*. In the literature, as in [128], they use the terms "black bursts", "bursts", "busy tones", ..., etc. The behavior of one user, during the entire sensing phase, can be summarized as a zero-one sequence, with one value for each minislot.
- In the sensing phase, at each minislot: each user also senses the energy of the jamming transmissions on the channel. In this work, we start with an idealized model, where 1) the users are able to exactly measure the amount of energy of the simultaneously transmitted jamming bursts, hence are able to exactly detect the number of transmitting users on the minislot, 2) the users are operating in full-duplex mode, therefore, are fully able to sense the channel while simultaneously transmitting a jamming burst, 3) they are able to complete the sensing for one minislot, before they have to decide to transmit or not a jamming burst for the next minislot. As a result of these assumptions, all nodes in the network share a common knowledge: the outcome of the sensing phase that can be summarized by an integer sequence, with one value for each minislot corresponding to the number of users simultaneously transmitting a jamming burst on this minislot.
- In the transmission phase: the nodes operate as in an RC-IRSA frame, by transmitting replicas of their data packets in the slots that they selected. The difference with (RC-)IRSA is that their slots' selection can be influenced by the collected information in the sensing phase. Note that we do not assume that the users are doing any sensing during the transmission phase itself.
- As with IRSA, the BS listens to the signals of the transmission phase and decodes the replicas of each user with SIC. We do not assume that the BS listens to the transmissions during the sensing phase.

Note that there are additional variations of the sensing system model that we are not exploring in this work:

- Half-duplex instead of full-duplex: with half-duplex, a user cannot sense the channel if they transmit a jamming burst.
- Binary sensing: the user can only detect if at least one user is transmitting on the same

minislot, but it has no information about the number of transmitting users. This is the assumption in many other works using jamming [128].

- Implicit sensing phase: instead of having an explicit sensing phase with minislots, the system could adopt a frameless version, and each user could use energy measurements on the (previous) IRSA slots.
- More precise energy model: instead of assuming that the received power is uniformly identical for all users, a path-loss model could be adopted.

9.3 Insights on Exploiting Sensing Information and Design Sensing-Based Protocols

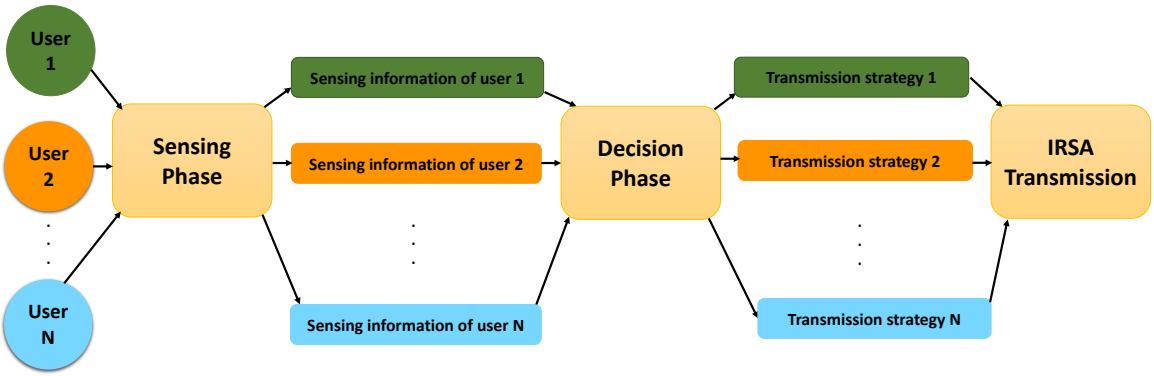


Figure 9.2: The general problem of IRSA with sensing

In this section, we discuss the general problem of introducing a sensing phase to IRSA and pose the problem of how to exploit it in the best possible way. Fig. 9.2 shows the general problem of IRSA with sensing: in the sensing phase, the users need to select their actions on each minislot (transmit or not a jamming signal); then with the knowledge of their actions and the observations of the energy on the channel (their sensing information), they have to decide a transmission strategy (the sets of slots where they will transmit replicas). Finally, the classical transmission of replicas is performed, and the BS decodes iteratively with SIC as for IRSA.

In our system model, the initial active users on a frame are actually *undifferentiated*. This is because we are in a context of mMTC communication where active nodes are essentially a small random subset of a very large set of devices and as a design choice, we assume that they have an identical behavior (i.e. not dependent on a pre-defined node identifier for instance). Active users start by interacting, sensing the channel, and collecting

	minislot 0	minislot 1	minislot 2	minislot 3
A	1	0	0	1
B	1	0	0	1
C	0	1	0	1
D	1	0	1	0
Total Sum of Energy	3	1	1	3

Figure 9.3: A simple illustrative example of the sensing phase of S-IRSA with 4 users and 4 minislots

some information. This collected information should be exploited to differentiate the users in order to enhance their transmission strategies. This could be done, by taking some decisions about, for instance, the slots where users send their replicas. The decision phase could be a specific algorithm or, as we opted, could involve a neural network model.

9.3.1 Differentiation: A Simple Illustrative Example

To illustrate how might sensing help to improve performance, Fig. 9.3 shows a simple example of the sensing phase of 4 users interacting on 4 minislots. In the example, user A is transmitting jamming bursts on minislots 0 and 3, user B as well, etc. In our model, all the users have the same information about the number of transmitters in each minislot through sensing. Indeed, the sequence of observations on the minislots $\begin{pmatrix} 3 & 1 & 1 & 3 \end{pmatrix}$ is known by all users.

An arguably interesting idea would be to take advantage of this globally known information, for instance, to help users to select some slots. We denote users who have transmitted alone a jamming signal on some minislots as *sole transmitters*[1]. This is the case for users C and D. Then observe that users C and D can fully be designated by the sentences "the sole transmitter on minislot 1" and "the sole transmitter on minislot 2" re-

[1]This is akin to the *singletons* in transmission phase with IRSA, but observe that SIC cannot be used in the sensing phase.

spectively. We also denote the consequence that they can be uniquely designated as being *fully differentiated*.

This differentiation can be exploited, for instance, by defining deterministic rules for slot selections. Indeed, imagine the following convention:

- First, all minislots with sole transmitters are considered to be fully differentiated (by all users).
- Some slots of the transmission frame will be reserved for the exclusive use of those sole transmitters: one slot for each of them. The transmission slots are reserved in the same order as the minislots, i.e. the very first slot of the transmission frame will be reserved for the first user who became a sole transmitter in the sensing frame.
- The remaining slots will be used by the remaining users, that have not been fully differentiated. Quite naturally, non-differentiated users could use classical IRSA to compete for the remaining slots in the transmitting frame.

In the example of Fig. 9.3, C is the first sole transmitter (minislot 1), D is the second one (minislot 2), hence the rule would grant them respectively the first slot and the second slot of the following transmission frame. A and B are never sole transmitters, hence they might use classical IRSA: they select a degree, and then they randomly select some slots starting from slot 3 till the end of the transmission frame. This helps to improve performance.[2]

The entire example relies on the fact that some nodes are fully differentiated on some minislots. A natural question arises, can other similar rules be established? And can users be *partly differentiated*? The sensing phase contains some information that could be the source for differentiation and could be translated into some kind of coordination between the users. The major issue is to find the best exploitation of this collected information to create a kind of coordination between users. Simultaneously, in the sensing phase, what is the best protocol for interacting and selecting minislots for transmitting jamming bursts?

The entire random access literature (or at least the RA literature with sensing) constructs protocols to achieve solutions for previous questions; however, these protocols are constructed explicitly and IRSA is not considered. A current trend is to introduce machine-learning techniques to automatically adapt protocols to different scenarios. In our case, using ML can offer one decisive advantage: one might not need to specify *which* sensing information to use, *how* to use it, and *what* is the best interaction in the sensing phase. In-

[2]It is not immediately clear that the performance will be improved, but looking at Table 8.2, we see that without sensing: with 4 users on 4 slots one expects to decode ≈ 2.06 of them, while with 2 users on 2 slots, the expectation is ≈ 1.33. Thus, on the example, applying the proposed conventions, we expect to decode ≈ 2 (differentiated users) + 1.33 (classical IRSA expected gain for 2 users competing for 2 slots) = 3.33 users, compared to ≈ 2.06 without sensing, a clear gain.

stead, one can envision automatically learn to how correctly interact and perfectly exploit sensing data. This is the ultimate goal of our work in this chapter.

In the next section, we briefly show some examples of possible definitions of *partial differentiation* to showcase the fact there are more options than just full differentiation.

9.3.2 Examples of Full Differentiation and Partial Differentiation

In this section, we first show how users can be fully differentiated. For the sake of the discussion, we assume here that the minislot phase is arbitrarily large. We observe that the sensing channel model of Section 9.2 is similar to the "collision channel with feedback" common in usual random access (RA) and contention resolution algorithms (CRA) [127, 130]: the difference is that a successfully transmitted packet in a RA protocol is here equivalent to be a sole transmitter. From this perspective, a collision is equivalent to having two or more transmitters on the same minislot. Hence, any classical slotted random access protocol can be easily transposed into a sensing phase protocol, where users transmit jamming bursts instead of packets and repeat them in case of "collisions" following the rule of a CRA.

In the case of an arbitrarily large sensing frame, each user will continue to interact until it becomes a sole transmitter. With any sensible RA, in the end, all users will be fully differentiated. Using the transmission slot allocation rule of Section 9.3.1, the transmissions in the transmission frame follow a TDMA (Time-Division Multiple Access) schedule, and IRSA is not required. This will maximize the efficiency of the transmission phase (no collision) but this will be at a cost of a very large sensing frame.

It is also possible to introduce a clear definition of the concept of *partial differentiation* in the case of specific protocols. Indeed, it is the case for some CRA algorithms, namely splitting algorithms (also called tree algorithms) [127]. Bertsekas and Gallager describe them as follows [127, Sec. 4.3.1]:

> "The first splitting algorithms were algorithms with a tree structure. When a collision occurs, say in the kth slot, all nodes not involved in the collision go into a waiting mode, and all those involved in the collision split into two subsets (e.g., by each flipping a coin). The first subset transmits in slot $k+1$, and if that slot is idle or successful, the second subset transmits in slot $k+2$. Alternatively, if another collision occurs in slot $k+L$ the first of the two subsets splits again, and the second subset waits for the resolution of that collision." [127, Sec. 4.3.1]

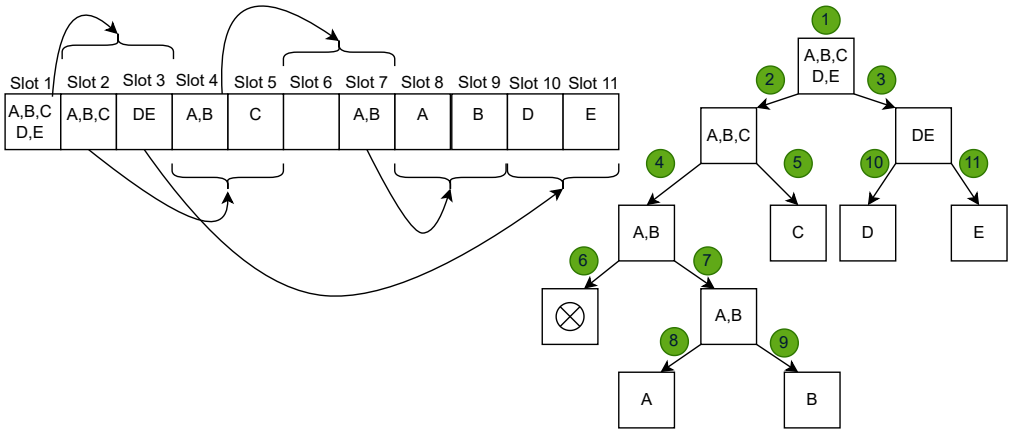


Figure 9.4: An example of a splitting algorithm with 5 users A, B, C, D, E

An example of the full operation of a splitting algorithm is represented in Fig. 9.4, with five users A, B, C, D, E, and 11 slots. As shown in the figure, all the users are being fully differentiated at the slot 11. Note that it is also possible to stop the algorithm before it ends, in other words, if the number of slots is not sufficient to differentiate all users, the algorithm stops before reaching the goal of differentiating all users.

One property of such algorithms is that the set of users that are in a collision on one slot is always included in the set of users that were in collision in one same previous slot (or they have never transmitted before). Using this property, it is then possible to designate users through *the last slot on which they have retransmitted* and partition them into disjoint sets. In Fig. 9.4, for instance, users could be put in separated sets as follows:

- After the end of slot 3, the partition is obtained from slots 2 and 3. Thus, it is: $\{A, B, C\}, \{D, E\}$.
- If we consider slot 9, users A, B and C are fully differentiated but not the two others, so the partition is then: $\{A\}$, $\{B\}$, $\{C\}$, $\{D, E\}$

Hence, at any point in time, users are always partitioned into clear subsets, so we can define clearly the notion of "partially differentiated". A fully differentiated user is a user that is only in a subset of size 1, and a *partly differentiated* user is a user that is in a subset of size > 1. In the case where all users still fall in only one set, they are still all undifferentiated.

Hence, it is possible to introduce partial differentiation through some specific sensing protocols for S-IRSA. However, we do not know if it would be the best approach to improve the performance of the transmission phase, nor how to best exploit the potential of this

partial differentiation. This is the motivation for adopting a machine learning approach to sensing protocols.

9.4 DS-IRSA: Deep Learning, Sensing-based IRSA

In this section, we present our approach to the version of IRSA with sensing, S-IRSA, that has a sensing phase followed by a transmission phase. We use a DRL approach to optimize its performance. In the next sections, we explain in detail the protocol design and how we apply the DRL approach. Furthermore, we give a simple example to clarify the concept of user differentiation.

9.4.1 DS-IRSA: Applying DRL to S-IRSA

DS-IRSA (Deep Learning, Sensing-based IRSA) is an approach that applies DRL to S-IRSA in the same way as Deep-IRSA is a DRL approach to classical IRSA. It does this by extending the DRL approach in Chapter 8, and by adding a sensing phase. A neural network model is used to select actions. Details are provided later, but an overview of the actions in each phase is as follows:

(a). The sensing phase, which consists of slots of small duration, is referred to as "minislots", where users can send jamming bursts. In DS-IRSA, we consider that each user, on each minislot, will either transmit a jamming burst or not: this decision is taken by the neural network model. Each user has also the capability to sense the channel and measure the total energy on the minislot. The information gradually collected during the sensing phase is fed online to the neural network model. The goal is that the model exploits the sensing information to introduce some differentiation between users, and synchronizes the users for the IRSA transmission phase. As discussed previously, a user could be differentiated at the sensing phase, if it has not collided with any other user, and it is left to the model, to induce some kind of differentiation and to exploit it.

(b). In the transmission phase, users have to select slots for IRSA transmission. This is done by the same neural network model that now chooses the actions of active users based on the sensing information. Each action is now a codeword of zeros and ones that specifies where the user should send its replicas on the slots. Our model represents a policy (as it outputs actions) and it is trained through a DRL Policy Gradient Method detailed in the next section.

9.4.2 Policy Gradient Methods

In this chapter, we use one of the policy gradient methods, Proximal Policy Optimization (PPO) [125] to optimize our proposed IRSA variant (DS-IRSA). In this section, we explain in detail the principles of PPO closely following its original description in [125]. Policy gradient methods [96] are a type of RL techniques that relies upon optimizing parametrized policies concerning the expected return (long-term cumulative reward) by gradient ascent. Other classical methods are action-value methods, such as Q-Learning [96, Chap. 6], that learn the values of actions and then select actions based on their estimated action values. Policy gradient methods are not action-value methods, and they learn a parameterized policy that can choose actions without a value function and are trained typically with a variant of the Policy Gradient Theorem [96, Sect. 13.2].

In PPO, a value function is still used to learn the policy parameters, but its aim is only to improve training convergence, and it is not directly involved in action selection. In the following, we further explain the principles of PPO. In PPO, the policy denoted π_θ, is a stochastic policy that takes the observed state s_t from the environment and suggests an action a_t to take as an output. It is given by a neural network model parametrized by a set of coefficients (weights) θ, that determines the probability $\pi_\theta(a_t|s_t)$ to select action a_t. During training, RL episodes are run, and the stochastic gradient ascent is used to iteratively improve the policy. This requires an estimator of the policy gradient from the sampling obtained through the episodes. The most commonly used gradient estimator of the policy gradient is derived from the Policy Gradient Theorem [96, Sect. 13.2] and has the form [125, Eq. 1]:

$$\hat{g} = \hat{E}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)\hat{A}_t\right] \tag{9.1}$$

where $\hat{A}_t$ is an estimation of the value of the *advantage* function at time step t. The definition and the value of $\hat{A}_t$ is the difference between the discounted long-term cumulative reward R_t up to time step t, and the baseline estimate $b_s(t)$, that is an estimation of the expected return from the time step t onwards. They define $\hat{A}_t$ as:

$$\hat{A}_t = R_t - b_s(t) \tag{9.2}$$

The value of $\hat{A}_t$ reflects how much better was the impact of the taken action (this is given by the cumulative reward R_t) based on the expectation of what normally should happen ($b_s(t)$), considering that we are in the state s_t.

In Trust Region Optimization (TRPO) Method [125], the log function in Eq. 9.2 is replaced with the division by the old value of the policy $\pi_{\theta_{old}}$, and by adding a KL constraint. The KL constraint is a measure of how the old policy is different from the updated policy,

and it is added to make sure that the updated policy does not move far away from the current policy, as follows:

$$\underset{\theta}{\text{maximize}} \qquad \hat{g} = \hat{E}\left[\frac{\pi_\theta\,(a_t|s_t)}{\pi_{\theta_{old}}\,(a_t|s_t)}\hat{A}_t\right] \tag{9.3}$$

$$\text{subject to} \qquad \hat{E}_t\left[KL\left[\pi_\theta(.|s_t), \pi_{\theta_{old}}(.|s_t)\right]\right] \leq \delta \tag{9.4}$$

The issue of the optimization with the TRPO method is that it adds extra overhead to the optimization process, so additional modifications are required.

Proximal Policy Optimization (PPO) [125] is an online policy gradient algorithm which optimizes the clipped surrogate objective (explained later in Eq. (9.5)). To understand the principle of PPO, first, let us introduce $r_t(\theta)$, which is simply the probability ratio between the newly updated policy output and the output of the old version of the policy:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$$

The central optimization objective behind PPO is the expectation operator of the minimum of two functions: the normal policy gradient objective $r_t(\theta)\hat{A}_t$ and a truncated version of $r_t(\theta)$ between $[1-\epsilon, 1+\epsilon]$:

$$L^{CLIP}(\theta) = \hat{E}_t\left[\min\left(r_t(\theta)\hat{A}(t), \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}(t)\right)\right] \tag{9.5}$$

The term $clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}(t)$, modifies the surrogate objective by clipping the probability ratio, which removes the incentive for moving $r_t(\theta)$ outside the interval $(1-\epsilon, 1+\epsilon)$. The minimum of the clipped and unclipped objective is taken, so the final objective is a lower bound (i.e., a pessimistic bound) on the unclipped objective. This means that the change in probability ratio that would make the objective improve is ignored (when the advantage function is positive), and it is included only when it makes the objective worse (when the advantage function is negative) (see [125], Fig. 1).

The final loss function that is used to train the neural network in PPO is as follows:

$$L_t^{CLIP+VS+S}(\theta) = \hat{E}_t\left[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S\,[\pi_\theta]\,s(t)\right] \tag{9.6}$$

where: $L_t^{CLIP}(\theta)$ is the clipped PPO objective. $c_1 L_t^{VF}(\theta)$ is used to update the baseline $b_s(t)$, which specifies how beneficial it is to be in a certain state or in other words, it computes the expected return from the current state onwards. The intuition behind combining these two terms in the same objective function is that the value estimate function shares some of its parameters with the policy function. The last term $c_2 S\,[\pi_\theta]\,s(t)$ is to guarantee that the agent

does enough exploration during the training process. c_1 and c_2 are coefficients, S denotes an entropy bonus and $L_t^{VF}(\theta)$ is s a squared-error loss of the value function $\left(V_\theta(s_t) - V_t^{targ}\right)^2$.

In this chapter, we use the DRL algorithm PPO and its implementation by OpenAI and others (stable baselines [131]). We chose this method as it has the stability and reliability of trust-region methods.

In the next section, we describe our DRL application to DS-IRSA.

9.4.3 Learning Environment and DRL Architecture

The application of DRL is done according to the following principles:

9.4.3.1 The Environment

IRSA is recast in the DRL framework, it becomes a multiagent system, where each node is an agent. The environment is the available physical resource, in this case, i.e., the channel, where time is divided into two phases, a sensing phase and a transmission phase (see Fig. 9.1), and each phase is divided to correspond to a sub-frame divided into slots or minislots. Every user interacts with the environment by taking actions and receiving rewards. The training of both phases, the sensing phase, and the transmission phase, is done simultaneously.

9.4.3.2 The Actions

The set of actions of an agent $m \in M$, is represented as A_m, and it is essentially composed of two parts:

- The sequence of actions in the sensing phase $A_m^{\text{sens.}}$ which has a length L, where L is the total number of minislots. Each value of this vector is a zero if the agent does not transmit a jamming signal, or a one if the agent transmits a jamming signal. In other words, the action $A_{m,t}^{\text{sens.}}$ of an agent m is to transmit if $A_{m,t}^{\text{sens.}} = 1$ at time t, or wait if $A_{m,t}^{\text{sens.}} = 0$.
- The action of the transmission phase, $A_m^{\text{trans.}}$, which represents the selection of the slots where the agent will transmit its replicas. In other words, the codeword selection: $s_i \stackrel{\Delta}{=} (s_i[0], s_i[1], ..., s_i[N-1])$ for $i = 0, 1, 2, ..., N-1$ and N is the total number of slots. s_i represents the choice of slots of the agent m on where it will transmit its replicas. The agent m transmits on the slot t if and only if $s_i[t] = 1$

  In our implementation, a single neural network is implementing the policy. In Fig. 9.5, we show the action selection process during one episode. During the sensing phase, for

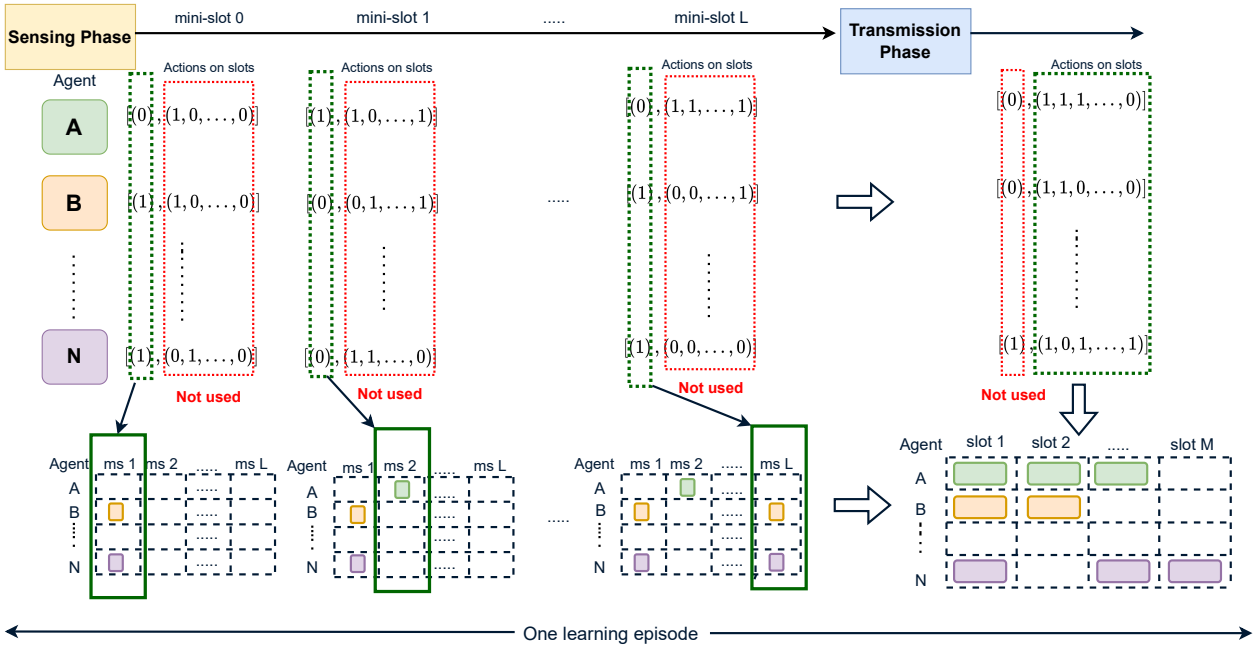


Figure 9.5: One learning episode of DS-IRSA with agent A, agent B, ..., agent N

each agent $m \in M$ and for each minislot $i \in L$, the neural network outputs an action $A^{\text{sens.}}_{m,i}$ to be executed on the minislot i. Note that the output of the model is always composed of two parts: $A^{\text{sens.}}_{m}$ and $A^{\text{trans.}}_{m}$, but only one of them is used, and the other is ignored depending on the current phase.

Fig. 9.5 shows that, during the training of the sensing phase, actions in green rectangles (the actions on the minislots $A^{\text{sens.}}_{m}$) are the only part of the action that is considered and applied on each minislot $i \in L$ separately, minislot by minislot. The second part of the action (the actions on the slots $A^{\text{trans.}}_{m}$), in red rectangles, is not used in the sensing phase. On the other hand, in the transmission phase, the second part of the action (the actions on the slots $A^{\text{trans.}}_{m}$), which represents a codeword of zeros and ones is applied on all the slots at the same time for all agents $m \in M$.

9.4.3.3 The States

The state of an agent $m \in M$ is a combination of some observed values and with additional random input values acting as entropy sources:

- The first component of the state is the observed energy on the minislots by the agent, which is represented by a vector of length L of integer values between 0 and M. Recall that L is the maximum number of minislots and M is the total number of agents.
- The second component of the state is the number (index) of the current minislot.

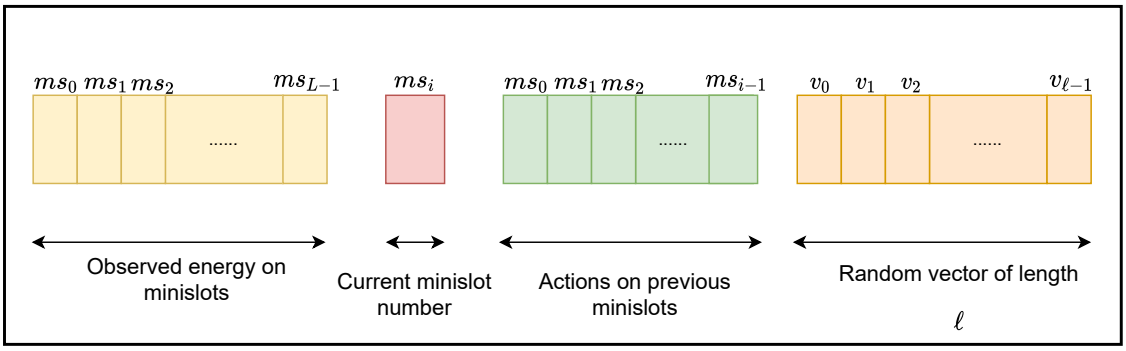


Figure 9.6: State computation of DS-IRSA

- The third component of the state is the actions that were taken by the user for previous minislots.
- The fourth component of the state is a random vector of a fixed length ℓ. In our simulations, we choose a vector of length $\ell = 10$ that has random values between [0, 15]. It acts as an entropy source to help the model output different codewords in the same state[3]. In Fig 9.6, we show how the state is constructed.

9.4.3.4 The Reward

The reward: Let R_m be the reward that the agent $m \in M$ obtains at the last time slot M. The reward depends on the action of the agent m, $A_m(t)$ and other agents' actions $A_{m'}(t)$. The reward for the agent m is defined as:

$$R_m = P_N \tag{9.7}$$

where P_N is the number of decoded packets at the end of the episode, after the slot N and the iterative decoding at the BS, where N is the total number of slots in the frame. The discount factor is set to 1. Fig. 9.7 shows the reward computation time. The reward is computed at the end of the frame. Before the end of the frame, the reward is set to 0.

9.4.4 An Example of Differentiating the Users Using DS-IRSA

Fig. 9.8 shows three possible scenarios of DS-IRSA of 3 users, 3 slots and 3 minislots.

Fig. 9.8.a shows the failure of fully differentiating any of the three competing users, as none of these three users has succeeded to send a jamming burst in the sensing phase, without being colliding with the two others. In this case, we might get only little (if any) extra

[3]observe that otherwise, with 0 minislots for instance, the model would output the same policy for all users, which cannot be optimal. Using $\ell = 10$ and values in $\{0, 1, \ldots 15\}$ is more than a sufficient number of entropy sources so that this is no longer limiting the performance.

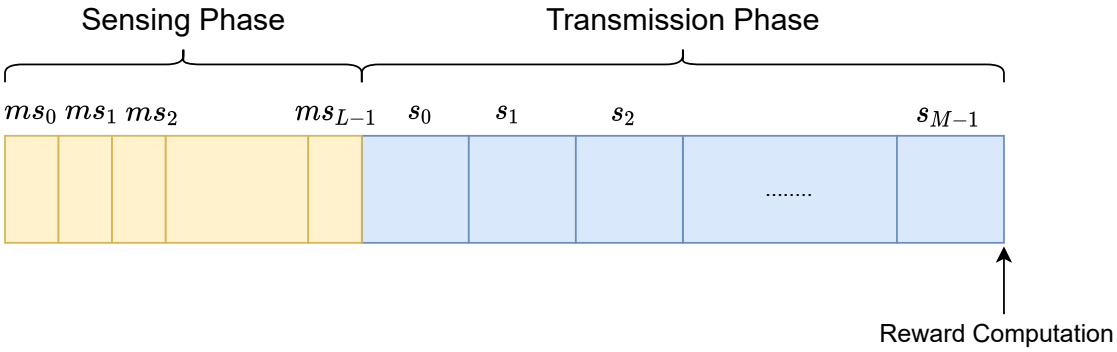


Figure 9.7: Reward computation for DS-IRSA

information from the sensing phase and the transmission phase, which can have close (or identical, it is an open question) performance to the one of a classical IRSA transmission. Another extreme case is shown in Fig. 9.8.b, where the three users have succeeded to be fully differentiated in the sensing phase. In this case, a slot could be reserved for each of the three users (see also Section 9.3.2). Note that as we employ SIC, more replicas could be sent, as Fig. 9.8.b shows in the transmission phase. Slot 2 has been reserved for user *C*, slot 1 is also reserved for user *B*. Note that user *C* can send other replicas (on slot 1 and slot 2) as its packet is assumed to be correctly removed from its reserved slot 3. It is also the case of user *B*, who sends twice on slot 2 and on slot 1.

The third final illustrated case in Fig. 9.8.c shows an intermediate case where one user (user *A*) is fully differentiated in the sensing phase and the two other users (users *B* and *C*) are not differentiated since all their actions are identical. In this case, the sampling strategy outlined in Section 9.3.2 prioritizes the differentiated user *A* by reserving him the slot 1. Users *B* and *C* have no choice than competing to send on slots 2 and 3 as in the classical IRSA scenario, and they cannot avoid the risk of collision with each other.

9.5 A Mathematical Analysis of S-IRSA with 2 Users and 2 Slots

In this section, we provide a mathematical analysis of the performance of the case of S-IRSA with 2 users and 2 slots and an arbitrary number of minislots. It will serve later as a benchmark for DS-IRSA; the same reasoning could be applied for S-IRSA with more slots but still 2 users. We do not have mathematical results for the cases with three users or more (which we believe to be significantly more complex).

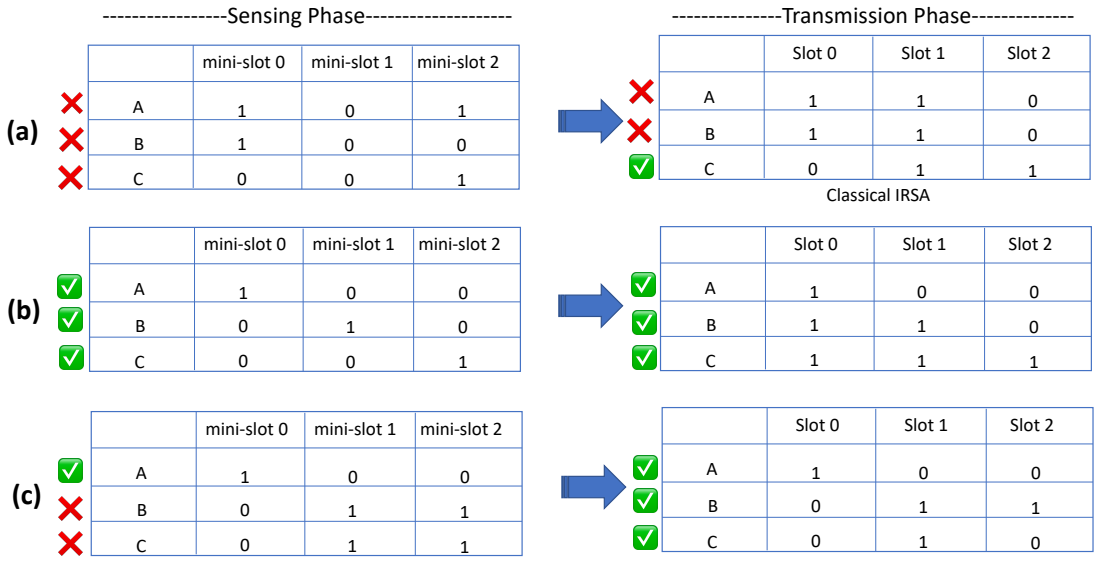


Figure 9.8: An example of DS-IRSA: (a). none of the users have been identified during the sensing phase. (b). all users have been identified during the sensing phase. (c). two out of three users have been identified during the sensing phase.

9.5.1 A Sensing Phase with One Minislot

We consider S-IRSA with 2 users competing for 2 slots with a sensing phase of only one minislot. Adding one minislot in the sensing phase changes the problem as follows: assume that π_0 is the probability to send 0 in the minislot (i.e. to be inactive). The users could be differentiated, if they send different bits[4] on the minislot, $(0,1)$ or $(1,0)$, otherwise, they could be decoded with a probability $P_s = \frac{2}{3}$ as in the case without sensing (see Section 8.3.1 and Eq. (8.3)). Thus, we can write the new success probability as follows:

$$P_s = \frac{2}{3}\left[\pi_0^2 + (1-\pi_0)^2\right] + 2\pi_0 \cdot (1-\pi_0) \tag{9.8}$$

Following the same approach in Eq. 8.1, gives an optimal value of $\pi_0 = \frac{1}{2}$, which in turn increases the probability of success to $P_s = \frac{5}{6}$. Note that, in this chapter, as in the previous chapter, we express the throughput as "the average number of decoded users" instead of the definition that we used before. In other terms, we express it in "decoded users/frame" instead of "decoded users/slot". The throughput is obtained from P_s and is:

$$T = N \cdot P_s = 2 \cdot \frac{5}{6} \approx 1.67 \quad \text{decoded users/frame}$$

[4]Note that the nodes send jamming signals, but we use the term "bits" for the ease of explanation

Note that in this specific case, adding a sensing phase of one bit helps to increase the throughput by 23.8% compared to the case without sensing in Section 8.3.1.

9.5.2 A Sensing Phase with Two Minislots

In this section, we add another minislot to the sensing phase described in the previous section. We also observe an important property for the case of two users: although each user should decide minislot by minislot their action on the next minislot, there are only two possible outcomes: either both users made the same choice of action, or they made a different choice of actions. In the first case, the minislot cannot bring any differentiation, and the users can just ignore what happened on the minislot. In the last case, one user is fully differentiated and as a consequence, the other one as well: then it does no longer matter what are the next actions of the users – and conversely, even if the users later ignore what happened on the minislot, they still would be fully differentiated.

In both cases, both users can ignore the feedback from the previous minislot(s) and still implement an optimal strategy. Thus, they can select their minislot transmission strategies at the beginning of the minislot phase: in this case, we can denote their jamming burst transmission strategy for the minislot phase as a "minislot codeword".

The two users will have $2^{\text{minislots}} = 2^2$ possible minislot codewords to use in the sensing phase. The users will not be differentiated if their activity in the sensing phase is represented by the same minislot codeword, i.e., $(00, 00), (01, 01), (10, 10)$ or $(11, 11)$. There are $16 - 4 = 12$ possible combinations of minislot codewords that allow both users to be differentiated. Suppose that: $\pi_{00}, \pi_{01}, \pi_{10}, \pi_{11}$ are the probabilities of using the minislot codewords: $00, 01, 10, 11$ respectively. The success probability is obtained by considering that if both users are fully differentiated it will be 1, and if they are not, the minislot had been useless and the success probability will be given by Eq. (8.3), i.e. $\frac{2}{3}$. Hence, it is given by the following equation:

$$P_s = 1 - \left(\pi_{00}^2 + \pi_{01}^2 + \pi_{10}^2 + \pi_{11}^2\right) + \frac{2}{3}(\pi_{00}^2 + \pi_{01}^2 + \pi_{10}^2 + \pi_{11}^2) \tag{9.9}$$

By using the method of Lagrange multipliers, we can find the probability to use each

minislot codeword such that we maximize the success probability:

$$\begin{aligned}
&\mathcal{L} = 1 - \left(\pi_{00}^2 + \pi_{01}^2 + \pi_{10}^2 + \pi_{11}^2\right) + \frac{2}{3}\left(\pi_{00}^2 + \pi_{01}^2 + \pi_{10}^2 + \pi_{11}^2\right) - \lambda\left(\pi_{00} + \pi_{01} + \pi_{10} + \pi_{11} - 1\right) \\
&\frac{\partial \mathcal{L}}{\partial \pi_{00}} = -2\pi_{00} + \frac{4}{3}\pi_{00} - \lambda = 0 \\
&\frac{\partial \mathcal{L}}{\partial \pi_{01}} = -2\pi_{01} + \frac{4}{3}\pi_{01} - \lambda = 0 \\
&\frac{\partial \mathcal{L}}{\partial \pi_{10}} = -2\pi_{10} + \frac{4}{3}\pi_{10} - \lambda = 0 \\
&\frac{\partial \mathcal{L}}{\partial \pi_{11}} = -2\pi_{11} + \frac{4}{3}\pi_{11} - \lambda = 0 \\
&\frac{\partial \mathcal{L}}{\partial \lambda} = \pi_{00} + \pi_{10} + \pi_{01} + \pi_{11} - 1 = 0
\end{aligned} \tag{9.10}$$

we get the following solution: $\left\{\pi_{00} = \pi_{01} = \pi_{10} = \pi_{11} = \frac{1}{4}\right\}$. This solution gives a probability of success $P_s = \frac{11}{12}$, and the throughput becomes:

$$T = N \cdot P_s = 2 \cdot \frac{11}{12} \approx 1.83 \quad \text{decoded users/frame}$$

which has increased by 9.6%, compared to the throughput that we obtained in case of one minislot in Sec. 9.5.1.

9.5.3 Generalization to a Sensing Phase with k Minislots

Let k be the number of minislots in the sensing phase. The number of possible minislot codewords in the sensing phase is $\omega = 2^k$ possible minislot codewords. We will compute the optimal probability to use each minislot codeword such that the probability to fully identify and then to decode both users is maximized.

Let us write the success probability for such a system:

$$P_s = 1 \cdot \left[1 - \left(\pi_0^2 + \pi_1^2 + \pi_2^2 + ... + \pi_{\omega-1}^2\right)\right] + \frac{2}{3} \cdot \sum_{i=0}^{\omega-1} \pi_i^2 \tag{9.11}$$

By writing the Lagrangian and solving the associated equations:

$$\begin{aligned}
&\mathcal{L} = 1 \cdot \left[1 - \left(\pi_0^2 + \pi_1^2 + \pi_2^2 + ... + \pi_{\omega-1}^2\right)\right] + \frac{2}{3} \cdot \sum_{i=0}^{\omega-1} \pi_i^2 - \lambda \left(\sum_{i=0}^{\omega-1} \pi_i - 1\right) \\
&\frac{\partial \mathcal{L}}{\partial \pi_0} = -2\pi_0 + \frac{4}{3}\pi_0 - \lambda = 0 \\
&\frac{\partial \mathcal{L}}{\partial \pi_1} = -2\pi_1 + \frac{4}{3}\pi_1 - \lambda = 0 \\
&\dots \\
&\frac{\partial \mathcal{L}}{\partial \pi_{\omega-1}} = -2\pi_{\omega-1} + \frac{4}{3}\pi_{\omega-1} - \lambda = 0 \\
&\frac{\partial \mathcal{L}}{\partial \lambda} = \left(\sum_{i=0}^{\omega-1} \pi_i - 1\right) = 0
\end{aligned} \tag{9.12}$$

From the first ω equations, we deduce: $\pi_0 = \pi_1 = ... = \pi_{\omega-1} = \frac{-3\lambda}{2}$. Using the last equation (the $\omega + 1$ equation), we compute $\lambda = \frac{-2}{3\omega}$ and this gives:

$$\pi_0 = \pi_1 = \pi_2 = ... = \pi_{\omega-1} = \frac{1}{\omega}$$

The final success rate is given by :

$$P_s = 1 \cdot \left[1 - (\sum_{0}^{\omega-1} \frac{1}{\omega^2})\right] + \frac{2}{3} \sum_{0}^{\omega-1} \frac{1}{\omega^2} = 1 - \frac{1}{3} \sum_{0}^{\omega-1} \frac{1}{\omega^2} = 1 - \frac{1}{3\omega} = 1 - \frac{1}{3.2^k} \tag{9.13}$$

We deduce that adding a sensing phase before transmitting the packets in the scenario of IRSA with 2 users and 2 slots leads to a significant improvement in the probability to decode both users (the success probability), and the obtained throughput. The important question that we answer in the next section: can we extend the same analysis to a scenario of IRSA with M users and N slots?

9.6 Numerical Results

In this section, we present the obtained numerical results with DRL using our simulator. We developed our own IRSA simulator written in Python. The simulations allow constructing frames and generating user transmissions. IRSA iterative decoding is performed using a collision model after which the decoding information is obtained (decoded users, throughput, etc.). For DRL, we used OpenAI stable baselines. The neural network model implementing the policy (and the baseline value function estimate) is used with the default parameters: it contains 2 layers of 64 neurons amounting to approximately 4000 weights.

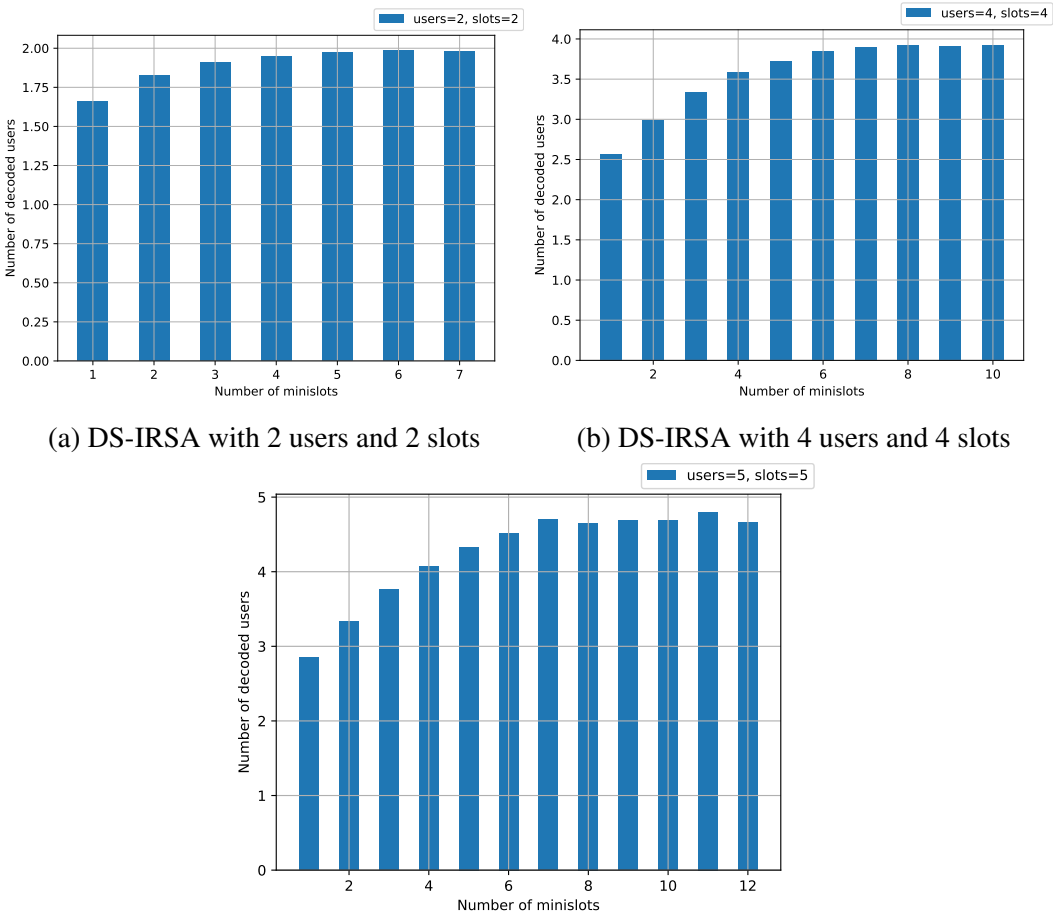
u/s	2	3	4	5	6
2	1.982	1.986	1.986	1.990	1.994
3	1.971	2.996	2.997	2.979	2.969
4	1.967	2.73	3.84	3.73	3.77
5	1.963	2.80	3.463	4.168	4.233
6	1.937	2.76	3.30	3.152	3.873

Table 9.1: The obtained throughput (expressed in terms of "average number of decoded users per frame") of DS-IRSA with 7 minislots using our DRL approach

The arrival of users in the system is fixed to M users per frame. Training is done in steps. Each step accounts for an action taken by one user (agent). A full episode is completed when all the agents take their final actions, which is the slot selection for the transmission phase, after which the reward is calculated (the reward is zero at initialization and before the end of the frame and discounting factor is set to $\gamma = 1$).

In Table. 9.1, we show the obtained throughput after applying our DRL approach DS-IRSA. The two phases were trained simultaneously. Each value in the table represents a different scenario of DS-IRSA where we varied the number of users and the number of slots. We obtain each value in Table. 9.1, by doing the average of 10 simultaneous simulations (training), after recording the number of decoding users at each simulation. All the presented scenarios in the table have 7 minislots in the sensing phase. We observe that DS-IRSA with 7 minislots is close to the maximum throughput of 2 recovered users in the case of 2 competing users (the first row of the table). DS-IRSA attains also almost the maximum throughput of 3 in the case of three users competing for ≥ 3 slots. We also note that adding a sensing phase of 7 minislots to IRSA has helped to increase the obtained throughput compared with the obtained values of throughput without sensing in Table 8.2 for all the studied scenarios. On the other hand, with sensing and when the number of users is ≥ 4, the throughput starts to converge slowly, and we do not recover all users. The reason is that the number of minislots that we used (7 minislots) is probably not sufficient to synchronize all users during the transmission phase, so more collisions occur, and not all users are recovered.

Fig. 9.9 shows the impact of increasing the number of minislots on the obtained throughput. Note that the number of episodes in the training phase for all the scenarios of Fig. 9.9 is one million for the Fig. 9.9a and 5 millions episodes for Fig. 9.9b and Fig. 9.9c. The figure shows that using more minislots in the sensing phase helps to increase the obtained



(a) DS-IRSA with 2 users and 2 slots

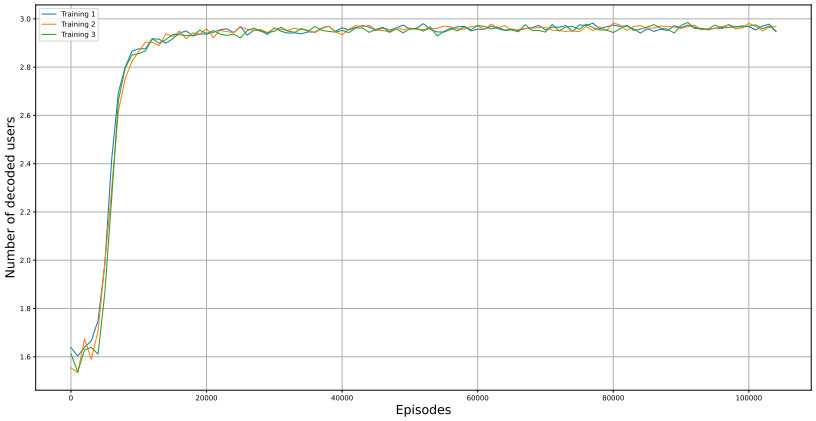
(b) DS-IRSA with 4 users and 4 slots

(c) DS-IRSA with 5 users and 5 slots

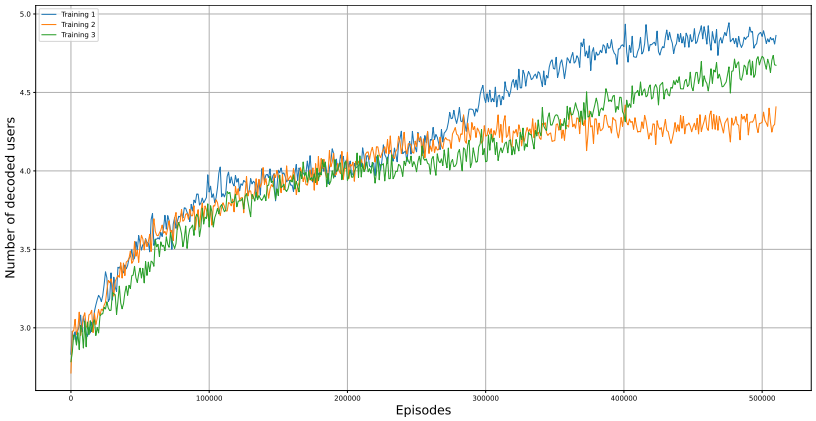
Figure 9.9: The impact of increasing the number of minislots in the sensing phase on the obtained throughput of DS-IRSA

throughput towards achieving the maximum performance of 1 [decoded user/slot]. In Section 9.5.3, it has been theoretically proven that increasing the number of minislots in the sensing phase increases the achieved throughput for DS-IRSA with 2 users, 2 slots, and k minislots which converges quickly towards the maximum of 2. The DRL approach achieves a throughput of 1.98 for DS-IRSA with 2 users, 2 slots and 7 minislots in the sensing phase (Fig. 9.9.a). When studying a scenario of DS-IRSA with strictly more than two slots and users, the DRL shows that using more minislots in the sensing phase will increase the achieved throughput, but this increase towards the optimal throughput is not always clearly observed as the convergence of the DRL approach is not guaranteed when the number of slots and the number of users increase. In this case, extra minislots are needed when the number of competing users increases, and this, in turn, increases the actions' space and can increase exponentially the complexity of the training.

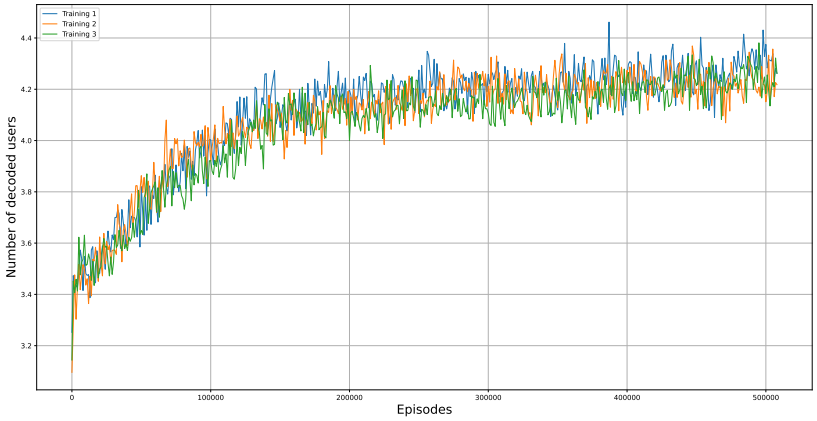
In Fig. 9.9.b, using 7 minislots in the sensing phase, DS-IRSA achieves a throughput of



(a) DS-IRSA convergence with 3 users and 3 slots and 7 minislots



(b) DS-IRSA convergence with 6 users and 6 slots and 7 minislots



(c) DS-IRSA convergence with 7 users and 7 slots and 7 minislots

Figure 9.10: The learning convergence of DS-IRSA for different scenarios

3.84 (average number of decoded users per frame) where using 10 minislots is increasing the throughput up to 3.91. On the other hand, the obtained throughput for 5 users using 9 minislots, in Fig. 9.9.c, is 4.68, which is still far from the maximum throughput. As the number of users increase, the number of minislots needed to synchronize the users increases, which increases the training complexity in terms of time and convergence stability.

Fig. 9.10 shows the convergence of the learning process of DS-IRSA for different scenarios. For each scenario, we performed three different trainings with different seeds. Note that the training starts by randomly initialized values for the weights of the neural network model, and each value is different depending on the used seed. In Fig. 9.10a, the maximum throughput of 3 is quickly reached, as the number of minislots is sufficient to find an optimized transmission strategy for the 3 competing nodes. On the contrary, in Fig 9.10b and Fig 9.10c, the learning convergence towards the maximum throughput is slower as the number of competing users is larger in both cases. Note that in both figures, Fig 9.10b and Fig 9.10c, the number of minislots and the number of learning episodes was not sufficient to converge towards the maximum throughput of 6 and 7 users per slot respectively. In the last two examples, the performance with 7 users and slots leads to a lower absolute average number of decoded users par frame than with 6 users and slots, which is somewhat unexpected and surely due to convergence issues. Additionally, checking the number of minislots proportional to the number of slots and users, we also have the throughput of only ≈ 3.4 for 6 users, 6 slots, and 14 minislots, compared to Fig. 9.9a. Thus, experimentally, with the current implementation and training procedure, adding more minislots to the sensing phase for both scenarios will not guarantee convergence, as the problem of finding the optimal codebook becomes more complicated. In the last two scenarios (Fig 9.10b and Fig 9.10c), the learning curves seem to continue their increase slowly even at the end of the episodes. The continuation of the training process could help to increase the obtained throughput, but at the cost of a long learning time.

9.7 Conclusion

In this chapter, we proposed a sensing protocol based on IRSA and trained with machine learning to synchronize the nodes during the transmission and avoid collisions. For that aim, we proposed DS-IRSA, Deep Learning Sensing-based IRSA protocol which is composed of two phases: a sensing phase, where the nodes can sense the channel and send short jamming signals, followed by a classical IRSA transmission phase. We use the DRL algorithm PPO and one of its implementations (by OpenAI and others, "stable baselines" [131]) to optimize its performance. Our proposed protocol has shown an excellent performance to achieve an optimal performance of almost 1 [*decoded user*/*slot*] for small frame sizes (≤ 5) slots and with enough minislots. Our proposed protocol has also been shown to achieve higher throughput than classical IRSA protocol or other optimized IRSA variants through deep learning. The problem of optimizing DS-IRSA with our DRL approach be-

comes more complicated in terms of stability and learning time when the size of the frame increases, e.g. beyond 6 or 7 users and slots for 1 million training episodes. Future work could be to optimize the learning approach to reduce the learning complexity for larger frame sizes.

Chapter 10

Conclusion and Future Perspectives

IoT is exploding, which leads to unprecedented connectivity, which requires and opens the door to new connection technologies. One main hurdle faced by the application of IoT networks is the issue of massive connectivity. Having a scalable IoT network that connects millions of devices and servers is critical for a large-scale IoT application. IoT connectivity should be aforethought before deployment, not an afterthought, therefore, many endeavors have been emerged to address this massive connectivity problem. One of the promising suggested solutions is to employ *modern random access* protocols for IoT communication, in particular, because they have been shown to offer excellent performance, i.e. to reach the bound of 1 $[packet/slot]$ [3].

Motivated by the important expected promise of *modern random access* protocols in future IoT applications, in this thesis, we focused on one of these protocols, Irregular Repetition Slotted Aloha (IRSA) which has been introduced in [1]. First, we extensively evaluated the performance of the IRSA protocol by studying one of the best tools to track its decoding performance: *Density Evolution* (DE). With this tool, we studied multiple scenarios of IRSA protocol and proposed several variants. Second, we applied different machine learning techniques to study variants of IRSA.

10.1 Main Contributions

Multiple Packet Reception (MPR), in which the receiver can recover multiple packets from a limited number of transmissions, had been investigated for throughput improvement in [2, 23, 24]. In Chapter 3, we revisited the scenario of IRSA with K-MPR capability at the receiver (K-IRSA). We exploited the density evolution equations for K-IRSA, and we presented an upper bound on the asymptotic performance of K-IRSA. It has been proven

that this bound cannot be reached naturally, owing to the existence of a gap between a target function $h_K(x)$ to match (to achieve the maximal capacity of k packet/slots) and any edge degree distribution $\lambda(x)$.

Capture effects have been exploited in the study of IRSA with multiple classes received with different signal power in Chapter 4. In such a scenario, the existence of different path loss factors or different transmission powers helps the capture effect to emerge at the receiver, which in turn improves the protocol performance. We analyzed the protocol behavior by introducing a new density evolution that is based on classes. We explored the achievable throughput and its associated gain.

Another contribution of this thesis, presented in Chapter 5, is to study IRSA taking into account realistic assumptions, namely errors due to an imperfect SIC process. A new parameter is introduced to specify the minimum target packet loss rate of the system, which is non-zero due to potential IC errors. We investigated the trade-off between the maximal throughput and the minimum target packet loss rate. The obtained results shed light on new optimal distributions designed and optimized taking into account the probability of SIC errors that outperform the known classical ones that are optimized without considering SIC errors.

Additionally, in Chapter 6, we proposed a new variant of IRSA, namely "IRSA-RM", which adopts a Reinforcement Learning (RL) approach, Regret Minimization, to optimize the transmission strategy of the protocol. Our simulation results show a good behavior of IRSA when it is optimized with Regret Minimization. Therefore, our learning approach is a good solution for optimizing the performance of such IoT protocols. This study was done using the density evolution tool, and IRSA-RM has been shown to help to approach the asymptotic performance under the assumptions of density evolution, i.e. in case of ideal conditions: infinite frame length and assuming an ideal decoder (no SIC errors).

We also addressed the IRSA access scheme in a distributed setting, where there is not a unique set of parameters for all users, in Chapter 7. The distributed approach is modeled as a non-cooperative game where the players are classes of users that can select their degree probabilities to improve their own throughput. We gave proof for the existence of the Nash Equilibria under some conditions and how to attain them. We provided extensive numerical results that assess the cost brought by the competition with this distributed approach. We also compared the centralized optimal approach, and the distributed approach through the price of anarchy, and found a very low discrepancy.

In Chapter 8, we introduced a Deep Reinforcement Learning (DRL) approach to optimize IRSA for short frame length. We presented RC-IRSA, an IRSA approach with random

codeword selection, where each codeword represents the transmission strategy of one user on the slots. The main goal of this chapter was to apply Deep Reinforcement Learning to optimize RC-IRSA (Deep-RC-IRSA): it resulted in the training of a Deep Neural Network model that selects exactly the slots on which the user will transmit. It illustrates that the fine slot selection can be achieved through DRL, illustrated some of its limits, and was also introduced to use it as a framework for the following chapter.

Finally, in Chapter 9, we introduced Deep-Learning and Sensing-based IRSA (DS-IRSA), a new variant of IRSA protocol that is based on sensing and that is optimized through Deep Learning, with a derivative of Deep-RC-IRSA. DS-IRSA consists of two phases: the first phase is a sensing phase, where users sense and send jamming signals to interact and potentially synchronize to avoid collisions. The second phase is as classical IRSA transmission, but each node will choose an adapted transmission strategy (the precise set of slots) to send its replicas, partly based on the collected information of the sensing phase. We used a Deep Reinforcement Learning approach based on the DRL method Proximal Policy Optimization (PPO) to optimize DS-IRSA. Our results showed the gain of having a sensing phase compared to RC-IRSA and classical IRSA, but they are still limited to very short frame sizes.

10.2 Perspectives

This section presents some future research directions and perspectives related to some key aspects of modern random access protocols, namely, IRSA, that were not addressed in this thesis and some aspects that require further investigations and improvements.

10.2.1 A Better Employment of the Physical Layer

Modern random access protocols have been showing a promising performance that could provide good solutions for the massive connectivity problem. With such methods that can function in high load regimes and resolve packets with high collisions, channel estimation for the SIC process, is a necessary task, but it is not an evident task. Better channel estimation could help to reduce the impact of residual channel estimation errors on the performance of interference cancellation [132] and thus, have a realistic representation of the protocol performance which in turn helps to enhance the decoding process and to avoid errors. Thus, a perspective of this work is considered more realistic channels, and study how to improve the SIC process, in the context of IRSA and the variants proposed here.

More than that, some advanced physical layer techniques could be applied, for instance, to extract information from collided packets. MuSCA [133] is one example of a generalization of CRDSA, where each user transmits several parts of a single codeword of an error-correcting code instead of sending replicas. At the receiver level, the decoder collects all these parts and includes them in the decoding process, even if they have interfered. MARSALA [132] is another decoding technique for CRDSA. At the receiver side, MARSALA takes advantage of correlation procedures to localize the replicas of a given packet, then combines the replicas to obtain a better Signal to Noise plus Interference Ratio. Applying such techniques on IRSA could positively affect its performance as more collisions could be resolved, especially collisions that occur in a form of stopping sets.

10.2.2 NOMA-based Modern Random Access

Another direction to improve the performance of modern random access protocols is to combine them with NOMA or study them in the context of NOMA. To apply NOMA in modern random access, power division multiple access (PDMA) [134] was proposed. In PDMA, the time division duplex (TDD) channel is assumed. That is to say, IoT devices need to obtain the uplink channel gain via downlink channel sensing. Then, each user adjusts the transmitted power to force the receiving power to belong to some predetermined power slots [135]. Recently, NOMA has been used in multiple slotted ALOHA (SA) methods [42, 136]. PDMA has been also combined with IRSA in [43]. The optimizing of these protocols, combined with NOMA, requires finding the optimal combination of degree and power distributions while taking into account sophisticated physical layers. Despite that NOMA-based modern random access has been shown to improve the performance via power diversity, many further research works can be done, including the fair comparison with orthogonal random access methods in different scenarios, advanced blind detection technologies, and non-orthogonal spatial domain [135]. Applying learning approaches to NOMA-based modern random access protocols could be also useful to optimize them if only to find more robust power and degree distributions.

10.2.3 Integrate Advanced Learning Techniques with Modern Random Access

Recently, machine learning has been widely applied to modern random-access [79, 80]. However, the search for protocols and algorithms that further reduce the resource allocation delay and optimize the performance of such protocols remains an area of active

research. Modern learning techniques, specifically deep learning, could be further applied to modern random access on the PHY layer or the MAC layer level. Machine learning techniques could help to find the robust performance of modern random access protocols when more realistic assumptions are considered as capture effect, decoding errors, or losing some packets due to a realistic collision channel. For example, machine learning algorithms could be employed to search correlations between received packets to reduce and resolve collisions, to control the transmission power of packets to exploit power diversity at the receiver as in [137], or to deeply exploit the collected information by the nodes in case of sensing scenarios. An important future work to our contribution in chapter 9, could be to use an advanced deep learning algorithm to extend the work for more realistic scenarios that consider a high, but a finite number of users or to change the slot selection process. More than that, learning could be exploited to provide some bounds or optimal codebooks for IRSA, if we change the problem dimensions (number of users, number of slots, number of minislots, ..., etc.). In addition, the sensing process itself could be further optimized and the transmission of the jamming signals could be done according to a predefined protocol, which can reduce the number of used minislots and strongly synchronize the users in the transmission phase. Sensing controlled through machine learning could also be studied in classical random access protocols, for instance for some V2X modes.

In recent years, part of the research went beyond the DQN-style algorithms, with algorithms employing policy gradients, such the Proximal Policy Optimization algorithm [125], which are more stable and intuitive to analyze theoretically. The long time and the huge data needed to train the neural networks has led to new research directions, including hybrid model-based/model-free solutions or model-based RL algorithms. Although learning a model of the environment may seem complex, incorporating specific details about the physical properties of components of the studied environment can speed up the learning process. Constructing model-based algorithms for some random access RA applications could be useful to fully understand the behavior of modern random access protocols. More generally, more complex methods of training and more sophisticated uses of the various types of neural network models, can be envisioned to both speeds up the training of the models, and improve the resulting performance of the random access.

10.2.4 General Perspectives

Recently, the Age of Information (AoI) is gaining more attention in many IoT applications. It is a metric used to evaluate the latency from receivers' perspectives. It measures the time elapsed between generating a packet at the transmitter and the time it reaches the

receiver. [138] and [139] studied some aspects of AoI of IRSA, as it is considered as a good metric to understand the link-layer solutions employed in IoT systems. According to our knowledge, there are no bounds on the AoI for *modern random access* protocols. In this context, the behavior of AoI provides insights about the activity of each active node in the system that is interrelated with those offered by other wide-studied metrics as the throughput or the latency. Hence, important future work is to study the impact of AoI in the context of *modern random access* protocols. Despite that, some important work has been started in [138] and [139], but more effort is needed to address the AoI for IRSA variants or frameless IRSA and design IRSA-style protocols that improve AoI.

Another important direction to improve the performance of *modern random access* protocols is to study the possibility of repeating the packet in case of packet loss (due to a failure of IRSA decoding). The users who fail to send their packets successfully in a frame may try to re-send their packets in the following frame in a randomized manner, which may increase the channel load in the next frame and create more collisions. Controlling the re-transmission process can have a positive impact on the system throughput compared to a blind retransmission mechanism. In our prior work [124], we studied short frame length IRSA that allows packet re-transmissions in case of collision through a DRL model. This work is considered as a first step to study a scenario of IRSA with re-transmissions.

Finally, we have studied IRSA as a generic random access protocol: it can be interesting to study how it can be integrated in actual communications standards (5G and beyond including 5G NR-Light, etc.).

Bibliography

[1] G. Liva, "Graph-Based Analysis and Optimization of Contention Resolution Diversity Slotted ALOHA," *IEEE Transactions on Communications*, vol. 59, no. 2, pp. 477–487, 2011.

[2] C. Stefanović, E. Paolini, and G. Liva, "Asymptotic Performance of Coded Slotted ALOHA With Multipacket Reception," *IEEE Communications Letters*, vol. 22, no. 1, pp. 105–108, Jan 2018.

[3] F. Clazzer, A. Munari, G. Liva, F. Lazaro, C. Stefanović, and P. Popovski, "From 5G to 6G: Has the Time for Modern Random Access Come?" 03 2019.

[4] M. Berioli, G. Cocco, G. Liva, A. Munari *et al.*, "Modern Random Access Protocols," *Foundations and Trends® in Networking*, vol. 10, no. 4, pp. 317–446, 2016.

[5] E. Paolini, G. Liva, and M. Chiani, "Graph-Based Random Access for the Collision Channel without Feedback: Capacity Bound," in *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*, 2011, pp. 1–5.

[6] K. R. Narayanan and H. D. Pfister, "Iterative Collision Resolution for Slotted ALOHA: An Optimal Uncoordinated Transmission Policy," in *Turbo Codes and Iterative Information Processing (ISTC), 7th International Symposium on*, 2012.

[7] R. Hassan, F. Qamar, M. Hasan, A. Hafizah, A. Aman, and A. S. A. Mohamed Sid Ahmed, "Internet of Things and Its Applications: A Comprehensive Survey," *Symmetry*, vol. 12, October, 2020.

[8] S. Grant, "3GPP Low Power Wide Area Technologies - GSMA White Paper," *gsma.com. GSMA.*, p. 49, Retrieved October 17, 2016.

[9] A. Khalili, A.-H. Soliman, M. Asaduzzaman, and A. Griffiths, "Wi-Fi Sensing: Applications and Challenges," *The Journal of Engineering*, vol. 2020, no. 3, pp. 87–97, 2019.

[10] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Next Century Challenges: Mobile Networking for "Smart Dust"," in *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*. New York, NY, USA: Association for Computing Machinery, 1999. [Online]. Available: https://doi.org/10.1145/313451.313558

[11] "Statista Research Department," https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/, 2016, [Online; accessed Nov 27, 2016].

[12] ITU-R, "IMT Vision - Framework and Overall Objectives of the Future Development of IMT for 2020 and Beyond," Sep 2015, geneva, Switzerland, ITU Recommendation M.2083-0.

[13] M. B. Shahab, R. Abbas, M. Shirvanimoghaddam, and S. J. Johnson, "Grant-Free Non-Orthogonal Multiple Access for IoT: A Survey," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 1805–1838, 2020.

[14] 3GPP, "Study on Scenarios and Requirements for Next Generation Access Technologies," *Technical Specification Group Radio Access Network, Technical Report 38.913*, 2016.

[15] Ericsson, "Cellular Networks for Massive IoT-enabling Low Power Wide Area Applications, Ericsson, Stockholm, Sweden Technical Report uen 284 23-3278," Jan. 2016.

[16] J. Ding, M. Nemati, C. Ranaweera, and J. Choi, "IoT Connectivity Technologies and Applications: A Survey," *IEEE Access*, vol. 8, pp. 67 646–67 673, 2020.

[17] G. Wunder, Č. Stefanović, P. Popovski, and L. Thiele, "Compressive Coded Random Access for Massive MTC Traffic in 5G Systems," in *2015 49th Asilomar Conference on Signals, Systems and Computers*. IEEE, 2015, pp. 13–17.

[18] H. F. Schepker, C. Bockelmann, and A. Dekorsy, "Exploiting Sparsity in Channel and Data Estimation for Sporadic Multi-User Communication," in *ISWCS 2013; The Tenth International Symposium on Wireless Communication Systems*. VDE, 2013, pp. 1–5.

[19] G. Choudhury and S. Rappaport, "Diversity ALOHA-A Random Access Scheme for Satellite Communications," *IEEE Transactions on Communications*, vol. 31, no. 3, pp. 450–457, 1983.

[20] E. Paolini, C. Stefanovic, G. Liva, and P. Popovski, "Coded Random Access: Applying Codes on Graphs to Design Random Access Protocols," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 144–150, 2015.

[21] E. Casini, R. D. Gaudenzi, and H. O. D. R., "Contention Resolution Diversity Slotted ALOHA (CRDSA): An Enhanced Random Access Scheme for Satellite Access Packet Networks," *IEEE Trans. on Wireless Commun.*, vol. 6, no. 4, p. 1408–1419, 2007.

[22] E. Paolini, G. Liva, and M. Chiani, "Coded Slotted ALOHA: A Graph-Based Method for Uncoordinated Multiple Access," *IEEE Transactions on Information Theory*, vol. 61, no. 12, pp. 6815–6832, Dec 2015.

[23] J.-L. Lu, W. Shu, and M.-Y. Wu, "A Survey on Multipacket Reception for Wireless Random Access Networks," *Journal of Computer Networks and Communications*, vol. 2012, 2012.

[24] M. Ghanbarinejad and C. Schlegel, "Irregular Repetition Slotted ALOHA with Multiuser Detection," in *Wireless On-demand Network Systems and Services (WONS)*, March 2013, pp. 201–205.

[25] Z. Chen, Y. Feng, C. Feng, L. Liang, Y. Jia, and T. Q. S. Quek, "Optimal Distribution Design for Irregular Repetition Slotted ALOHA with Multi-Packet Reception," 2021, working paper or preprint. [Online]. Available: https://arxiv.org/abs/2110.08166v1

[26] F. Clazzer, E. Paolini, I. Mambelli, and C. Stefanovic, "Irregular Repetition Slotted ALOHA Over the Rayleigh Block Fading Channel with Capture," in *ICC*. IEEE, 2017.

[27] G. Interdonato, S. Pfletschinger, F. Vázquez-Gallego, J. Alonso-Zarate, and G. Araniti, "Intra-slot Interference Cancellation for Collision Resolution in Irregular Repetition Slotted ALOHA," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 2069–2074.

[28] F. Babich and M. Comisso, "Coded Slotted Aloha (CSA) with Capture," in *European Wireless 2018; 24th European Wireless Conference*, 2018, pp. 1–6.

[29] ——, "Impact of Header on Coded Slotted Aloha with Capture," in *2019 IEEE Symposium on Computers and Communications (ISCC)*, 2019, pp. 1–6.

[30] ——, "Impact of Segmentation and Capture on Slotted Aloha Systems Exploiting Interference Cancellation," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 2878–2892, 2019.

[31] C. Stefanović, M. Momoda, and P. Popovski, "Exploiting Capture Effect in Frameless ALOHA for Massive Wireless Random Access," in *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, 2014, pp. 1762–1767.

[32] E. K. C. A. A. Alloum and P. Mühlethaler, "Near-far Effect on Coded Slotted ALOHA," *2017-IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–7, Octobre 2017.

[33] A. Kumar, P. Hegde, R. Vaze, A. Alloum, and C. Adjih, "Breaking the Unit Throughput Barrier in Distributed Systems," *arXiv preprint arXiv:2010.07430*, 2020.

[34] B. Zhao, G. Ren, X. Dong, and H. Zhang, "Optimal Irregular Repetition Slotted ALOHA Under Total Transmit Power Constraint in IoT-Oriented Satellite Networks," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 465–10 474, 2020.

[35] A. Hasanzadeh, J.-F. Chamberland, and K. Narayanan, "Rate Selection and Power Adaptation Using Maximal Ratio Combining for the Random Access Gaussian Channel," January 2018.

[36] J. Dia, F. Zesong, and Z. Yasheng, "Irregular Repetition Slotted ALOHA with Total Transmit Power Limitation," *Information Sciences*, vol. 64, no. 2, p. 129301, 2019.

[37] T. Nonaka, T. Fujii, O. Takyu, and M. Ohta, "Adapting the Number of Replicas in the E-IRSA System Using the Power Control," in *2020 International Conference on Information Networking (ICOIN)*, 2020, pp. 787–792.

[38] Z. Sun, Y. Xie, J. Yuan, and T. Yang, "Coded Slotted ALOHA for Erasure Channels: Design and Throughput Analysis," *IEEE Trans. Commun.*, vol. 65, no. 11, pp. 4817–4830, 2017.

[39] M. Ivanov, F. Brannstrom, A. Graell i. Amat, and P. Popovski, "Error Floor Analysis of Coded Slotted ALOHA Over Packet Erasure Channels," *IEEE Commun. Letters*, vol. 19, no. 3, pp. 419–422, March 2015.

[40] M. Ivanov, F. Brännström, A. Graell i Amat, and G. Liva, "Unequal Error Protection in Coded Slotted ALOHA," *IEEE Wireless Communications Letters*, vol. 5, no. 5, pp. 536–539, 2016.

[41] Y. Yuan, Z. Yuan, and L. Tian, "5G Non-Orthogonal Multiple Access Study in 3GPP," *IEEE Communications Magazine*, vol. 58, no. 7, pp. 90–96, 2020.

[42] E. Balevi, F. T. A. Rabee, and R. D. Gitlin, "ALOHA-NOMA for Massive Machine-to-Machine IoT Communication," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1–5.

[43] X. S. Z. S. M. Y. S. Gu and Q. Guo, "NOMA-Based Irregular Repetition Slotted ALOHA for Satellite Networks," *IEEE Communications Letters*, vol. 23, no. 4, pp. 624–627, April 2019.

[44] S. A. Tegos, P. D. Diamantoulakis, A. S. Lioumpas, P. G. Sarigiannidis, and G. K. Karagiannidis, "Slotted ALOHA with NOMA for the Next Generation IoT," *IEEE Transactions on Communications*, vol. 68, no. 10, pp. 6289–6301, 2020.

[45] J. Choi, "NOMA-Based Random Access With Multichannel ALOHA," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 12, pp. 2736–2743, 2017.

[46] ——, "Multichannel NOMA-ALOHA Game With Fading," *IEEE Transactions on Communications*, vol. 66, no. 10, pp. 4997–5007, 2018.

[47] T. Richardson and R. Urbanke, *Modern Coding Theory*. USA: Cambridge University Press, 2008.

[48] M. E. Newman, S. H. Strogatz, and D. J. Watts, "Random Graphs with Arbitrary Degree Distributions and Their Applications," *Physical review E*, vol. 64, no. 2, p. 026118, 2001.

[49] H. S. Wilf, *Generating Functionology*. CRC press, 2005.

[50] A. Lushnikov, "Evolution of Coagulating Systems. II. Asymptotic Size Distributions and Analytical Properties of Generating Functions," *Journal of Colloid and Interface Science*, vol. 48, no. 3, pp. 400–409, 1974.

[51] M. Luby, M. Mitzenmacher, and A. Shokrollahi, "Analysis of Random Processes via And-Or Tree Evaluation," 08 1998.

[52] F. Lázaro and C. Stefanović, "Finite-Length Analysis of Frameless ALOHA With Multi-User Detection," *IEEE Com. Lett.*, vol. 21, no. 4, pp. 769–772, April 2017.

[53] J.-L. Lu, W. Shu, and M.-Y. Wu, "A Survey on Multipacket Reception for Wireless Random Access Networks," *Journal of Computer Networks and Communications*, 2012.

[54] A. Ananthapadnabhan, D. Kumar, and S. S, "A Survey on Multi-Packet Reception With Random Access in Wireless Networks," *International Journal of Scientific and Engineering Research*, vol. Volume 7, pp. 8–12, 04 2016.

[55] S. Verdu, *Multiuser Detection*, 1st ed. USA: Cambridge University Press, 1998.

[56] W. L. Huang, K. B. Letaief, and Y. J. Zhang, "Cross-Layer Multi-Packet Reception Based Medium Access Control and Resource Allocation for Space-Time Coded MIMO/OFDM," *IEEE Transactions on Wireless Communications*, vol. 7, no. 9, pp. 3372–3384, 2008.

[57] G. Liva, E. Paolini, M. Lentmaier, and M. Chiani, "Spatially-Coupled Random Access on Graphs," in *2012 IEEE International Symposium on Information Theory Proceedings*, July 2012, pp. 478–482.

[58] H. Yu, Z. Fei, C. Cao, M. Xiao, D. Jia, and N. Ye, "Analysis of Irregular Repetition Spatially-Coupled Slotted ALOHA," *Science China Information Sciences*, vol. 62, no. 8, p. 80302, 2019.

[59] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth, "On the Lambert W Function," *Advances in Computational Mathematics*, vol. 5, no. 1, pp. 329–359, Dec 1996. [Online]. Available: https://doi.org/10.1007/BF02124750

[60] K. Narayanan, "The Peeling Decoder: Theory and some Applications," http://pfister.ee.duke.edu/nasit16/Narayanan.pdf, 2016, accessed: 2022–01-10.

[61] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge university press, 2008.

[62] G. Liva, "Graph-Based Analysis and Optimization of Contention Resolution Diversity Slotted ALOHA," *IEEE Transactions on Communications*, vol. 59, no. 2, pp. 477–487, February 2011.

[63] N. L. Johnson and S. Kotz, *Urns Models and Their Applications: An Approach To Modern Discret Probability Theory*. USA: JOHN WILEY & SONS, 1977.

[64] Y. Onozato, J. Liu, and S. Noguchi, "Stability of a Slotted ALOHA System with Capture Effect," *IEEE Transactions on Vehicular Technology*, vol. 38, no. 1, pp. 31–36, 1989.

[65] F. Baccelli and B. Blaszczyszynz, *Stochastic Geometry and Wireless Networks, Volume I - Theory*. NoW Publishers,1, 2009.

[66] X. Zhang and M. Haenggi, "The Performance of Successive Interference Cancellation in Random Wireless Networks," *IEEE Transactions on Information Theory*, vol. 60, no. 10, pp. 6368 –6388, 2014.

[67] S. A. S. Durrani and X. Zhou, "Enhancing CRDSA With Transmit Power Diversity for Machine-Type Communication," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 8, pp. 7790–7794, August 2018.

[68] E. Paolini, G. Liva, and M. Chiani, "Coded Slotted ALOHA: A Graph-based Method for Uncoordinated Multiple Access," *IEEE Trans. Inf. Theory*, vol. 61, no. 12, pp. 6815–6832, 2015.

[69] C. R. Srivatsa and C. R. Murthy, "Throughput Analysis of PDMA/IRSA under Practical Channel Estimation," in *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Jul. 2019, pp. 1–5, ISSN: 1948-3252.

[70] H.-C. Bui, K. Zidane, J. Lacan, and M.-L. Boucheret, "A Multi-Replica Decoding Technique for Contention Resolution Diversity Slotted ALOHA," in *IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, 2015, pp. 1–5.

[71] K. Zidane, J. Lacan, M.-L. Boucheret, C. Poulliat, M. Gineste, D. Roques, C. Bes, and A. Deramecourt, "Effect of Residual Channel Estimation Errors in Random Access Methods for Satellite Communications," in *IEEE 81st Vehicular Technology Conference (VTC Spring)*, 2015, pp. 1–5.

[72] N. I. Miridakis and D. D. Vergados, "A Survey on the Successive Interference Cancellation Performance for Single-Antenna and Multiple-Antenna OFDM Systems," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 1, pp. 312–335, 2013.

[73] S. M. R. Islam, N. Avazov, O. A. Dobre, and K.-s. Kwak, "Power-Domain Non-Orthogonal Multiple Access (NOMA) in 5G Systems: Potentials and Challenges," *IEEE Communications Surveys Tutorials*, vol. 19, no. 2, pp. 721–742, 2017.

[74] C. Stefanovic, P. Popovski, and D. Vukobratovic, "Frameless ALOHA Protocol for Wireless Networks," *IEEE Commun. Lett.*, vol. 16, no. 12, pp. 2087–2090, Dec. 2012.

[75] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic Information Transfer Functions: Model and Erasure Channel Properties," *IEEE Trans. on Information Theory*, vol. 50, no. 11, p. 2657–2673, Sep. 2006.

[76] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved Low-Density Parity-Check Codes Using Irregular Graphs," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 585–598, 2001.

[77] S. Diamond and S. Boyd, "CVXPY: A Python-Embedded Modeling Language for Convex Optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, 2016.

[78] P. Popovski, v. Stefanović, J. J. Nielsen, E. de Carvalho, M. Angjelichinoski, K. F. Trillingsgaard, and A.-S. Bana, "Wireless Access in Ultra-Reliable Low-Latency Communication (URLLC)," *IEEE Trans. on Communications*, vol. 67, no. 8, pp. 5783–5801, 2019.

[79] H. Yang, X. Xie, and M. Kadoch, "Machine Learning Techniques and A Case Study for Intelligent Wireless Networks," *IEEE Network*, vol. 34, no. 3, pp. 208–215, 2020.

[80] H. Fourati, R. Maaloul, and L. Chaari, "A Survey of 5G Network Systems: Challenges and Machine Learning Approaches," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 2, pp. 385–431, 2021.

[81] Y. Chu, S. Kosunalp, P. D. Mitchell, D. Grace, and T. Clarke, "Application of Reinforcement Learning to Medium Access Control for Wireless Sensor Networks," *Engineering Applications of Artificial Intelligence*, vol. 46, pp. 23–32, 2015.

[82] S. Kosunalp, P. D. Mitchell, D. Grace, and T. Clarke, "Practical Implementation Issues of Reinforcement Learning Based ALOHA for Wireless Sensor Networks," in *ISWCS 2013; The Tenth International Symposium on Wireless Communication Systems*, 2013, pp. 1–5.

[83] Y. Chu, P. D. Mitchell, and D. Grace, "Reinforcement Learning based ALOHA for Multi-Hop Wireless Sensor Networks with Informed Receiving," in *IET Conference on Wireless Sensor Systems (WSS 2012)*, 2012, pp. 1–6.

[84] M. Zhang, L. de Alfaro, and J. Garcia-Luna-Aceves, "UsingReinforcement Learning in Slotted ALOHA for Ad-Hoc Networks," in *Proceedings of the 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2020, pp. 245–252.

[85] L. Xiao, Y. Li, C. Dai, H. Dai, and H. V. Poor, "Reinforcement Learning-Based NOMA Power Allocation in the Presence of Smart Jamming," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 4, pp. 3377–3389, 2018.

[86] J. Zhang, X. Tao, H. Wu, N. Zhang, and X. Zhang, "Deep Reinforcement Learning for Throughput Improvement of the Uplink Grant-Free NOMA System," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6369–6379, 2020.

[87] D. Fooladivanda, A. Al Daoud, and C. Rosenberg, "Joint Resource Allocation and User Association for Heterogeneous Wireless Cellular Networks," vol. 12, 09 2011, pp. 384–390.

[88] X. Ge, X. Li, H. Jin, J. Cheng, and V. C. M. Leung, "Joint User Association and User Scheduling for Load Balancing in Heterogeneous Networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 5, pp. 3211–3225, 2018.

[89] C. Nguyen, H. Dinh Thai, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of Deep Reinforcement Learning in Communications and Networking: A Survey," 10 2018.

[90] O. Naparstek and K. Cohen, "Deep Multi-User Reinforcement Learning for Dynamic Spectrum Access in Multichannel Wireless Networks," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, 2017, pp. 1–7.

[91] A. Destounis, D. Tsilimantos, M. Debbah, and G. S. Paschos, "Learn2MAC: Online Learning Multiple Access for URLLC Applications," in *IEEE INFOCOM*

2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, 2019, pp. 1–6.

[92] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep Reinforcement Learning for Dynamic Multichannel Access in Wireless Networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 2, pp. 257–265, 2018.

[93] Y. Chu, P. D. Mitchell, and D. Grace, "ALOHA and Q-Learning based Medium Access Control for Wireless Sensor Networks," in *Proceedings of ISWCS 2012*, Aug. 2012, pp. 511–515, iSSN: 2154-0225.

[94] L. Toni and P. Frossard, "IRSA Transmission Optimization via Online Learning," *arXiv:1801.09060 [cs, math]*, Jan. 2018, arXiv: 1801.09060. [Online]. Available: http://arxiv.org/abs/1801.09060

[95] E. Nisioti and N. Thomos, "Fast Q-Learning for Improved Finite Length Performance of Irregular Repetition Slotted ALOHA," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 2, pp. 844–857, 2020.

[96] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*, 2nd ed., ser. Adaptive Computation and Machine Learning Series. Cambridge, Massachusetts: The MIT Press, 2018.

[97] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep Reinforcement Learning for Multiagent Systems: A Review of Challenges, Solutions, and Applications," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3826–3839, 2020.

[98] J. Balcázar, F. Bonchi, A. Gionis, and M. Sebag, *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part II*, 01 2010, vol. 6322.

[99] D. Bloembergen, K. Tuyls, D. Hennes, and M. Kaisers, "Evolutionary Dynamics of Multi-Agent Learning: A Survey," *JAIR*, vol. 53, pp. 659–697, Aug. 2015. [Online]. Available: https://jair.org/index.php/jair/article/view/10952

[100] A. Blum and Y. Mansour, "Learning, Regret Minimization, and Equilibria," in *Algorithmic Game Theory*, N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, Eds. Cambridge University Press, 2007, pp. 79–102. [Online]. Available: https://www.cambridge.org/core/product/identifier/CBO9780511800481A051/type/book_part

[101] M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione, "Regret Minimization in Games with Incomplete Information," p. 8.

[102] D. N. K. Akkarajitsakul, E. Hossain and D. I. Kim, "Game Theoretic Approaches for Multiple Access in Wireless Networks: A Survey, in IEEE Communications Surveys & Tutorials,," *IEEE Trans. Commun.*, vol. 13, no. 3, pp. 372–395, Sep. 2011.

[103] T. Cui, L. Chen, and S. H. Low, "A Game-Theoretic Framework for Medium Access Control," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 7, pp. 1116–1127, 2008.

[104] A. MacKenzie and S. Wicker, "Selfish Users in ALOHA: "A Game-Theoretic Approach," in *IEEE 54th Vehicular Technology Conference. VTC Fall 2001. Proceedings (Cat. No.01CH37211)*, vol. 3, 2001, pp. 1354–1357 vol.3.

[105] H. Inaltekin and S. B. Wicker, "The Analysis of Nash Equilibria of the One-Shot Random-Access Game for Wireless Networks and the Behavior of Selfish Nodes," *IEEE/ACM Transactions on Networking*, vol. 16, no. 5, pp. 1094–1107, 2008.

[106] E. Altman, N. Bonneau, m. Debbah, and G. Caire, "An Evolutionary Game Perspective to ALOHA with Power Control," 02 2014.

[107] J. Choi, "A Game-Theoretic Approach for NOMA-ALOHA," in *2018 European Conference on Networks and Communications (EuCNC)*, 2018, pp. 54–9.

[108] F. Clazzer, "Selfish Users in Graph-Based Random Access," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2018, pp. 1960–1966.

[109] I. Hmedoush, C. Adjih, P. Mühlethaler, and L. Salaün, "Multi-Power Irregular Repetition Slotted ALOHA in Heterogeneous IoT Networks," in *2020 9th IFIP International Conference on Performance Evaluation and Modeling in Wireless Networks (PEMWN)*, 2020, pp. 1–6.

[110] S. Lasaulce, M. Debbah, and E. Altman, "Methodologies for Analyzing Equilibria in Wireless Games ," in ieee signal processing magazine," *IEEE Trans. Commun.*, vol. 26, no. 5, pp. 41–52, 2009.

[111] S. G. Krantz and H. R. Parks, *The Implicit Function Theorem: History, Theory, and Applications*. Springer Science & Business Media, 2012.

[112] J. W. Friedman and C. Mezzetti, "Learningin Games by Random Sampling," *Journal of Economic Theory*, vol. 98, no. 1, pp. 55–84, 2001.

[113] M. C. Moroğlu, H. M. Gürsu, F. Clazzer, and W. Kellerer, "Short Frame Length Approximation for IRSA," *IEEE Wireless Communications Letters*, vol. 9, no. 11, pp. 1933–1936, 2020.

[114] W. Kautz and R. Singleton, "Nonrandom Binary Superimposed Codes," *IEEE Transactions on Information Theory*, vol. 10, no. 4, pp. 363–377, 1964.

[115] H. A. Inan, S. Ahn, P. Kairouz, and A. Ozgur, "A Group Testing Approach to Random Access for Short-Packet Communication," in *2019 IEEE International Symposium on Information Theory (ISIT)*, 2019, pp. 96–100.

[116] H. Zhu and G. B. Giannakis, "Exploiting Sparse User Activity in Multiuser Detection," *IEEE Transactions on Communications*, vol. 59, no. 2, pp. 454–465, 2011.

[117] J. Massey and P. Mathys, "The Collision Channel Without Feedback," *IEEE Transactions on Information Theory*, vol. 31, no. 2, pp. 192–204, 1985.

[118] E. Paolini, G. Liva, and A. G. i Amat, "A Structured Irregular Repetition Slotted ALOHA Scheme with Low Error Floors," in *IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.

[119] C. Boyd, R. Vehkalahti, O. Tirkkonen, and A. Laaksonen, "Code Design Principles for Ultra-Reliable Random Access with Preassigned Patterns," in *2019 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2019, pp. 2604–2608.

[120] D. Duchemin, J.-M. Gorce, and C. Goursaud, "Code Domain Non Orthogonal Multiple Access versus ALOHA: A simulation based study," in *25th International Conference on Telecommunications (ICT)*, 2018, pp. 445–450.

[121] Y. Zhang, Y. Chen, Y.-H. Lo, and W. S. Wong, "The Zero-Error Capacity of a Collision Channel With Successive Interference Cancellation," in *2017 IEEE International Symposium on Information Theory (ISIT)*, 2017, pp. 1653–1657.

[122] H. Yu, Y. Kang, Z. Shi, Y. Shao, Y. Lin, and Y. Zhang, "Design of Deterministic Grant-Free Access with Deep Reinforcement Learning," in *2020 IEEE 20th International Conference on Communication Technology (ICCT)*, 2020, pp. 944–948.

[123] B.-M. Robaglia, A. Destounis, M. Coupechoux, and D. Tsilimantos, "Deep reinforcement learning for scheduling uplink iot traffic with strict deadlines," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 1–6.

[124] I. Ayoub, I. Hmedoush, C. Adjih, K. Khawam, and S. Lahoud, "Deep-IRSA: A Deep Reinforcement Learning Approach to Irregular Repetition Slotted ALOHA," in *10th IFIP International Conference on Performance Evaluation and Modeling in Wireless and Wired Networks (PEMWN)*, 2021, pp. 1–6.

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

[126] B. Verweij, S. Ahmed, A. J. Kleywegt, G. Nemhauser, and A. Shapiro, "The Sample Average Approximation Method Applied to Stochastic Routing Problems: a Computational Study," *Computational optimization and applications*, vol. 24, no. 2, pp. 289–333, 2003.

[127] D. Bertsekas and R. Gallager, "Data Networks," *(2nd edition), Prentice Hall*, 1992.

[128] H. Al-Mefleh and O. Al-Kofahi, "Taking Advantage of Jamming in Wireless Networks: A Survey," *Computer Networks*, vol. 99, pp. 99–124, 2016.

[129] E. Pesce and G. Montana, "Improving Coordination in Small-Scale Multi-Agent Deep Reinforcement Learning Through Memory-Driven Communication," *Machine Learning*, pp. 1–21, 2020.

[130] M. L. Molle and G. C. Polyzos, "Conflict Resolution Algorithms and Their Performance Analysis," *University of Toronto, CS93-300, Tech. Rep*, 1993.

[131] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, "Stable Baselines," https://github.com/hill-a/stable-baselines, 2018.

[132] K. Zidane, "Techniques D'amélioration des Performances des Méthodes D'accès Aléatoire Synchrones pour Les Communications par Satellite," Ph.D. dissertation, Toulouse, ISAE, 2016.

[133] H.-C. Bui, J. Lacan, and M.-L. Boucheret, "MultiDegree Distribution," in *IEEE First AESS European Conference on Satellite Telecommunications (ESTEL)*, 2012, pp. 1–6.

[134] J. Choi, "Re-Transmission Diversity Multiple Access Based on SIC and HARQ-IR," *IEEE Transactions on Communications*, vol. 64, no. 11, pp. 4695–4705, 2016.

[135] Y. Ma, Z. Yuan, W. Li, and Z. Li, "Novel Solutions to NOMA-Based Modern Random Access for 6G-Enabled IoT," *IEEE Internet of Things Journal*, vol. 8, no. 20, pp. 15 382–15 395, 2021.

[136] M. Elkourdi, A. Mazin, E. Balevi, and R. D. Gitlin, "Enabling Slotted ALOHA-NOMA for Massive M2M Communication in IoT Networks," in *2018 IEEE 19th Wireless and Microwave Technology Conference (WAMICON)*, 2018, pp. 1–4.

[137] N. Ye, J. An, and J. Yu, "Deep-Learning-Enhanced NOMA Transceiver Design for Massive MTC: Challenges, State of the Art, and Future Directions," *IEEE Wireless Communications*, vol. 28, no. 4, pp. 66–73, 2021.

[138] A. Munari, "Modern Random Access: An Age of Information Perspective on Irregular Repetition Slotted ALOHA," *IEEE Transactions on Communications*, vol. 69, no. 6, pp. 3572–3585, 2021.

[139] S. Saha, V. B. Sukumaran, and C. R. Murthy, "On the Minimum Average Age of Information in IRSA for Grant-FreemMTC," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 5, pp. 1441–1455, 2021.