



# **LoRa Networking in Mobile Scenarios using UAV Gateways**

**Marco Stellin**

Thesis to obtain the Master of Science Degree in

## **Electrical and Computer Engineering**

Supervisors: Prof. António Manuel Raminhos Cordeiro Grilo  
Sérgio Sabino, M.Sc.

### **Examination Committee**

Chairperson: Prof. Horácio Cláudio De Campos Neto  
Supervisor: Prof. António Manuel Raminhos Cordeiro Grilo  
Member of the Committee: Prof. Manuel Alberto Pereira Ricardo

**October 2018**



## **Declaration**

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.



## **Abstract**

In harsh and remote environments like mountains, forests or sub-urban areas, traditional communication technologies might be unavailable or they might offer an intermittent and unreliable service. Immediately after a disaster, such as a flood, a wildfire or an earthquake, networks might be congested or disrupted and not suitable for supporting the traffic generated by rescuers. In these situations, the use of a traditional fixed-gateway approach, would not be effective due to the mobility of the rescuers. In the present work, a double-layer network system called LoRaUAV has been designed and evaluated with the purpose of finding a solution to the aforementioned issues. LoRaUAV is based on a WiFi ad hoc network of Unmanned Aerial Vehicles (UAVs) gateways acting as relays for the traffic generated between mobile LoRaWAN nodes and a remote Base Station (BS). The core of the system is a completely distributed mobility algorithm based on virtual spring forces that periodically updates the UAV topology to adapt to the movement of ground nodes. The system has been successfully implemented in a simulation environment and has been assessed using the Packet Reception Rate (PRR) and the delay as the main QoS metrics. It is observed that the mechanisms implemented in LoRaUAV effectively help improving the PRR, with the only disadvantage of a bigger delay affecting a small percentage of packets caused by buffer delays and disconnections.

**Keywords:** LoRaWAN, UAV, WiFi, relay, ad hoc



## **Resumo**

Em ambientes hostis e remotos, como montanhas, florestas ou áreas suburbanas, as tecnologias de comunicação tradicionais podem não estar disponíveis ou podem oferecer um serviço intermitente e não confiável. Imediatamente após um desastre, como uma inundação, um incêndio florestal ou um terremoto, as redes podem estar congestionadas ou interrompidas e não serem adequadas para apoiar o tráfego gerado pelos socorristas. Nessas situações, o uso de uma abordagem tradicional baseada em gateways fixos não seria eficaz devido à mobilidade dos socorristas. No presente trabalho, um sistema de rede chamado LoRaUAV foi projetado e avaliado com o objetivo de encontrar uma solução para os problemas mencionados anteriormente. O LoRaUAV é baseado em uma rede ad hoc WiFi de gateways montados em UAVs que atuam como retransmissores para o tráfego gerado entre nós LoRaWAN móveis e uma BS remota. O núcleo do sistema é um algoritmo de mobilidade completamente distribuído baseado em forças de mola virtuais que atualiza periodicamente a topologia dos UAV para se adaptar ao movimento dos nós que se movem na superfície. O sistema foi implementado com sucesso em um ambiente de simulação e foi avaliado usando o PRR e o atraso como as principais métricas de QoS. Observa-se que os mecanismos implementados no LoRaUAV efetivamente ajudam a melhorar o PRR, com a única desvantagem de um maior atraso que afeta uma pequena porcentagem de pacotes e causado pelo tempo de buffer e pelas desconexões.

**Palavras-Chave:** LoRaWAN, UAV, WiFi, relay, ad hoc





# Contents

<b>List of Symbols</b>	<b>ix</b>
<b>List of Tables</b>	<b>xi</b>
<b>List of Figures</b>	<b>xiii</b>
<b>Acronyms</b>	<b>xv</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	2
1.2 Objectives . . . . .	3
1.3 Thesis Outline . . . . .	3
<b>2 Background</b>	<b>5</b>
2.1 Low Power Wide Area Networks . . . . .	5
2.2 LoRaWAN . . . . .	7
2.2.1 Architecture and Topology . . . . .	7
2.2.2 LoRaWAN Physical Layer . . . . .	8
2.2.3 End Devices (EDs) Operation Modes . . . . .	11
2.2.4 MAC Frames and Commands . . . . .	11
2.2.5 Adaptive Data Rate . . . . .	13
2.2.6 Activation of End Devices (EDs) . . . . .	13
2.3 The IEEE 802.11 Protocol . . . . .	15
2.3.1 IEEE 802.11 Architecture . . . . .	15
2.3.2 IEEE 802.11 MAC Layer . . . . .	15
2.3.3 IEEE 802.11 PHY Layer . . . . .	17
2.4 MANET Routing Protocols . . . . .	19
2.4.1 Proactive Routing: DSDV and OLSR . . . . .	19
2.4.2 Reactive Routing: DSR and AODV . . . . .	20
2.4.3 Geographical Routing: GPSR . . . . .	21
2.5 Unmanned Aerial Vehicles . . . . .	22
2.5.1 UAVs classification . . . . .	22
2.5.2 UAV system . . . . .	23
<b>3 Related Work</b>	<b>25</b>
3.1 Performance Assessments of LoRa and LoRaWAN . . . . .	25
3.2 Network Simulators and LoRaWAN Simulation Modules . . . . .	28
3.3 UAVs Path Planning and Optimal Placement for Relaying . . . . .	31
3.4 Summary . . . . .	34

<b>4</b>	<b>LoRaUAV Design and Simulation</b>	<b>35</b>
4.1	Architecture . . . . .	35
4.2	Assumptions . . . . .	38
4.3	LoRaUAV Mobility . . . . .	38
4.3.1	The VSF Mobility Algorithm . . . . .	38
4.3.2	Connection Recovery and Maintenance (CRM) Algorithm . . . . .	40
4.3.3	Movement Prediction (MP) Algorithm . . . . .	42
4.4	Simulation Model . . . . .	44
4.4.1	Simulation Scenario . . . . .	45
4.4.2	Channel Propagation Models . . . . .	45
4.4.3	Mobility Model of UAVs . . . . .	47
4.4.4	Mobility Model of Firefighters . . . . .	48
4.4.5	Applications . . . . .	49
4.4.6	Simulation Script . . . . .	49
4.5	Summary . . . . .	50
<b>5</b>	<b>Results and Analysis</b>	<b>51</b>
5.1	Simulation Parameters . . . . .	51
5.2	QoS Metrics . . . . .	53
5.3	Generation of Scenarios . . . . .	53
5.4	Behaviour and Performance of the VSF Algorithm . . . . .	54
5.4.1	Study of the Impact of $K_p$ on the Coverage of GNs . . . . .	55
5.4.2	Comparison between LoRaUAV VSF and DF VSF algorithms . . . . .	61
5.5	Study of the impact of the Connection Recovery and Maintenance (CRM) algorithm . . . . .	64
5.6	Study of the impact of the MP algorithm . . . . .	68
5.7	Frequency and Duration of Disconnections . . . . .	71
5.8	Summary . . . . .	75
<b>6</b>	<b>Conclusions</b>	<b>77</b>
6.1	Future Work . . . . .	78
	<b>Bibliography</b>	<b>80</b>
<b>A</b>	<b>Additional Plots</b>	<b>85</b>
A.1	Trend of some QoS metrics with the basic LoRaUAV VSF algorithm for 3 teams of Ground Nodes (GNs) . . . . .	85

# List of Symbols

$P_{max}$	Maximum number of pause intervals. 40
$\Delta T$	Mobility algorithm time step. 39, 40
$D_{BS}$	Distance from the BS. 40
$K_{AtA}$	Weight of AtA springs. 39
$K_{AtG}$	Weight of AtG springs. 39
$K_p$	Weight factor of AtA springs. 39
$LB_{ij}$	Link budget of link $ij$ . 39
$LB_{req}$	Desired value of link budget. 39
$N_{neighs}^{max}$	Maximum number of UAV neighbours. 39
$R_{max}$	Maximum estimated range of a UAV. 40
$\vec{F}_{ij}$	Force between node $i$ and node $j$ . 39
$\vec{F}_{tot}^i$	Total force applied to drone $i$ . 39
$k_{max}$	Maximum number of clusters. 44
$n_{neighs}$	Current number of UAV neighbours. 39
$p_{shared}$	Percentage of shared GNs. 41
$s_{avg}$	Average silhouette index. 44
$s_{thr}$	Minimum silhouette index value. 44
$t_{AtG}$	Length of AtG time window. 39, 48
$t_{exp}$	Expiration time of an hologram. 44
$t_{load}$	Length of time window of <i>load</i> parameter. 40, 48
$t_{lost}$	Time after which a GN is marked as lost. 48
$t_r$	Maximum time between registered GN entries. 48
$u_j$	Number of UAVs covering GN $j$ . 39
$u_{max}$	Current biggest number of UAVs sharing also some GNs with the current UAV. 39



# List of Tables

2.1	LoRa Data Rates. . . . .	10
2.2	MAC message types (MTypes). . . . .	13
3.1	EDs PHY configurations used in the experiments performed in [24]. . . . .	26
4.1	ns-3 modules used to simulate LoRaUAV. . . . .	44
4.2	Values of the parameters of the Log Distance model. . . . .	46
5.1	List of fixed simulation parameters. . . . .	52
5.2	Firefighters mobility model parameters. . . . .	54
5.3	Parameters used for simulations described in Section 5.4.1 and Section 5.4.2. . . . .	55
5.4	AE-PRR point by point minimum, maximum and average variation between the LoRaUAV VSF algorithm and the DF VSF algorithm. . . . .	63
5.5	Average value of the ATD in milliseconds for the LoRaUAV VSF algorithm and the DF algorithm and relative variation obtained for $K_p \in [25 - 30]$ . . . . .	64
5.6	Parameters used for simulations described in Section 5.5. . . . .	64
5.7	Difference between AE-PRR with and without CRM and relative confidence interval for some values of $K_p$ . . . . .	66
5.8	Buffered Packets and Average Buffer Delay with and without CRM algorithm for a scenario with 12 UAVs, 3 teams of GNs and $K_p = 25$ . . . . .	68
5.9	Parameters used for simulations described in Section 5.6. . . . .	69
5.10	Comparison between runs with VSF + MP and runs with only VSF when team splitting is not active. . . . .	69
5.11	Comparison between runs with VSF + MP and runs with only VSF when team splitting is active. . . . .	69
5.12	Comparison between runs with VSF + CRM + MP and runs with only VSF + CRM when team splitting is not active. . . . .	69
5.13	Comparison between runs with VSF + CRM + MP and runs with only VSF + CRM when team splitting is active. . . . .	70
5.14	Parameters used for simulations described in Section 5.7. . . . .	71
5.15	Total average disconnection time and relative confidence interval obtained with different algorithm combinations in a scenario with parameters listed in Table 5.14. . . . .	75



# List of Figures

1.1	LoRaUAV system architecture. . . . .	2
2.1	LoRaWAN stack. . . . .	7
2.2	Topology of a LoRaWAN network. . . . .	8
2.3	Visualisation of LoRa chirps (source: LinkLabs). . . . .	9
2.4	MAC frames structure. . . . .	12
2.5	Architecture of an IEEE 802.11 network. . . . .	16
2.6	Channels in the 2.4 GHz band [10]. . . . .	17
2.7	Failure of Greedy Perimeter Stateless Routing (GPSR) greedy forwarding behaviour [15].	21
2.8	Two of the most common drone configurations: fixed-wing and the multi-rotor. . . . .	23
2.9	A general UAV system [16]. . . . .	23
3.1	Fraction of correctly received packets with one Gateway (GW) and a varying number of EDs having different PHY configurations [24]. . . . .	27
3.2	Fraction of correctly received packets with multiple GWs and a varying number of EDs [24].	27
4.1	System stack of GNs. . . . .	36
4.2	System stack of UAVs. . . . .	37
4.3	System stack of the BS. . . . .	37
4.4	CRM algorithm. . . . .	41
4.5	Log Distance path loss trend for AtG links. . . . .	46
4.6	Firemen mobility model. All teams start from a single starting point and begin their ope- ration in the furthest cell from the origin in their assigned column. . . . .	49
5.1	Initial configuration of simulation scenarios. . . . .	54
5.2	AE-PRR trend with 3 teams of 20 GNs in a scenario with parameters listed in Table 5.3. .	57
5.3	AE-PRR trend with 1 team of 20 GNs in a scenario with parameters listed in Table 5.3. .	57
5.4	AE-PRR trend with 2 teams of 20 GNs in a scenario with parameters listed in Table 5.3. .	58
5.5	AE-PRR trend with 4 teams of 20 GNs in a scenario with parameters listed in Table 5.3. .	58
5.6	AE-PRR trend with 5 teams of 20 GNs in a scenario with parameters listed in Table 5.3. .	59
5.7	AE-PRR trend with 3 teams composed of different numbers of GNs in a scenario with parameters listed in Table 5.3. . . . .	59
5.8	ATD trend with 3 teams of 20 GNs in a scenario with parameters listed in Table 5.3. . . .	60
5.9	Average percentage of buffered packets in a scenario with 3 teams of 20 GNs and the parameters listed in Table 5.3. . . . .	60
5.10	AE-PRR comparison when the LoRaUAV VSF algorithm and the DF VSF algorithm are used in a scenario with 3 teams of 20 GNs. Parameters are listed in Table 5.3. . . . .	62
5.11	AE-PRR comparison when the LoRaUAV VSF algorithm and the DF VSF algorithm are used in a scenario with 4 teams of 20 GNs. Parameters are listed in Table 5.3. . . . .	62

5.12 AE-PRR comparison when the LoRaUAV VSF algorithm and the DF VSF algorithm are used in a scenario with 5 teams of 20 GNs. Parameters are listed in Table 5.3. . . . .	63
5.13 AE-PRR comparison when CRM algorithm is activated in a scenario with 3 teams of 20 GNs and parameters listed in Table 5.6. . . . .	65
5.14 AE-PRR obtained with VSF + CRM in scenario configured as reported in Table 5.3. . . .	67
5.15 ATD comparison when CRM algorithm is activated in a scenario with 3 teams of 20 GNs and parameters listed in Table 5.6. . . . .	67
5.16 AE-PRR with VSF+CRM+MP (no team splits) in scenario with parameters of Table 5.3. .	71
5.17 Frequency and duration of disconnections of 100 runs of a scenario using the DF Virtual Spring Force (VSF) algorithm and with parameters listed in Table 5.14. . . . .	73
5.18 Frequency and duration of disconnections of 100 runs of a scenario using the LoRaUAV VSF algorithm and with parameters listed in Table 5.14. . . . .	73
5.19 Frequency and duration of disconnections of 100 runs of a scenario using the LoRaUAV VSF + CRM algorithms and with parameters listed in Table 5.14. . . . .	74
5.20 Frequency and duration of disconnections of 100 runs of a scenario using the LoRaUAV VSF + CRM + Movement Prediction (MP) algorithms and with parameters listed in Table 5.14. . . . .	74



# Acronyms

**3GPP** Third Generation Partnership Project. 5

**A-MPDU** Aggregated MAC Protocol Data Unit. 17

**A-MSDU** Aggregated MAC Service Data Unit. 17

**ABP** Activation By Personalisation. 14

**ADR** Adaptive Data Rate. 11–13, 29, 31

**AES** Advanced Encryption Standard. 14

**AODV** Ad-hoc On-demand Distance Vector. 19–21, 30

**AP** Access Point. 15–17

**AS** Application Server. 7, 8, 12, 14, 36

**AtA** Air-to-Air. 32, 39, 44, 46, 48, 57

**AtG** Air-to-Ground. 25, 32, 39, 43, 44, 46, 48, 51, 53, 57, 76

**BAN** Body Area Network. 45

**BER** Bit Error Rate. 30

**BLE** Bluetooth Low Energy. 5

**BS** Base Station. iii, v, xiii, 2, 3, 35–38, 40, 41, 45, 50, 51, 55, 56, 58, 75, 76

**BSS** Basic Service Set. 15, 17

**BSSID** Basic Service Set ID. 15

**CA** Collision Avoidance. 15

**CD** Collision Detection. 15

**CID** Command Identifier. 12

**CP** Command Post. 2, 3, 45, 49

**CRM** Connection Recovery and Maintenance. viii, xi, xiii, xiv, 40, 42, 48, 51, 53, 56, 58, 65–70, 72, 73, 75, 76

**CSMA** Carrier Sense Multiple Access. 15

**CSS** Chirp Spread Spectrum. 8, 9, 26

**CTS** Clear to Send. 16

**DCF** Distributed Coordination Function. 15, 16

**DFWMAC** Distribution Foundation Wireless MAC. 15

**DIFS** Distributed Interframe Space. 15, 16

**DS** Distribution System. 15

**DSDV** Destination-Sequenced Distance Vector. 19, 30

**DSR** Dynamic Source Routing. 19–21, 30

**DSSS** Direct Sequence Spread Spectrum. 8, 9, 17, 18, 26

**ED** End Device. vii, xi, xiii, 7, 8, 11–14, 25–27, 29, 31, 45, 50, 54

**ESS** Extended Service Set. 15

**ETSI** European Telecommunications Standards Institute. 5

**FEC** Forward Error Correction. 10

**FHDR** Frame Header. 11

**FHSS** Frequency Hopping Spread Spectrum. 17

**FPort** Frame Port. 11, 12

**GCS** Ground Control Station. 24, 45, 46

**GN** Ground Node. viii, xi, xiii, xiv, 35, 36, 38–41, 43, 44, 46, 48, 49, 51, 53–64, 66–70, 72, 73, 75, 76, 81, 83

**GPSR** Greedy Perimeter Stateless Routing. xiii, 21, 30

**GSE** Ground Support Equipment. 24

**GW** Gateway. xiii, 7, 11, 25–27, 29, 34, 36, 45, 51, 54, 58, 75

**IBSS** Independent Basic Service Set. 15, 17

**IEEE** Institute of Electrical and Electronic Engineers. 5, 15

**IETF** Internet Engineering Task Force. 5

**IFS** Interframe Space. 16

**IoT** Internet of Things. 1, 5

**ISM** Industrial, Scientific and Medical. 6, 15

**JS** Join Server. 8, 14, 36

**LAN** Local Area Network. 15

**LOS** Line Of Sight. 1, 24, 25, 46, 47

**LPWAN** Low Power Wide Area Network. 1, 2, 5–9, 26, 34

**MANET** Mobile Adhoc Network. 3, 5, 19, 20, 36, 44, 53

**MHDR** MAC Header. 11

**MIC** Message Integrity Code. 11, 14

**MP** Movement Prediction. 38, 51, 53, 56, 58, 70, 72, 73, 75, 76

**MPDU** MAC Protocol Data Unit. 17

**MSDU** MAC Service Data Unit. 17

**NAV** Network Allocation Vector. 16, 17

**NB** Narrowband. 6

**NRM** Network Recovery Mobility. 40, 41, 56

**NS** Network Server. 7, 12–14, 29, 31, 36

**OFDM** Orthogonal Frequency Division Multiplexing. 18

**OLSR** Optimized Link State Routing Protocol. 19, 20, 30, 36, 53

**OTA** Over The Air. 11, 14

**PCF** Point Coordination Function. 15–17

**PDR** Packet Drop Rate. 25

**PIFS** Point Interframe Space. 16

**PN** Pseudorandom Noise. 8, 9

**PRR** Packet Reception Rate. iii, v, 51, 53, 66, 70, 72, 73, 75, 76

**RF** Radio Frequency. 1, 8, 9

**RPV** Remote Piloted Vehicle. 22

**RTS** Request to Send. 16

**SDO** Standards Developing Organization. 5

**SF** Spreading Factor. 9–11, 13, 25, 26, 29, 30, 45, 46, 53, 54

**SFD** Start of Frame Delimiter. 10

**SIFS** Short Interframe Space. 15, 16

**SIR** Signal-to-Interference Ratio. 29, 30

**SM** Stationary Mobility. 40

**SNR** Signal to Noise Ratio. 8, 13

**SS** Spread Spectrum. 1, 6

**SSID** Service Set ID. 15, 17

**STA** Station. 15

**TTN** The Things Network. 25

**UAV** Unmanned Aerial Vehicle. iii, v, vii, xi, xiii, 1–3, 5, 22–25, 31–41, 43–51, 54–58, 63, 65, 66, 68, 70, 72, 73, 75, 76

**UNB** Ultra Narrowband. 6

**VSF** Virtual Spring Force. 33, 34, 36, 38–40, 45, 49, 51, 53, 56, 58, 63, 65, 66, 73, 75, 76

# Chapter 1

## Introduction

The ubiquity of the Internet, new and innovative communication protocols and the miniaturization of computational devices gave rise to a new paradigm called Internet of Things (IoT). The number of IoT devices is projected to grow steadily in the years to come <sup>1</sup>, especially thanks to their possible applications in a vast and diverse range of fields. IoT solutions are in fact being used in the industry, in agriculture, in smart cities and in many other sectors [1]. In many cases, IoT devices are battery powered and subject to very strict power constraints. For this reason, a new range of low power wireless communication protocols have been developed and standardized in order to support the operation of Low Power Wide Area Networks (LPWANs). These networks are typically formed by inexpensive, simple devices that need to communicate infrequently over long distances at low bitrates. LoRaWAN is one of the most promising and versatile technologies enabling LPWANs. LoRaWAN takes advantage of Spread Spectrum (SS), chirp orthogonality and the good propagation characteristics of the sub-GHz spectrum to provide reliable communication over long distances. This comes at the expense of the bit rate and of the maximum time interval between consecutive transmissions due to duty cycle limitations in the bands used by the protocol.

UAVs, commonly called drones, are flying vehicles that can operate without the need of a remote pilot. Exclusive prerogative of the military for many years, UAVs are now commercially available and their relatively low price makes them appealing for a wide variety of applications in a diverse range of scenarios. The price range can vary between 20\$-300\$ for small battery-powered Radio Frequency (RF) drones such as the *Parrot Bebop 2*, to more than 3000\$ for professional drones with advanced control systems and sensors such as the *DJI Spreading Wings S1000*. In the consumer market segment, drones are mainly used for taking aerial pictures and videos, while in the commercial segment the range of applications is more diverse: mapping of geographical areas, product delivery and logistic, data collection, crop monitoring and surveillance. Drones are increasingly being used also in applications such as crime prevention, weather and meteorology, search and rescue operations, border and maritime patrol and forest fire monitoring. In case of a disaster, such as an earthquake, a flood or a wildfire, UAVs can be used to support the operations of the rescuers, to localize the victims and to create a backhaul network in case of absence or disruption of the communication facilities. In this last circumstance, the relay network formed by UAVs can support the data generated by isolated IoT devices that have been deployed in advance or during the rescue operations. UAVs provide many advantages compared to ground-based solutions: movement in an obstacle-free environment, better overview of the monitored area, Line Of Sight (LOS) between the drone and the targets and faster data acquisition over large areas. On the other side, UAVs are still limited in terms of flying range, degree of autonomy and flying time. To overcome these issues, UAV swarms are increasingly being considered as a possible solution

---

<sup>1</sup><https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/>

and many efforts are being directed in the development of suitable communication protocols and swarm mobility models.

## 1.1 Motivation

The integration of UAVs and LPWAN protocols in disaster scenarios may offer a new cost-effective and energy efficient way of tackling many problems arising during the operations of the rescuers. Unfortunately, the related literature is still scarce, therefore leaving many issues open to further investigation. In this work, a particularly challenging scenario is considered: a forest in which a wildfire has been detected. Wildfires represent a worldwide problem, but America and Europe are the two continents where the incidence of forest fires is higher. The costs, in terms of economic losses and human deaths, are considerable [2] [3]. Wildfires typically affect rural or suburban areas where the network coverage of traditional communication networks (e.g. cellular networks) is scarce, intermittent or completely lacking. This situation is worsened by the huge propagation loss introduced by foliage and trees in the signal propagation path. In this situation UAVs can be used to establish a mesh network acting as a relay between the BS installed in the Command Post (CP), the place where operations are managed, and the sensors carried by firemen, therefore providing more precise situational awareness to the rescuers. The mobility of the fire fighters represents another huge challenge, since the UAVs mesh network needs to adjust its position while maintaining the connectivity to the BS and to firemen on the ground. In this thesis, LoRaUAV, a system based on LoRaWAN and WiFi, is designed and evaluated. Thanks to its good propagation characteristics, its power efficiency and its versatility, LoRaWAN represents a good candidate for sending data generated by sensors. A LoRaWAN module is carried by firefighters and takes care of periodically transmitting relevant data. The UAVs are equipped with LoRaWAN gateways able to receive data from the ground LoRaWAN devices and relaying that data over WiFi to the BS. WiFi is a widely used protocol that can provide high data rates in the unlicensed 2.4 GHz bandwidth, thus reducing the formation of bottlenecks in the mesh network. The architecture of the system is depicted in Figure 1.1

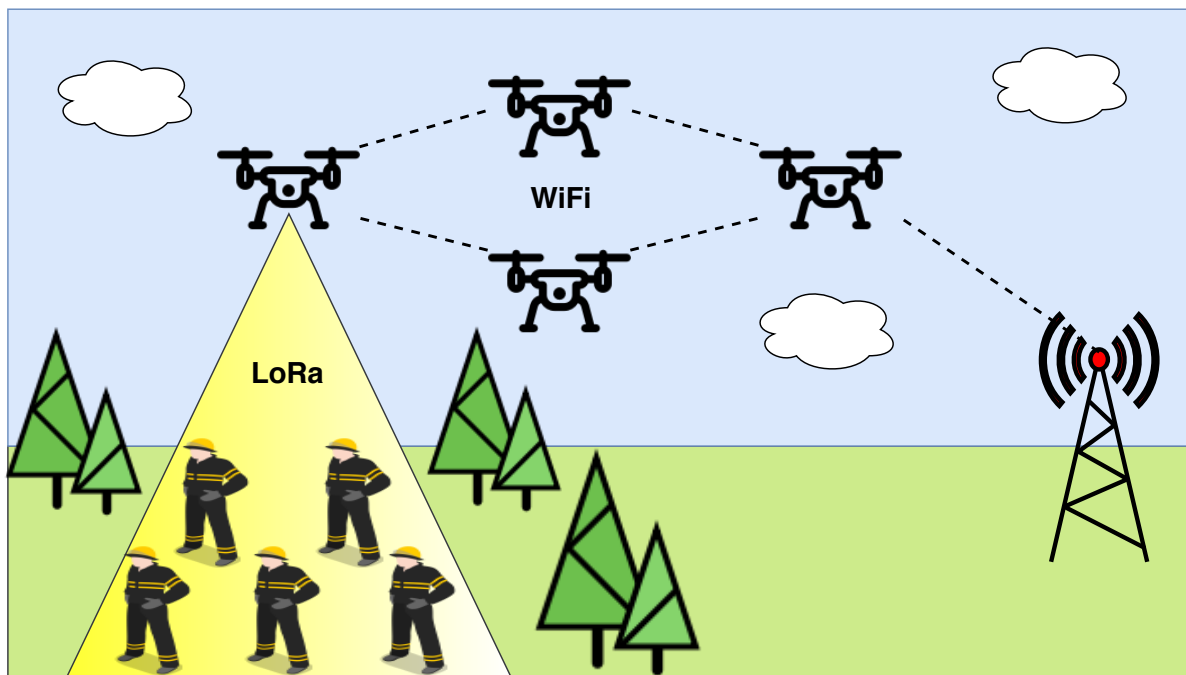


Figure 1.1: LoRaUAV system architecture.

## 1.2 Objectives

The objective of this work is to design and evaluate a UAV mesh network system, called LoRaUAV, following the architecture defined in Section 1.1. This main objective can be decomposed into the following objectives:

- Development of a decentralized topology control algorithm for the UAVs, allowing them to preserve connectivity with the BS in the CP despite the mobility of the ground nodes;
- Development of a simulation model of the system in ns-3;
- Development of a simulation model of an existing algorithm proposed in the literature to serve as term of comparison;
- Development of suitable application scenarios that are then implemented in the simulations;
- Performance evaluation of the system in ns-3 under selected QoS metrics.

## 1.3 Thesis Outline

The present thesis is organized as follows:

**Chapter 2 (Background)** Provides a description of LoRa and LoRaWAN, of the IEEE 802.11 protocol (WiFi) and the main Mobile Adhoc Network (MANET) routing protocols. A brief overview of UAVs is also provided at the end of the chapter;

**Chapter 3 (Related Work)** Review of the literature regarding LoRa and LoRaWAN and their performance, the most relevant network simulators and the LoRaWAN modules developed for them. Finally, some centralized and distributed UAV path planning algorithms are presented;

**Chapter 4 (LoRaUAV Design and Simulation)** Describes the main design choices of the LoRaUAV system and the developed UAV decentralized mobility algorithm. The simulation model developed for ns-3 is also presented;

**Chapter 5 (Results and Analysis)** Provides a critical analysis of the performance of the developed algorithms relative to the considered application scenario. This is done by resorting to selected QoS metrics collected by means of multiple computer simulations;

**Chapter 6 (Conclusion)** Evaluates the achieved results, considering the initial objectives. Some future directions to improve the work are also given.





# Chapter 2

## Background

This chapter provides a comprehensive overview of the technologies and protocols used in LoRaUAV. First, the main characteristics of LPWANs and the PHY and MAC layers of the LoRaWAN protocol are described. The IEEE 802.11, better known as WiFi, is then presented as a solution to interconnect UAVs in an ad hoc network. The main MANET routing protocols are also described. Finally, a brief overview of the main UAV characteristics is given and the composition of a UAV system is presented.

### 2.1 Low Power Wide Area Networks

In many applications, such as smart cities, environmental monitoring and smart agriculture, IoT devices are densely deployed and powered by traditional AA batteries. For this reason, they must be able to operate under strict power constraints using simple and cheap hardware components. Transmitting data over a wireless medium is a very costly operation in terms of power, due to many unpredictable environmental factors such as heat, humidity, wind and buildings and due to the inherent complexity of traditional wireless technologies. Protocols like Bluetooth Low Energy (BLE) and ZigBee try to address the power limitation problem for short-range communications, but for a vast number of IoT applications, where a high number of devices are deployed over large areas (e.g. a city), short range solutions are suboptimal and expensive due to the complex and dense network infrastructure that needs to be installed and maintained. Existing technologies, such as cellular networks, cover large areas, but are not power efficient. Moreover 2G, 3G and LTE are typically available in urban areas, but not always in suburban or rural areas. To address the aforementioned problems, a new range of protocols and technologies are being developed with the explicit purpose of enabling the operation of LPWANs. As the name suggests, LPWANs have the primary objective of providing low-power and low-cost connectivity over large geographical areas. The applications are multiple and diverse: smart cities, smart grids, smart metering, logistics, industrial monitoring, smart agriculture, etc. Recently, all the principal Standards Developing Organizations (SDOs) such as the Internet Engineering Task Force (IETF), the Institute of Electrical and Electronic Engineers (IEEE), the Third Generation Partnership Project (3GPP) and the European Telecommunications Standards Institute (ETSI) are intensifying their efforts in the standardization of LPWAN protocols. Moreover, numerous organizations and alliances work to promote specific LPWAN technologies. Examples of such alliances are the LoRa Alliance promoting the LoRaWAN protocol, the Dash7 Alliance promoting the Dash7 Alliance protocol and the Weightless-SIG promoting the Weightless protocol. Even if the existing LPWAN protocols and solutions are extremely diverse and address different niches in the same application segment, most of them use similar solutions to solve the same range of problems. The most relevant features are listed below:

- **Use of sub-GHz bands:** the vast majority of LPWAN protocols take advantage of the good properties of sub-GHz bands in terms of attenuation, multipath fading and obstacle penetration. Moreover, sub-GHz bands are currently less congested than other available bands, such as the overly used 2.4 GHz band. Most LPWAN technologies (e.g. LoRa and Sigfox) use unlicensed Industrial, Scientific and Medical (ISM) bands. Cellular operators are instead trying to exploit already owned licensed frequency bands. The use of unlicensed ISM bands reduces the operational costs of running the network, but most of the available bands are subject to limitations imposed by the legislative authorities of each country. As an example, the 868.0-868.6 MHz band, the one used by LoRa, is subject in Europe to a 1% duty cycle limitation [4], which means that, in a day, a single device can transmit for at most 14.4 minutes.
- **Use of Narrowband (NB) or Spread Spectrum (SS) modulations:** NB modulations have the advantage of reducing the adverse effects of noise on the transmitted signals. Using these techniques, receivers can successfully recover severely attenuated signals and achieve a sensitivity level of -130 dBm and below. As an example, the LoRa Semtech SX1276 chip achieves a sensitivity level of -148 dBm [5]. On the other side, the small bandwidth only allows low data rates in the order of few kbps or even a few hundred of bps if an Ultra Narrowband (UNB) modulation is used. In this last case the bandwidth can be as low as 100 Hz. Some protocols, such as LoRa, use variations of SS techniques to spread the narrowband signal over a wider bandwidth. The resulting signal has noise-like characteristics that makes it more resistant to jamming and eavesdropping and more resilient to interference. To overcome the less efficient use of bandwidth, SS is typically used in conjunction with orthogonal codes. This allows to multiple overlapping signals that are spread using different codes to be successfully recovered at the receiver.
- **Low power operation:** in a LPWAN, end devices are typically battery-powered and might need to operate using the same battery for many years. To reduce power consumption, LPWAN networks are organized in a simple and efficient star topology or a star-of-stars topology, where all the devices are directly connect to one or multiple gateways or base stations. Multi-hop protocols are typically not used since some network nodes, called hot-spots, might experience more traffic than the others. Moreover, hot-spots have to listen periodically for new messages coming from the other nodes thus reducing their overall battery lifetime. Duty-cycling the devices, i.e. alternating ON-OFF communication periods, is another frequently used technique to reduce the power consumption. For example, an uplink transmission can start only when new data is ready and a downlink reception can start only at a predefined scheduled time, usually following an uplink transmission. Simple random access ALOHA-like MAC protocols, requiring simple and cheap hardware and no synchronization, are typically used. Finally, to save on computational power, some complex operations, like detecting duplicated packets, are offloaded to the backend side of the network.
- **Low cost deployment:** installing, maintaining and operating an LPWAN network must be as cheap as possible. The price of LPWANs chips is usually in the order of a few dollars. This can be achieved by resorting to simple and sometimes inaccurate hardware components. The network backbone usually requires only few base stations to cover areas of tens of kilometres, thus additionally reducing the deployment cost of the network. Finally, most LPWAN solutions use unlicensed ISM bands thus avoiding wasting money for the acquisition of licenses.
- **Scalability:** support of a large number of devices is usually achieved by diversifying as much as possible in time, space and channels. Since devices use simple hardware, this diversification is usually implemented in the backend by using multi-channel multi-antenna base stations to parallelize the reception of signals. Some protocols also implement mechanisms to dynamically adapt

the data rate and to select the best channel, but given the strong link asymmetry of most LPWAN solutions, this mechanism, highly reliant on downlink information transmitted by the base station, is typically quite limited.

## 2.2 LoRaWAN

The LoRaWAN standard [6], promoted by the LoRa Alliance, defines the MAC layer and the network architecture of an LPWAN network that uses the proprietary LoRa modulation as a physical layer technology. The complete LoRaWAN stack is shown in Figure 2.1. An overview of the main LoRaWAN physical and MAC layer characteristics and features is given in the following sections.

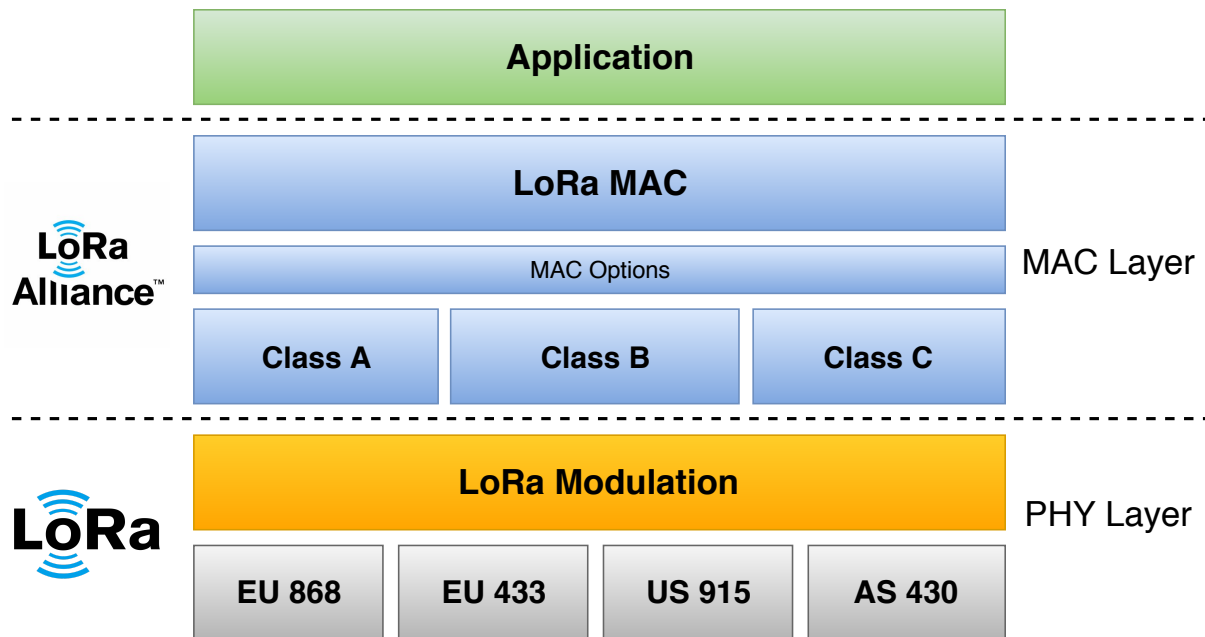


Figure 2.1: LoRaWAN stack.

### 2.2.1 Architecture and Topology

The overall architecture of LoRaWAN is shown in Figure 2.2. LoRaWAN implements a star-of-stars topology that interconnects five different network elements:

- **EDs:** they are typically low-power devices with sensing capabilities. EDs transmit or receive data using the LoRa radio channel. They can establish communication only with LoRa gateways. Once an ED is registered in the network, uplink messages can be received by multiple gateways at once;
- **GWs:** one or multiple gateways are responsible for collecting and aggregating the data coming from the EDs. Gateways are responsible for demodulating the LoRa uplink transmissions and relaying the uplink messages to a network server using a traditional IP-based backhaul network. Upon request, GWs can also send downlink messages to EDs;
- **Network Server (NS):** collects the packets coming from the GWs and routes them to the appropriate Application Server (AS). The NS is also responsible for sending and receiving MAC commands and discard duplicated packets;

- **Application Servers (ASs)**: implement the application specific logic. An AS aggregates, analyses and/or stores data and present it to the end users in a meaningful way. End users only interact with the services offered by an AS;
- **Join Server (JS)**: manages EDs activation. It derives and stores the keys used by the network for encrypting the communications.

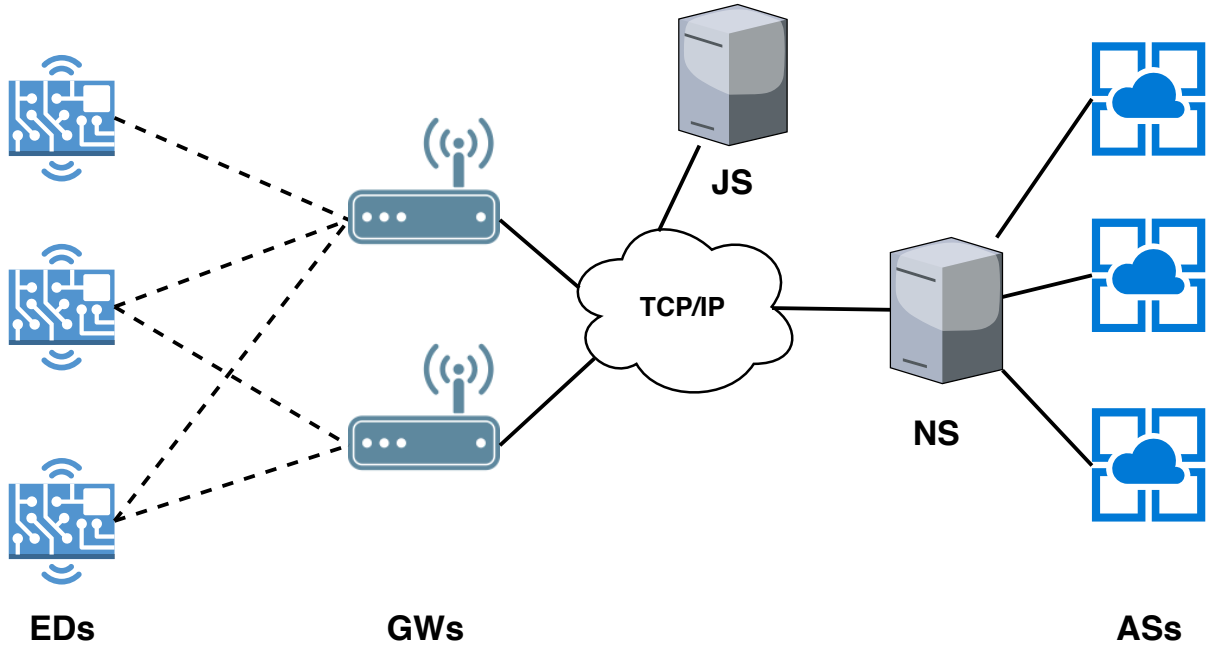


Figure 2.2: Topology of a LoRaWAN network.

## 2.2.2 LoRaWAN Physical Layer

LoRa is a proprietary modulation technique based on Chirp Spread Spectrum (CSS) originally developed by Cycleo and then acquired by Semtech in 2012. As many other LPWAN technologies, LoRa uses unlicensed sub-GHz bands (868 MHz in Europe, 915 MHz in North America, 433 MHz in Asia), therefore taking advantage of the good propagation properties of the spectrum. Since the LoRa modulation is proprietary, no official comprehensive description of the modulation exists as of today, but some attempts to reverse engineer the protocol have been made. The description of the LoRa physical layer given in the present work is partially based on the little information given in the official LoRaWAN standard [6] and on Semtech technical documents, such as [7]. More in depth details are taken from the reverse engineering efforts undertaken by Knight et al. [8] [9].

### 2.2.2.1 LoRa Modulation

The LoRa modulation is a variation of the CSS modulation technique, which is in turn a subcategory of Direct Sequence Spread Spectrum (DSSS). In DSSS data is spread over a wider bandwidth by multiplying the transmitted symbol by a Pseudorandom Noise (PN) sequence called chip sequence. The PN sequence is generated with a frequency higher than the symbol generation frequency. The result is a signal with a very low Signal to Noise Ratio (SNR) that makes it practically unrecognisable from background noise and very resilient to other RF noise sources. The PN sequence, known also by the receiver, is used to de-spread the signal and recover the original data. These properties make DSSS an

ideal solution for LPWANs. Unfortunately, DSSS requires a highly accurate and expensive clock source to synchronize the PN sequence at the transmitter and at the receiver. This makes traditional DSSS not particularly suited for low-cost end devices such as the ones deployed in a LPWAN. CSS is a variation of DSSS initially used for radar systems and for a long time ignored in communication systems, but recently revalued due to its robustness to a wide range of channel degradation mechanisms such as multipath fading, signal fading, Doppler shift and jamming interferers. In CSS, the chip sequences are substituted by chirps, i.e. signals of continuously increasing or decreasing frequency inside a defined frequency range. An increasing frequency chirp is called up-chirp, while a decreasing frequency chirp is called down-chirp. To simplify transmitter and receiver's hardware, up-chirps and down-chirps are typically linear in frequency as in the case of LoRa. In LoRa, a symbol is first chipped at a higher frequency that depends on the Spreading Factor (SF) and then modulated into a chirp. The chirp is characterized by the SF, a minimum frequency  $f_{min}$  and a maximum frequency  $f_{max}$ . An initial frequency  $f_0$  is assigned to the symbol. The chirp sweeps the frequencies from  $f_0$  to  $f_{max}$ , wrapping around from  $f_{max}$  to  $f_0$  when hitting the end of the available bandwidth  $B = f_{max} - f_{min}$ . A visualization of LoRa CSS can be seen in Figure 2.3. The period of the chirp is determined by Equation 2.1. The modulation data rate can then be computed as shown in Equation 2.2.

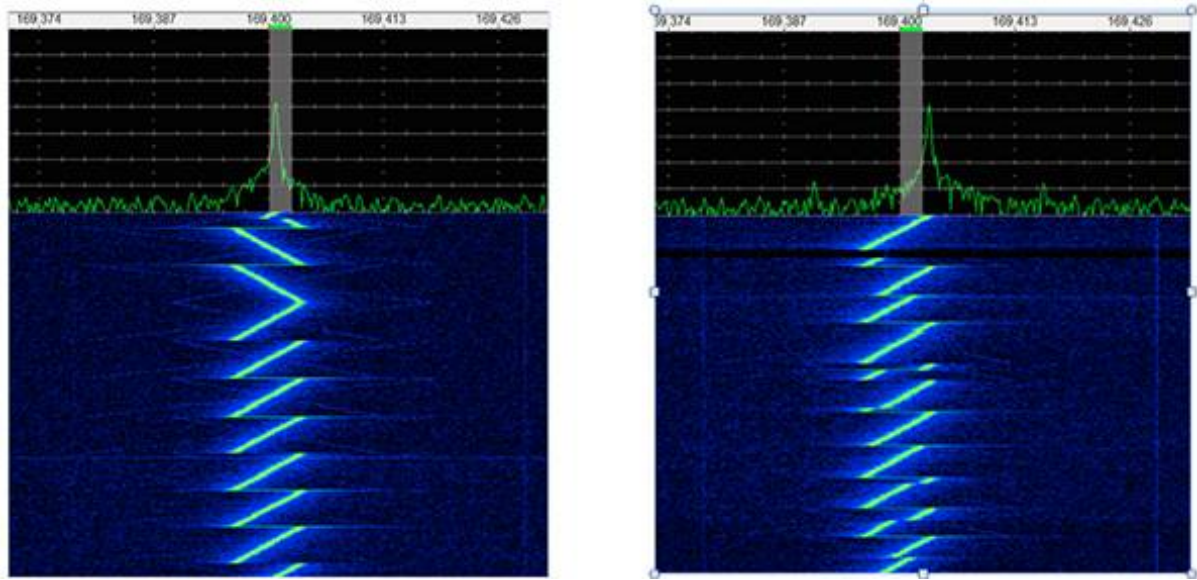


Figure 2.3: Visualisation of LoRa chirps (source: LinkLabs).

$$T_C = \frac{2^{SF}}{B} \text{ s} \quad (2.1)$$

$$R_b = SF * \frac{1}{T_C} \text{ bit/s} \quad (2.2)$$

From equations 2.1 and 2.2, it is possible to observe that a bigger SF corresponds to a longer chirp period and consequently a lower data rate. On the other side a higher SF makes the signal more resilient to noise and other RF interferers, thus allowing for a lower receiver sensitivity and therefore the possibility of successfully recovering the signal from higher distances. This robustness is counterbalanced by a higher probability of collision, since the signal takes more time to be transmitted. In LoRa the SF can vary between 7 and 12. If the coding rate is not considered, the typical LoRa data rates can be easily computed and are presented in Table 2.1. Another advantage of CSS is the Bandwidth-SF orthogonality between transmissions that use different chirp gradients. This characteristic allows to achieve a higher

throughput by using different Bandwidth-SF combinations.

SF	Bandwidth	Bitrate
12	125 kHz	366 bps
11	125 kHz	671 bps
10	125 kHz	1220 bps
9	125 kHz	2197 bps
8	125 kHz	3906 bps
7	125 kHz	6835 bps

Table 2.1: LoRa Data Rates.

### 2.2.2.2 LoRa Physical Layer Packet

The LoRa physical layer packet is composed of three fields:

- **Preamble:** used for the synchronization between the receiver and the transmitter. It's composed of a variable sequence of up-chirps having the same duration;
- **Start of Frame Delimiter (SFD):** indicates the beginning of a frame. It consists of 2 down-chirps;
- **Header and Data:** up-chirps of varying length. The data is extracted from the instantaneous frequency transitions of the chirps.

Some additional techniques are used to increase the robustness of the LoRa modulation:

- **Gray Indexing:** before being transmitted, the symbols are Gray indexed. Gray indexing is a way of encoding digits such that two consecutive digits have a change only in one bit position. This allows for more error tolerance. In the actual implementation, the symbols are read as gray-coded and de-grayed before transmission. Therefore the decoder needs to gray-code the symbols to recover the original symbols;
- **Data whitening:** this process induces randomness in the sequence of bits in order to reduce long sequences of repeating bits. This allows a better clock recovery by the receiver. The whitening is done by XORing the symbols with a pseudorandom sequence that is known by both the transmitter and the receiver;
- **Interleaving:** the initial ordered sequence of bits is scrambled using a scrambling sequence. The receiver uses the inverse operation to reconstruct the original bit sequence. This technique is used in conjunction with Forward Error Correction (FEC) to spread multiple consecutive errors over the whole frame. LoRa uses a variation of a diagonal interleaver;
- **FEC:** enables bits damaged during the transmission to be recovered at the receiver by introducing some redundancy in the transmitted data. LoRa implements a Hamming FEC that substitutes a 4-bit word with a codeword with length in the range [5-8] bits. A longer codeword provides additional protection at the cost of additional bits. When an Hamming (5,4) or (6,4) is used, only error detection is available. A Hamming (7,4) can correct a single bit error, while a Hamming (8,4) can correct a two bit error.

To improve reliability, code diversity is also exploited. Different spreading factors use semi-orthogonal codes, thus enabling the recovery of two signals overlapping in time and frequency if they have different SFs. This feature allows the network to achieve a higher overall throughput, thanks to the fact that colliding packets with different SFs can still be correctly demodulated.

### 2.2.3 End Devices (EDs) Operation Modes

LoRaWAN defines three classes of EDs with different characteristics and functionalities:

- **Class A EDs:** class A is the default operation mode of a LoRaWAN device. Class A EDs open two short downlink receive windows immediately after an uplink transmission, therefore making this mode of operation best suited for EDs that need a feedback from the network only after a successful transmission. The first receive window (RX1) is opened immediately after the uplink transmission. During RX1, the downlink frequency and data rate are set according to the same set of parameters used in uplink. In the second receive window (RX2), the downlink frequency and data rate are fixed and can be configured using specific MAC commands. Both RX1 and RX2 must stay open at least for the time needed for a GW to detect the downlink preamble and are only closed when the eventual downlink transmission is completed. Thanks to the short and limited downlink periods, Class A EDs are the most power-efficient devices in a LoRaWAN network;
- **Class B EDs:** these devices retain all the functionalities of class A EDs and furthermore, they can open additional receive windows at specific time instants agreed with the GWs. GWs must periodically send a beacon that specifies the exact time at which the additional receive windows must be opened. This feature allows for a more flexible use of the downlink channel, therefore representing a good compromise in terms of power consumption and chances of sending downlink messages;
- **Class C EDs:** EDs operating in this mode keep the receiving windows continuously open, except when the ED is transmitting a message in uplink. Class C operation mode is the least power-efficient class and should only be used for EDs with relaxed power constraints that benefit from low latency downlink messages.

All the EDs must implement Class A operation mode and may optionally implement one or all of the other available classes. An ED can switch to Class B or Class C operation modes by using specific MAC commands only after having successfully joined the LoRaWAN network.

### 2.2.4 MAC Frames and Commands

The structure of LoRaWAN MAC frames is shown in Figure 2.4. MAC frames are encapsulated in the physical layer packet payload and are formed by a MAC Header (MHDR), a MAC payload (MACPayload) and a Message Integrity Code (MIC), that is used to verify the integrity of the PHY payload. The MHDR contains, among other fields, a 3-bit MType field that specifies the message type. The available types are listed in Table 2.2. Data messages are used for transmitting MAC commands and/or application data. A distinction is made between data messages that require an acknowledgement from the receiving device (an ED or a GW) and data messages that do not require an explicit acknowledgement. Join-request, Join-Accept and Rejoin-request messages are used for Over The Air (OTA) activation as described in section 2.2.6.1. The MACPayload field is additionally divided in Frame Header (FHDR), Frame Port (FPort) and Frame Payload (FRMPayload). The FHDR contains a 4-byte Device Address (DevAddr), a 1-byte Frame Control (FCtrl) to request the Adaptive Data Rate (ADR) or an acknowledgement, a

2-byte Frame Counter (FCnt) and an optional Frame Options (FOpts) field containing up to 15 bytes of MAC commands. The mandatory 1-byte FPort field is used to specify the port used by the application. This field is used by the NS to route the message to the correct application in the AS. Other than data, LoRaWAN MAC frames can carry one or more MAC commands. MAC commands are simple commands exchanged exclusively between the NS and the EDs. MAC commands can be piggybacked to a data message using the FOpts field or they can be carried in the FRMPayload by specifying a FPort value of 0. Each command is defined by a 1-byte Command Identifier (CID) followed by zero or more command specific octets. MAC commands can be used to:

- Check the line connectivity;
- Limit the duty cycle of an ED;
- Set the receive parameters of the second receive window;
- Request status info to EDs;
- Set the delay between a transmission and the opening of the first receive window;
- Change ADR parameters;
- Request current time and date to the network.

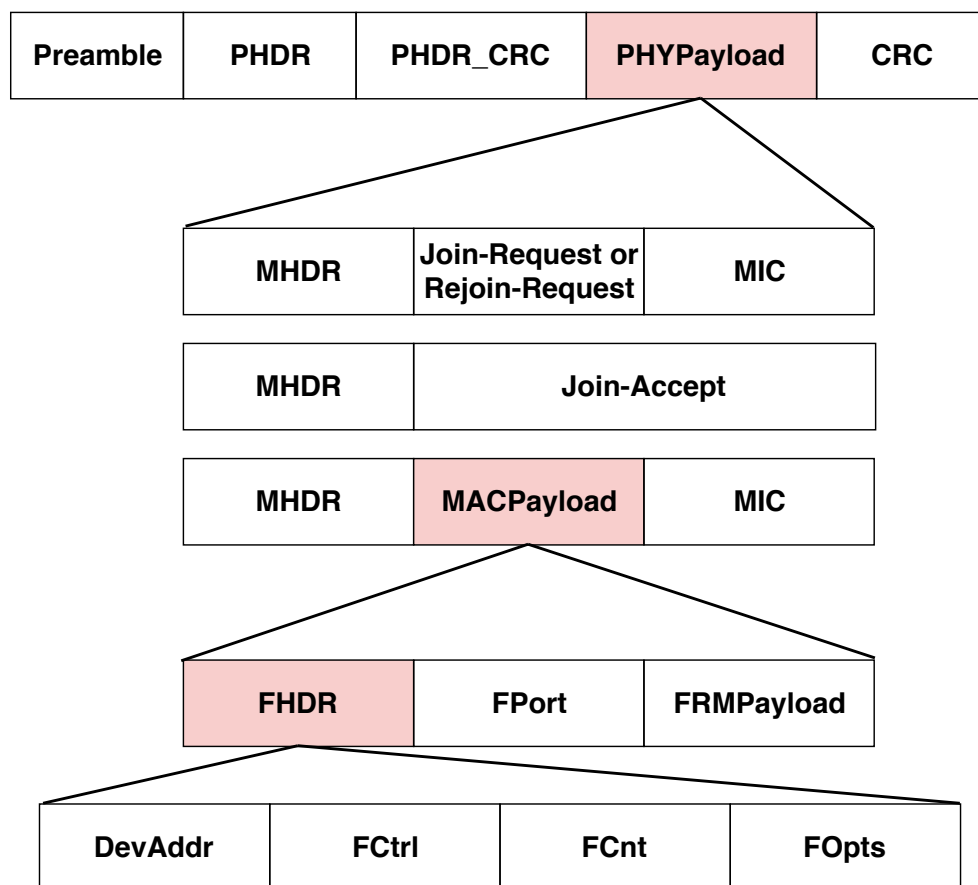


Figure 2.4: MAC frames structure.



MType	Description
000	Join-Request
001	Join-Accept
010	Unconfirmed Data Up
011	Unconfirmed Data Down
101	Confirmed Data Down
110	Rejoin-Request
111	Proprietary

Table 2.2: MAC message types (MTypes).

### 2.2.5 Adaptive Data Rate

LoRaWAN specifies an ADR mechanism that is used to optimize the data rate, transmission power and spectrum usage of EDs according to the quality of the link as measured by different metrics. When ADR is active, the SF of the modulation is adjusted in order to match the quality of the channel: it is increased when the SNR is low in order to increase the receiver sensitivity and decreased when the SNR is well above the threshold in order to minimize the power consumption and the time-on-air of the messages. An ADR request is always started by an ED by setting the ADR bit in the FCtrl field. This bit signals to the NS that the node is willing to accept ADR parameters variations based on the observations collected by the NS. The ADR should only be used by static nodes or by "smart" mobile nodes that are able to detect when they are parked for an extended period of time.

The ADR algorithm runs asynchronously in both the ED and the NS. The algorithm running on the ED is specified by the LoRa Alliance and consists in a progressive increment of the SF after a predefined number of downlink frames is not received. According to this algorithm, the SF can only be increased. To overcome this limitation, an ADR algorithm can be set up in the NS. It's worth noting that the specification doesn't specify an algorithm to be used for ADR in the NS, thus leaving the implementation details to the manufacturer of the network equipment or to the network operator. As an example, Semtech suggests a simple ADR algorithm: when ADR is requested, the NS starts collecting the uplink frames sent by the ED and uses the 20 most recent ones to compute

$$SNR_{margin} = \max_{20 \text{ frames}} (SNR) - SNR(DR) - margin\_db \quad (2.3)$$

where  $SNR(DR)$  is the minimum required SNR necessary to correctly demodulate the signal for the incoming SF and  $margin\_db$  is a device specific margin that is typically set to 10 dB.  $SNR_{margin}$  is used to compute the parameter

$$N_{step} = \lceil SNR_{margin}/3 \rceil \quad (2.4)$$

which can be positive or negative. If  $N_{step}$  is positive and the SF is bigger than the minimum allowed, the SF is decremented by  $N_{step}$ . If instead the SF is already at its minimum, the transmission power is decremented by  $3 \times N_{step}$  dB. Finally, if  $N_{step}$  is negative, the transmission power is increased by  $3 \times N_{step}$  dB.

### 2.2.6 Activation of EDs

A LoRaWAN network implementation is by default protected by some security mechanisms that ensure data integrity and confidentiality and prevent replay attacks. These three requirements are guaranteed

respectively by MICs, 128 bits Advanced Encryption Standard (AES) encryption and nonces. All security features require the presence of a set of session keys, that must be derived during the ED mandatory join procedure. Each ED must join the network using either OTA activation or Activation By Personalisation (ABP). The only difference between the two procedures is in the way in which the session keys are derived: in OTA activation, keys are generated on-demand, while in ABP keys are manually loaded. Only OTA activation is discussed in this section.

#### 2.2.6.1 OTA Activation

Before starting the join procedure, each ED is provided with four pieces of information that must be stored securely in the device:

- **Join EUI:** a 64 bit global identifier of the JS responsible for the join procedure;
- **Device EUI (DevEUI):** a 64 bit global identifier of the ED. The DevEUI is typically written also on a label on the back of the ED;
- **Network Key (NwkKey):** an AES-128 root key, assigned when the ED is manufactured, used to derive the Forwarding Network Session Integrity Key (FNwkSIntKey), the Serving Network Session Integrity Key (SNwkSIntKey) and the Network Session Encryption Key (NwkSEncKey);
- **Application Key (AppKey):** an AES-128 root key, assigned when the ED is manufactured, used to derive the Application Session Key (AppSKey).

The join procedure is always started by an ED by sending a Join-Request message to the JS containing the Join EUI, the DevEUI and a Device Nonce (DevNonce). The DevNonce is used to avoid the occurrence of replay attacks. The ED must store the last used DevNonce value and increment it at each subsequent Join-Request. The last value of the nonce is also permanently stored in the NS and every time a message is received, the DevNonce is compared with the stored value. If the received nonce is lower than the value stored in the NS, the message is discarded. If the JS accept the ED registration, a Join-Accept message is returned in downlink. The message contains the join nonce, the Network ID (NetID), the 32-bit Device Address (DevAddr) and a set of network configuration parameters. The join nonce is also in this case a progressive number, but it is used to derive the session keys. After the completion of the join procedure, the following session keys are established:

- **FNwkSIntKey:** used by the ED to compute the MIC of all uplink messages;
- **SNwkSIntKey:** used by the ED to verify the MIC of all downlink messages;
- **NwkSEncKey:** used by the ED and the NS to encrypt/decrypt uplink and downlink MAC commands;
- **AppSKey:** used by the ED and the AS to encrypt end-to-end the payload field of application-specific data messages. It's worth noting that even if confidentiality is guaranteed end-to-end, the integrity of messages is only guaranteed hop-by-hop, thus potentially allowing a malicious NS to tamper with the data;

## 2.3 The IEEE 802.11 Protocol

IEEE 802.11, best known as WiFi, is a widely used MAC and Physical (PHY) layer protocol suite for establishing wireless Local Area Networks (LANs) operating in the license-free 900 MHz, 2.4 GHz, 5.7 GHz and 60 GHz ISM bands. The first version of the standard was published in 1997 by the IEEE. Since then, many revisions of the protocol have been issued, most of them targeting mainly the PHY layer with the explicit purpose of increasing the maximum achievable throughput. The most relevant PHY amendments (IEEE 802.11a/b/g/n/ac) are discussed in this section. Also the MAC layer received some upgrades over the years in order to improve specific aspects, like QoS (IEEE 802.11e), security (IEEE 802.11i) and throughput (IEEE 802.11n). In this section, the original MAC specification and the most relevant modifications introduced with IEEE 802.11n are discussed.

### 2.3.1 IEEE 802.11 Architecture

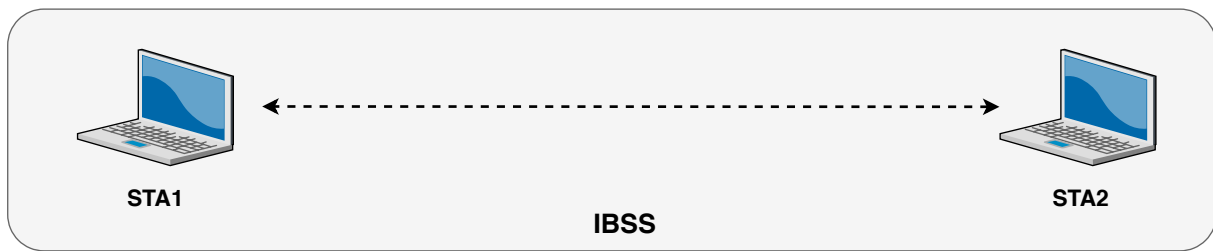
An IEEE 802.11 network can operate in one of two modes: ad-hoc mode or infrastructure mode. In ad-hoc mode each Station (STA) can only establish a direct communication with other STAs in their wireless coverage area, forming in this way an Independent Basic Service Set (IBSS). In infrastructure mode, STAs connect exclusively to a special distribution STA, called Access Point (AP), that must take care of coordinating and bridging the traffic among the connected STAs. An AP and its associated STAs form what is called an infrastructure Basic Service Set (BSS), characterized by a unique Basic Service Set ID (BSSID) (corresponding to the MAC of the AP) and a Service Set ID (SSID). To extend the coverage of the network, multiple infrastructure BSS can be interconnected through their APs by a Distribution System (DS). The set of infrastructure BSS having the same SSID and interconnected by a DS, form an Extended Service Set (ESS). Both ad hoc mode and infrastructure mode architectures are depicted in Figure 2.5.

### 2.3.2 IEEE 802.11 MAC Layer

IEEE 802.11 MAC layer is responsible for reliable data delivery, access control and security. The IEEE 802.11 MAC protocol, known as Distribution Foundation Wireless MAC (DFWMAC), provides both a mandatory decentralized, contention-based access protocol called Distributed Coordination Function (DCF) and an optional centralized, contention-free protocol called Point Coordination Function (PCF).

The DCF uses Carrier Sense Multiple Access (CSMA) to access the medium. According to this technique, each station listens to the medium and starts the transmission only if no other station is transmitting. If the medium is busy, the transmission is deferred to a later time after a backoff time chosen at random. CSMA with Collision Detection (CD) is typically used in wired media and it relies on the capability of all stations to receive a feedback in case a collision occurs. This assumption is not valid in wireless media, since not all stations are in the communication range of all the other stations participating in the network. Therefore a collision might occur at the receiver and go undetected at the transmitter. This issue is known as the hidden terminal problem. To overcome this complication, the DCF uses CSMA with Collision Avoidance (CA). As specified by this protocol, the transmitting station senses the medium for a fixed amount of time (Distributed Interframe Space (DIFS)) and if at the end of the period the medium is still free, the frame is sent, otherwise a back-off time is uniformly chosen in the interval  $[0, CW]$  and progressively decremented during the channel idle times. The value of  $CW$  starts at  $CW_{min}$  and it is doubled at every collision up to  $CW_{max}$ . When a data frame is successfully received, the receiving station, after a short fixed interval (Short Interframe Space (SIFS)), replies with an acknowledgement frame (ACK). If the ACK is not received before the expiration of a timeout, a collision

## AD HOC MODE



## INFRASTRUCTURE MODE

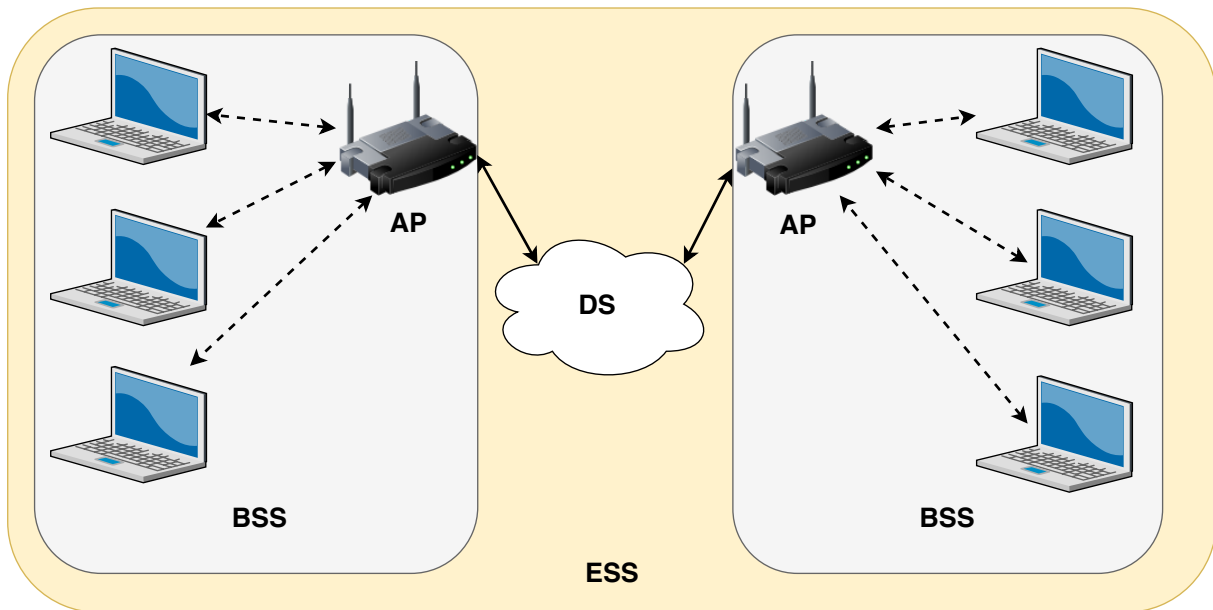


Figure 2.5: Architecture of an IEEE 802.11 network.

happened at the receiver and the sender needs to retransmit the frame. The medium access priority is determined by Interframe Spaces (IFSs) of varying length, such that a larger IFS implies a lower priority transmission. Since a SIFS is smaller than a DIFS, frames associated with SIFS, like ACKs, have a higher probability of taking control of the channel. To reduce even more the chances of collisions, the DCF implements an optional virtual carrier sense mechanism based on Request to Send (RTS) and Clear to Send (CTS) frames. After DIFS, a transmitter can send a RTS frame with the indication of the reservation parameters. If the sender is allowed to transmit, the receiver replies with a CTS. At this point, the exchange continues with the transmission of a data frame and the following ACK frame. Stations that are listening to the medium receives RTS or CTS frames containing the duration of the transmission and react by setting the Network Allocation Vector (NAV), which is an indicator of how much a station should defer the access to the medium. In this way, even if a station is not able to hear the transmission of another station, it can determine that someone is transmitting simply by detecting an outgoing CTS frame.

The PCF relies on a centralized polling mechanism and it is therefore available only to networks working in infrastructure mode, where the AP acts as polling coordinator. In order for PCF and DCF to coexist, time is divided in slots called *superframes*. The first part of a superframe is reserved for contention-free access. During this time, the polling coordinator gains control of the medium by waiting for the duration of a Point Interframe Space (PIFS) (which is shorter than a DIFS) and then sending a beacon that specifies the maximum duration of the PCF period. All the stations receiving the beacon

set their NAV according to the specified duration. Finally, the polling coordinator polls the stations in the polling list one by one.

The correct operation of a IEEE 802.11 network depends on the periodic broadcast of beacon frames containing information about the BSS, e.g. the timestamp, the beacon interval, the SSID, the capabilities of the network, the DSSS parameters and the PCF parameters. Moreover beacons are necessary for synchronizing the stations in the BSS. Beacons periodic broadcasting is typically performed by the AP, but in a IBSS, due to the lack of a central coordinator, such function is taken in turn by all the stations. Each station randomly sets a timer and a beacon is sent when the timer expires and no other beacon is received in the meanwhile.

With IEEE 802.11n, some MAC enhancements have been introduced to achieve higher throughputs. The main introduction is the aggregation of multiple data packets coming from the transport layer in one single frame. This technique reduces the impact of the inter frame time and of the MAC overhead. Two aggregation methods are defined: Aggregated MAC Service Data Unit (A-MSDU) and Aggregated MAC Protocol Data Unit (A-MPDU). An A-MSDU aggregates multiple MAC Service Data Units (MSDUs). Since a single MAC header is generated for each A-MSDU, all the aggregated frames must have the same source and destination address. Conversely, an A-MPDU can aggregate multiple MAC Protocol Data Units (MPDUs), each one with its own MAC header and possibly different source and destination addresses, and eventually even multiple A-MSDUs. This difference is particularly relevant in combination with another mechanism introduced by the protocol: block acknowledgements. According to this mechanism, upon reception of an A-MPDU, the receiver responds with an ACK containing only the indication of the correctly received MPDUs. Therefore only MPDUs with errors need to be retransmitted.

### 2.3.3 IEEE 802.11 PHY Layer

The IEEE 802.11 PHY layer evolved rapidly after the first 1997 draft. The first version of the standard, never actually implemented, specifies a physical layer based on DSSS and Frequency Hopping Spread Spectrum (FHSS), with a maximum bitrate of 2 Mbps. Since then, many amendments, improving the efficiency and the bitrate of the PHY layer have been issued. In this section the most widely adopted PHY layers are described.

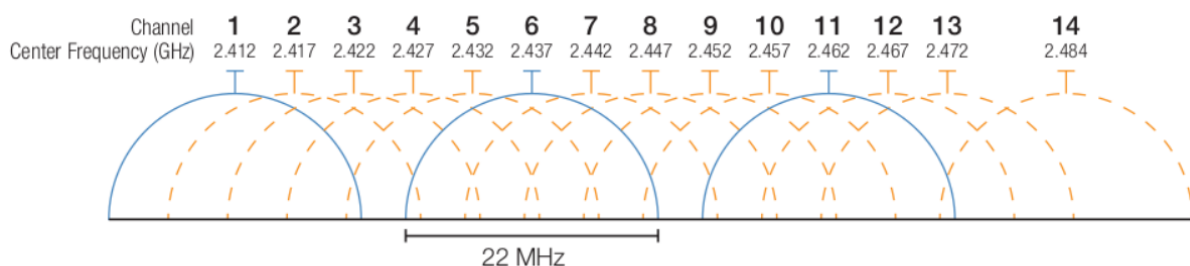


Figure 2.6: Channels in the 2.4 GHz band [10].

#### 2.3.3.1 IEEE 802.11b

IEEE 802.11b is the first revision that has been broadly implemented in consumer and enterprise devices. It shares with the original PHY implementation the use of the DSSS modulation and of the 2.4

GHz band, but with the advantage of an increased maximum raw bitrate that reaches 11 Mbit/s. This huge bitrate increment was made possible by substituting the Barker's code used in DSSS with the more efficient Complementary Code Keying, that allowed for more bits to be encoded in the spreading sequence. The most critical part of 802.11b is its use of the 2.4 GHz band also used by many household electric devices. Moreover, as shown in Figure 2.6, depending on regional regulations, only three or four 20 MHz-wide non-overlapping channels are available in the 2.4 GHz band, thus increasing the chance of interference.

#### **2.3.3.2 IEEE 802.11a**

IEEE 802.11a has been released almost at the same time of IEEE 802.11b, but has never been widely adopted, due to the popularity of IEEE 802.11b and the high cost of producing hardware operating in the 5 GHz band. This revision substitutes the original DSSS modulation with a more efficient Orthogonal Frequency Division Multiplexing (OFDM) with 52 sub-carriers, that allows a maximum bitrate of 54 Mbit/s. The biggest advantage of IEEE 802.11a over IEEE 802.11b is the use of the 5 GHz band, which is far less congested than the 2.4 GHz band. Moreover, the presence of 24 non-overlapping channels guarantees a significant increase in capacity. The only disadvantage is the slightly reduced range due to the worse propagation characteristics of the signal at higher frequencies, but this is counterbalanced by the good properties of OFDM in terms of attenuation of the effects of multipath fading.

#### **2.3.3.3 802.11g**

IEEE 802.11g has been issued in 2003 and immediately became very popular, thanks to the rapid increase in demand of higher throughput wireless networks. To maintain the compatibility with IEEE 802.11b, this new revision operates in the 2.4 GHz band and offers some additional mechanisms to ensure interoperability between the two protocols. This comes at the expense of the maximum achievable throughput in mixed 802.11b/g networks. IEEE 802.11g uses the same OFDM modulation used by IEEE 802.11a, thus achieving a maximum bitrate of 54 Mbit/s. For backward compatibility, lower data rates with DSSS and Complementary Code Keying are supported.

#### **2.3.3.4 802.11n**

IEEE 802.11n introduces enhancements to both the PHY and MAC layer that allow to reach a maximum bitrate of 600 Mbit/s. This is made possible by taking advantage of wider channels (40 MHz), shorter guard bands (400 ns) and especially multiple antennas (MIMO) that can differentiate up to four simultaneous spatial streams in the same channel. Beamforming is also supported. The standard can operate on both the 2.4 GHz and the 5 GHz bands, but protection mechanisms need to be established in the 2.4 GHz band to avoid interference with legacy devices. In fact, using a 40 MHz channel results in the occupation of 9 of the 14 available channels of the 2.4 GHz band. The standard support various combinations of modulation schemes (up to 64 QAM) and coding rates (up to 5/6), that allow to adapt the data rates to the condition of the channel and of the environment.

#### **2.3.3.5 802.11ac**

IEEE 802.11ac builds on the features of IEEE 802.11n and introduces enhancements with the explicit purpose of supporting very high throughput applications, such as multimedia applications. The standard now supports the bonding of four channels (80 MHz) or eight channels (160 MHz), a denser modulation (256 QAM) and even more MIMO streams (up to 8 simultaneous spatial streams). MIMO is also

extended to support Multiple Users MIMO (MU-MIMO), a technology that allows to send multiple frames to multiple clients at the same time over the same spectrum. All these features are expected to produce a maximum bitrate of 3.47 Gbit/s, but the first wave of products is expected to achieve lower bitrates (1.3 Gbit/s). IEEE 802.11ac only operates in the 5 GHz band, thus dropping the backward compatibility with 802.11b/g devices, but integrating almost seamlessly 802.11a/n devices with just small adaptations.

## 2.4 MANET Routing Protocols

A MANET is a network exclusively composed of wireless nodes, with a certain degree of mobility, operating in ad hoc mode. In ad hoc mode a link is established between a pair of nodes only if those nodes are in each other's range, but, in many circumstances, it is desirable to communicate with another node that is not directly in range, but is in the range of another nearby node. For this to happen, a routing algorithm is necessary. Traditional routing algorithms are not suited for mobile networks, due to the high bandwidth required to flood the updates. This condition is worsened by the frequent topology changes that might trigger even more updates. Moreover, wireless networks are not symmetric, i.e. a link may not be bi-directional, and traditional routing algorithms don't cope well given this assumption. MANET routing protocols have been developed to solve the aforementioned problems. These protocols can be divided in two broad categories: reactive and proactive. Reactive protocols create a route only when it's needed. This typically coincides with a better usage of the bandwidth and more energy efficiency at the expense of the delay needed to establish the route. Proactive protocols keep an updated routing table to any destination of the network, therefore routes are immediately available, but a bandwidth-intensive flood of updates needs to be done periodically to keep the tables up to date. In this section, two reactive routing protocols, Dynamic Source Routing (DSR) and Ad-hoc On-demand Distance Vector (AODV), and two proactive routing protocols, Destination-Sequenced Distance Vector (DSDV) and Optimized Link State Routing Protocol (OLSR), are briefly analysed. A geographical routing protocol, Greedy Perimeter Stateless Routing (GPSR), is also described.

### 2.4.1 Proactive Routing: DSDV and OLSR

DSDV [11] and OLSR [12] are proactive routing protocols and as such they keep a routing table to all known destinations in the network. Each entry contains at least the destination address and the next hop. DSDV is the oldest of the two and it uses a modification of the traditional Bellman-Ford algorithm for distance vector routing. The main addition to the algorithm are the sequence numbers associated with each update. This mechanism is fundamental, since it avoids the formation of loops. Each node has to send periodic updates, tagged with a sequence number bigger than the previous issued update. Such updates can have two forms: full dumps or incremental updates. Full dumps are mandatory and contain all the available routing information, while incremental updates are smaller optional updates that are triggered by changes in the network topology, e.g. a broken link. DSDV suffers from many problems. First of all, the protocol doesn't scale well as the nodes in the network increase. Control traffic is proportional to the number of nodes and there are no mechanisms to limit the number of update packets. Moreover, the protocol is not suited for highly mobile networks, since each topology change triggers an update with higher sequence number that needs time to propagate, thus increasing the risk of having stale routes. In order to reduce routes fluctuations, DSDV waits for a certain amount of time for the route to settle before propagating the updates, thus increasing even more the delay.

OLSR tries to tackle some of the problems that affect DSDV. Contrary to DSDV, OLSR is a link state protocol, therefore every node is aware of the entire topology of the network and not just of their direct

neighbourhood. The most promising idea of OLSR are *multipoint relays (MPRs)*: a special subset of nodes responsible for the flooding of the link state updates. MPRs of a node N are defined as the set of nodes that have a bidirectional link to all the two-hop neighbours of N. MPRs are computed based on HELLO packets exchanged by neighbouring nodes: each node periodically broadcast to its one-hop neighbours a HELLO packet containing a list of all the neighbours with a valid bidirectional link and a list of all heard neighbours which are still not confirmed as having a bidirectional link. It's worth to point out that HELLO packets are not flooded, but only sent to direct neighbours. After this message exchange, each node is aware of its two-hop neighbours and the state of the links that connect them, so the selection of the MPR set is straightforward. Each MPR keeps also a list of *MPR selectors*, i.e. the set of nodes that chose that node as their MPR. MPRs periodically generate *Topology Control (TC)* messages, containing the addresses of the MPR selectors, and flood them in the network. The information contained in TC messages is used at each node to generate a complete graph of the network and to determine the shortest path to any destination. Thanks to the MPR mechanism, OLSR is particularly suited for large and dense networks. In fact, the bigger the network, the higher are the chances of having less MPRs and consequently less band-consuming updates that need to be flooded. Thanks to these good properties, OLSR is one of the most promising MANET routing protocols.

## 2.4.2 Reactive Routing: DSR and AODV

DSR [13] and AODV [14] are two reactive routing protocols with a similar behaviour, but a different approach on how routes are established and maintained. In fact, DSR is based on source routing and full paths are therefore stored in the source node, while AODV is a distance vector protocol and the source has only a local knowledge of the network and the exact path taken by a packet is unknown. DSR operates in two phases: route discovery and route maintenance. The route discovery phase is triggered whenever a node needs to transmit a packet to a given destination, but a route is not available in the cache, either because the route is expired or invalidated or because it is the first time a packet is sent to the destination. As a result, a route request packet, uniquely identified by the couple  $\langle \text{originator address}, \text{request id} \rangle$ , is flooded in the network. Upon reception of a request, each node that is not the intended destination appends his own address and rebroadcast the packet. When a route request eventually reaches the destination, a route reply, containing all the hops from source to destination, is sent to the originator in unicast by reversing the address chain specified in the route request. To avoid the propagation of stale route requests, the only requests that are considered are the ones that have a *request id* that is bigger than any other previously received request with the same *originator*. A route request is also discarded when the host address is already present in the address chain, thus avoiding the formation of loops. While a route is active, the route maintenance procedure monitors the route and check for any routing errors. When a problem is detected, e.g. a link is no longer working, a route error message is generated and sent to the originator, that must truncate all the cached routes containing that link and eventually start a new route discovery. To optimize the route discovery process, an intermediate node that already knows a route to the destination might respond immediately with a route reply. Intermediate nodes can also cache partial or complete routes heard from route replies or route requests.

One of the main problems of DSR is the significant message overhead that comes from fact that each route request need to log the addresses of all the nodes along the path. AODV solves this problem by setting up routing tables in each intermediate node. Upon reception of a route request, a node creates an entry in the routing table specifying the originator address and the next hop to the originator, which is the address of the host that sent the received request. In this way, a route request packet only needs to contain the originator address and the destination address and not the list of all nodes in the



path. When the route request reaches the intended destination, a route reply is sent back in unicast mode by following the reverse path set up during the propagation of the route request. AODV uses procedures very similar to the ones implemented in DSR for discarding stale route requests, caching routes and maintain active routes. To avoid the formation of loops, destination sequence numbers, uniquely identifying a route to a given destination, are used. With this mechanism, only the routes with higher destination sequence numbers advertised by other nodes are accepted as valid updates to the previously existing routes. Contrary to DSR, each origin/destination pair has only a single active route at any given time.

### 2.4.3 Geographical Routing: GPSR

GPSR [15] is a routing protocol that uses geographical information to route packets to the intended destination. To achieve scalability, GPSR makes routing decisions that are based only on the position of nearby nodes. Positions are easily acquired by listening for beacons sent by all the nodes in the network at regular intervals. If no beacon is received from a node within a given time out interval, the node is assumed to be unreachable and it is therefore deleted from the list of neighbours. The routing decision is based on two approaches: *Greedy forwarding* and *Perimeter forwarding*. Greedy forwarding is always the behaviour with highest priority and it consists in forwarding the packet to the neighbouring node that is closer to the destination with respect to the current node position. The process is repeated until the packet reaches the destination. Unfortunately, greedy forwarding is not always able to make a decision. This happens, as it is shown in Figure 2.7, when the forwarding node is closer to the destination than any of its neighbours, but the destination is out of range.

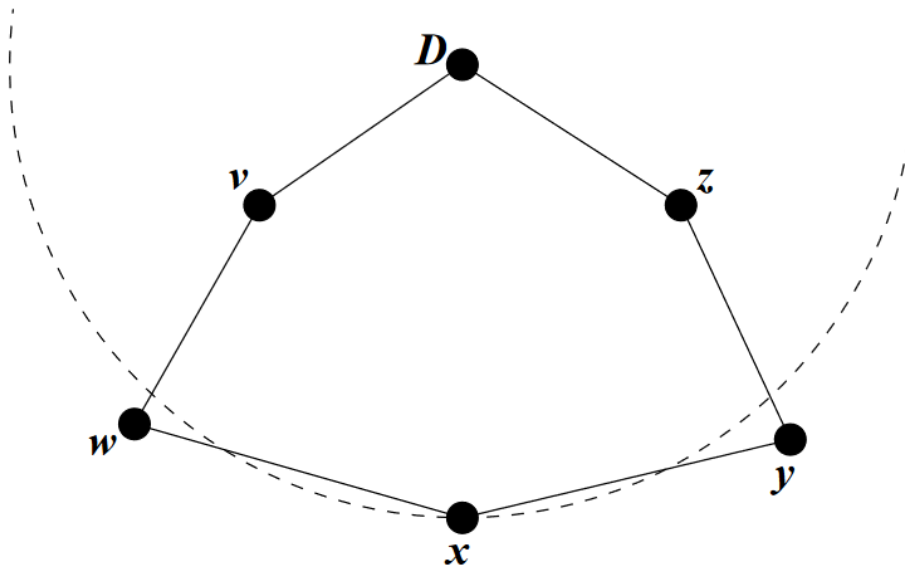


Figure 2.7: Failure of GPSR greedy forwarding behaviour [15].

Perimeter forwarding solves this problem by using the right hand rule for traversing a graph. This rule states that a packet reaching node  $x$  from node  $y$  must be forwarded through the first counterclockwise edge with respect to edge  $(x, y)$ . The sequence of edges traversed using the right hand rule is called *perimeter*. To speed up the decision, perimeters can be mapped beforehand and used when the greedy rule fails. One disadvantage of the right hand rule is that it only applies to planar graphs, i.e. graphs without crossing edges. Specific algorithms must therefore be implemented to remove crossing edges

using only local information without causing the disconnection of the graph. An algorithm for this purpose is presented in detail in [15].

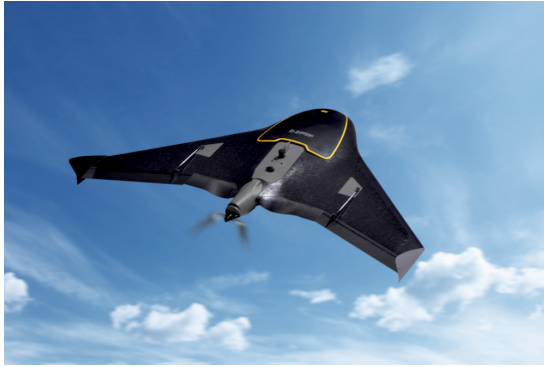
## **2.5 Unmanned Aerial Vehicles**

UAV is a general term used to identify all aircraft that are able to fly without the presence of an on-board human pilot. Depending on the level of achieved autonomy, two additional categories of UAVs with a lower degree of autonomy can be defined: Remote Piloted Vehicles (RPVs) and drones. As the name suggests, RPVs are not fully autonomous and need a complete or partial control of a remote pilot using some sort of remote control system (a joystick for example). Despite being used as a synonym of UAV, drone is a term that should be more appropriately used to indicate "dumb" UAVs only capable of fairly limited actions and with a just as limited set of functionalities (e.g. target drones). Since the use of drone as a synonym of UAV is very common in both the academic literature and in the everyday language, in the rest of the present work the two terms are used interchangeably.

### **2.5.1 UAVs classification**

UAVs are classified depending on a wide range of characteristics: size, range, aerial platform, degree of autonomy, engine type and fuel source. All these features are analysed in this section.

The first distinction that can be made is based on the type of aerial platform that is used to keep UAVs up in the air. Drones available today belong to one of two main categories: fixed-wing drones and multi-rotor drones. These two types of drone are shown in Figure 2.8. Fixed-wing drones are equipped with fixed, static wings and they use a forward propulsion to move. The way in which they behave is similar to a traditional airplane. Multi-rotor drones, on the other hand, are more similar to helicopters, since they use one or more rotary wings to generate lift. The most common multi-rotor drones use four rotors and are therefore called quadrotors. Depending on the application, one or the other configuration may be more suitable: fixed-wing drones are typically faster than multi-rotors, but they require more space to take off and they can not hover in a fixed position. Multi-rotor drones can overcome these limitations, but are much slower than fixed-wing models. UAVs can be further divided depending on their size and weight. This characteristic reflects the ability of the drone to carry payloads, with bigger drones being able to carry heavy payloads and small drones only able to carry lightweight payloads. Very small UAVs have dimensions of 30-50 cm and are usually multi-rotors. Small UAVs are slightly larger (up to 1-2 meters along one dimension) and are usually fixed-wing. The same applies to medium and large UAVs with the only difference that they can be as big or bigger than a light aircraft. They can carry very heavy payloads (approximately 200 kg) and are generally only used for military applications. UAV engines can be powered using many different fuel sources: traditional aircraft fuel (kerosene or gasoline), fuel cells, batteries and solar cells. With the exception of few models, the vast majority of drones are powered either by aircraft fuel or batteries. Drones using aircraft fuel as their power source are able to fly for longer times, but due to the weight of the fuel and the size of the engine, they have bigger dimensions. The most common engines, in order of stability level they can provide, are the four-cycle/two-cycle internal combustion engine, the rotary engine and the gas turbines. Battery-powered drones, equipped with a simple electric motor, are cheap and lightweight, but their flight autonomy is limited to only a few minutes, thus drastically restricting their application in many context.



(a) Fixed-wing drone.



(b) Multi-rotor drone.

Figure 2.8: Two of the most common drone configurations: fixed-wing and the multi-rotor.

## 2.5.2 UAV system

The operation of one or more UAVs requires the deployment of a suitable infrastructure, that is necessary to manage and control all the aspects of the mission. A typical UAV system, as shown in Figure 2.9, is at least composed of the following elements:

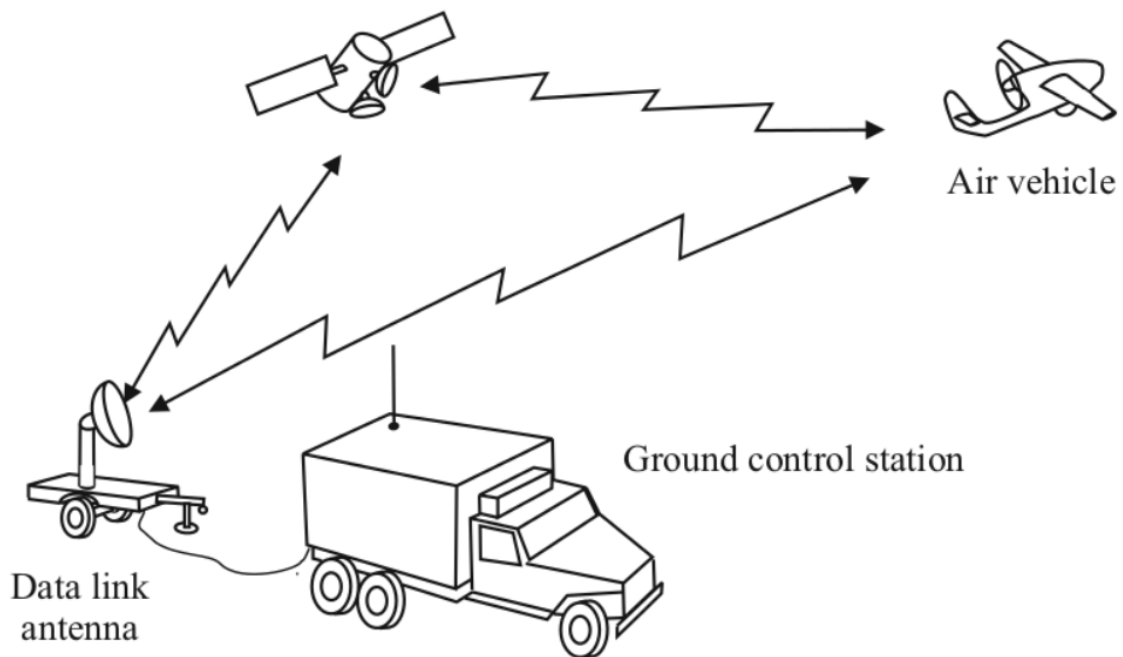


Figure 2.9: A general UAV system [16].

- **Air vehicles:** they are the core part of the system. In order to fly with a high degree of autonomy, air vehicles need to be equipped with some sort of controller that regulate, for example, the altitude and the airspeed. This control is usually achieved by using a feedback or closed-loop controller in conjunction with a set of sensors that typically include an altimeter, a speed meter and some sort of positioning device (a GPS for example). The measurements of the sensors are used by the controller to compute the right actuation to adjust the roll, pitch and/or yaw of the vehicle. Typically the control is limited to one axis (roll) or two axis (roll + pitch) to limit the controller complexity. Almost all UAVs are nowadays equipped with a GPS, that can be used to complement the measurements of the other sensors and for accurate path planning;

- **Payload:** drones' main mission is to carry a payload. Typical payloads include HD cameras, communication gateways, infrared sensors and thermal sensors. The payload greatly influences the characteristics of the drone that carries it. For example, a heavy payload may require a bigger and more powerful drone;
- **Ground Control Station (GCS):** it is usually a vehicle that can transport the drones to the place where they need to operate. The GCS is also equipped with all the necessary instruments to process and display the data received from the drones payloads. If the UAV system is small, the GCS can be reduced to the size of a backpack that can be carried around by the drones operator. A GCS can also be located at a fixed position, for example in a building;
- **Data link antennas:** antennas are used to establish a two way communication with the air vehicles. The channel is typically split in two sub channels: a low rate channel for transmitting control commands and receiving feedback from the UAVs and a high rate channel for receiving sensors data (e.g. the video stream). LOS is usually required for the communication to happen;
- **Ground Support Equipment (GSE):** this unit is used to maintain the UAVs and may contain spare parts and additional fuel or batteries to extend the UAVs on-air time. The GSE unit is optional, but its use is generally required, especially in systems with many drones and for missions that require extended operation times.

# Chapter 3

## Related Work

The objective of the present work is to assess the performances of a system that provides situational awareness to rescuers during a wildfire. The envisioned system is based on two main ideas: the deployment of UAVs to dynamically provide network coverage to ground nodes and the usage of the LoRaWAN protocol as Air-to-Ground (AtG) communication protocol. In order to establish a background of notions that can support and contextualize the contributions given in the present thesis, some related work is presented in this chapter. The main studies and results regarding LoRa and LoRaWAN and their performance are first described. The main network simulators and the available LoRaWAN simulation modules, with their characteristics, advantages and drawbacks are discussed next. Finally, the most relevant centralized and distributed UAV swarm mobility models are presented.

### 3.1 Performance Assessments of LoRa and LoRaWAN

LoRa is a relatively recent technology. The first patent for the LoRa modulation was issued in 2008, while the first version of the LoRaWAN standard was released only in June 2015. Even if it is a young technology, LoRaWAN got the attention of both the academy and the general public, especially thanks to the availability in the market of cheap LoRaWAN EDs and GWs. The Things Network (TTN) is a notable example of a global LoRaWAN network that is continuously expanded and supported by the members of the community. As of today, TTN consists of more than 3500 active gateways deployed all around the world and gives the opportunity to the users to deploy their devices and build their applications. LoRaWAN also got the attention of many researchers. As a result, a vast literature focusing on various aspects of the technology has been produced in the last couple of years.

Petajajarvi et al. [17] have conducted empirical measures to determine the maximum range of LoRa in a small rural city in Finland. The results show that, by using the maximum SF and a bandwidth of 125 kHz in the 868 MHz band, the signal can reach the GW from 15 km of distance on ground and 30 km on water, with increments of the Packet Drop Rate (PDR) proportional to the distance. Jörke et al. [18] have performed a similar analysis in a medium density urban area, noting in this case, that the maximum range under the same conditions described in [17] is only 5.8 km, mainly due to the presence of obstacles in the LOS. In [19], the reception in presence of indoor EDs is also evaluated: the authors report that at a distance of 1.2 km from the gateway, the PDR is already between 60% and 90%. The conclusion of the authors is that the presence of LOS is particularly important and that LoRa is not suited for indoor environments if long range and low PDR are required. The importance of LOS is also outlined by Iova et al. [20]. In this case the experiments were conducted in a forest environment in presence of numerous trees between the transmitter and the receiver. In the experiment, the transmitter was moved

further and further away from the receiver and at around 90-100 m the signal was completely lost. The authors conclude that vegetation has a huge impact on LoRa signal propagation, in fact dropping the maximum achievable range by an order of magnitude.

The use of CSS, a variation of DSSS, makes the LoRa modulation very resilient to noise. This assumption is verified by Angrisani et al. [21]. In the experiment, different levels of white Gaussian noise are applied to the signal starting with a minimum SNR of -10 dBm. As expected, the use of a smaller bandwidth and, more significantly, the use of a higher SF improves the performance of the modulation. The coding rate, on the other hand, has a less significant impact on the performance in noisy environments.

One of the main requirements of an LPWAN is its ability to scale to hundreds or even thousands of devices. Early attempts of assessing the scalability of LoRaWAN were performed by resorting to mathematical models. Mikhaylov et al. [22] assessed the scalability of a LoRa network in presence of a single GW. The authors modelled the LoRaWAN MAC as pure ALOHA and reached the conclusion that LoRaWAN can potentially scale to millions of devices in case of EDs generating only intermittent uplink traffic, but most of the devices need to be positioned very close to the GW. This means that only a limited set of EDs can take advantage of the range boost given by higher SFs. The authors also point out that huge scalability issues are present in downlink, since GWs are still subject to duty cycle limitations and acknowledging every uplink packet of a large network is impossible. Augustin et al. [23] considered a single LoRaWAN logical channel, characterized by the couple  $\langle \text{frequency}, SF \rangle$ , and performed some simulations assuming Poisson packet arrivals, packet payloads uniformly distributed between 1 and 51 bytes and a 1% duty cycle. The authors conclude that LoRaWAN behaves almost exactly like ALOHA, in fact the maximum capacity usage is 18% with a load factor of 0.48. Bor et al. [24] developed LoRaSim, a collision simulator that supports multiple GWs and models also the capture effect, i.e. the effect that cause a signal that is received with more power to be correctly decoded even if it is overlapping with other interfering signals. A more detailed description of the simulator features is given in Section 3.2. The authors used LoRaSim to conduct a series of experiments consisting in measuring the Data Extraction Rate (DER), defined as the ratio between the received packets and the transmitted packets, as a function of the number of deployed EDs. In the first experiment the DER is measured for the three different transmitter configurations shown in Table 3.1.

Parameter	$SN^1$	$SN^2$	$SN^3$
Tx Power (dBm)	14	14	14
Center frequency (MHz)	868	868	868
SF	12	6	12
Bandwidth (kHz)	125	500	125
Coding Rate	4/8	4/5	4/5
Packet arrival rate	$1 \times 10^{-6}$	$1 \times 10^{-6}$	$1 \times 10^{-6}$

Table 3.1: EDs PHY configurations used in the experiments performed in [24].

As shown in Figure 3.1, the results confirm that LoRaWAN behaves exactly like ALOHA with the simple collision assumption of [23]. However, with the more realistic collision model of LoRaSim, LoRaWAN performs better than ALOHA in typical SF 12 configurations. The authors suggest that ALOHA can be used as a good lower bound estimation of the LoRaWAN behaviour. A second experiment, using  $SN^1$  and a varying number of GWs produced the results depicted in Figure 3.2. As it is noticeable, the presence of multiple GWs can significantly increase the number of supported EDs having  $DER \geq 0.9$ .

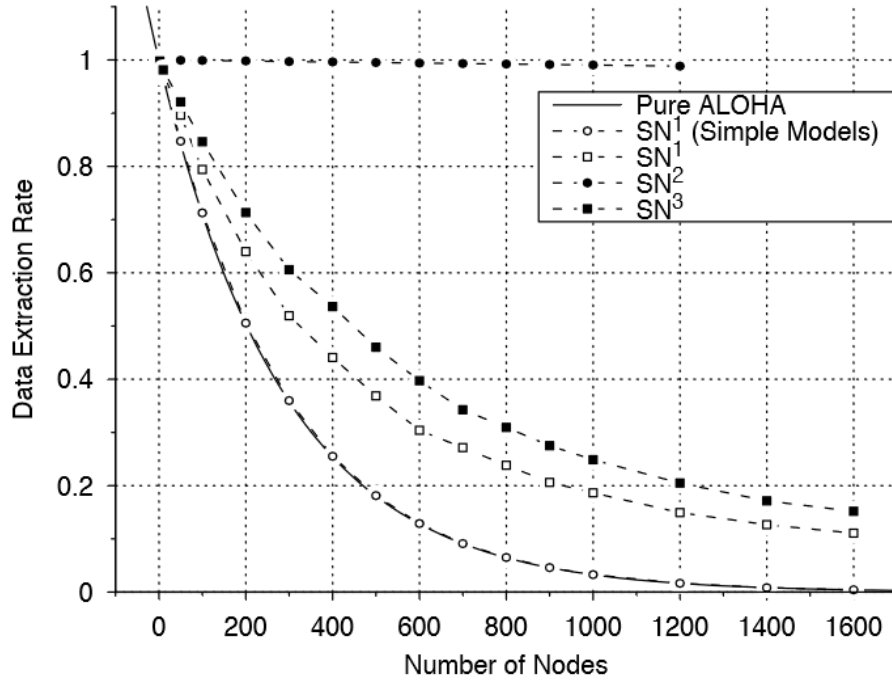


Figure 3.1: Fraction of correctly received packets with one GW and a varying number of EDs having different PHY configurations [24].

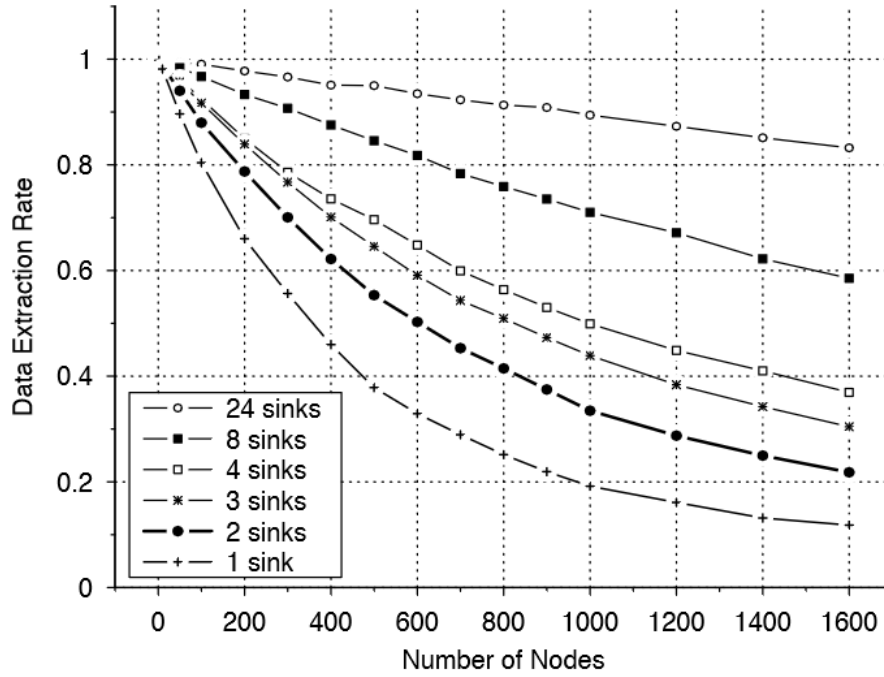


Figure 3.2: Fraction of correctly received packets with multiple GWs and a varying number of EDs [24].

## 3.2 Network Simulators and LoRaWAN Simulation Modules

In order to study different aspects of a complex network or system, researchers have always resorted to one of three methods: theoretical analysis, simulations or experiments using prototypes. Simulations are in the middle ground with respect to the other methods, since they offer a theoretically accurate and cost-effective environment to test how complex systems behave and interact with one another. LoRa and LoRaWAN are recent technologies and as such the choice of simulation modules is very limited. Nonetheless, a bunch of notable implementations are available, covering the most relevant aspects of the protocols. In this section, the available LoRaWAN simulation modules and the network simulators implementing them are described.

### 3.2.0.1 ns-3

One of the most well known open source general purpose network simulators is ns-3 [25]. ns-3, acronym that stands for network simulator 3, is a discrete event simulator built in C++. Its principal purpose is the simulation of networks based on the IP protocol, even if multiple modules are available to simulate also non-IP networks. ns3 is highly modular and allows for an easy integration of different network models that can be linked together in a very easy and flexible way. This modularity allows the usage of already existing well tested and well document models to build new functionalities or simulation scripts. In terms of performance, as it is pointed out by Weingartner et al. [26], ns-3 is one of the best simulators in terms of speed and memory usage. ns-3 is a discrete event simulator. As such, all the events that may arise in the network (e.g. sending/reception of a packet) are previously determined and scheduled for execution at a specific time. When the simulation start, the scheduled events start being executed in order until the last event is completed. During the simulation, new events can be dynamically added to the scheduler. In ns-3 it's possible to distinguish the following basic components:

- *Nodes*: they are abstractions of the physical devices that form the network (e.g. mobile phones, PCs, sensors) and they are instances of the *Node* class. When they are instantiated, they are just empty containers in which it's possible to install applications, net devices, protocols, etc.
- *Application*: abstractions of the applications running on the network devices. They give the ability to the devices to perform particular tasks, for example sending or receiving UDP packets. Every application is an instance of the *Application* class that has been specialized by the author of the application in order to perform a certain function;
- *Channels*: a network needs some sort of connection among the devices. The connection can be physical (e.g. an Ethernet cable) or wireless. In ns-3 all these connections are modelled with a specialization of the *Channel* class;
- *Net Devices*: in the simulator logic, they substitute Network Interface Cards (NIC) and the drivers necessary to operate the network. They allow a node to communicate to another node through a channel. Every net device is a specialization of the *NetDevice* class;
- *Helpers*: tools that simplify the process of creating a network by providing specific methods to create set of nodes, install net devices and channels and configure the protocol stack.

ns-3 implements also some systems to collect and analyse the data generated by the simulation:

- *Logging module*: this module allows to display messages in the terminal during the execution of the simulation. The logging module is organized in levels, ordered based on an increasing level of



verbosity. The logging module can be enabled for all the components of the simulation or only for some modules;

- *Tracing System*: the logging module is useful during the debugging phase, but it's inefficient when the real simulation is executing. For this reason, ns-3 implements a tracing system based on the concepts of tracing source and tracing sink. A trace source is a piece of code that signals the occurrence of a particular event during the simulation (e.g. a packet is sent). A trace source needs to be linked to a trace sink that consumes the event fired by the trace source. The trace sink makes active use of the callback mechanism to invoke a user-defined function to process the event fired by the trace source.

The topology of the network, the nodes mobility model and the setup of the simulation parameters are done in the main body of the simulation script.

Magrin et al [27] developed a module for the ns-3 network simulator that models a subset of functionalities of the LoRaWAN PHY and MAC layers. Each ED has an associated *EndDeviceLoraPhy* class that models the behaviour of a LoRa chip and tracks its state (standby, sleep, transmission or reception). Some downlink parameters are also set in this class: the receiving SF and frequency and the sensitivity of the chip for different SFs according to the datasheet of SX1272. An analogous class, *GatewayLoraPhy*, is assigned to GWs. The PHY layer of GWs behaves exactly like the PHY layer of EDs. The only difference is in the fact that *GatewayLoraPhy* defines 8 parallel reception paths, each one of them able to lock on an incoming packet on a given frequency. If all paths are occupied the packet is lost. Reception paths are subject to a set of assumptions:

- I. A reception path can be configured to listen to any frequency;
- II. Frequencies can be distributed to paths freely;
- III. It is not necessary to assign a SF to a reception path;
- IV. Only one single reception path listening to a given frequency locks on a packet transmitted at that frequency.

The PHY layer of both a GW and an ED lock on a reception only if the receiver is idle and listening to the right frequency and if the received power is bigger than the sensitivity threshold of the receiver. Whenever a reception starts, the PHY layer notifies a *LoraInterferenceHelper* class that determines if the packet is lost due to interference between concurrent LoRa transmissions. The *LoraInterferenceHelper* class use a Signal-to-Interference Ratio (SIR) matrix, in which the rows represent the SF of the incoming packet and the columns the SF of the interfering transmissions. Each entry in the matrix represents the SIR margin that an incoming packet  $i$  with a given SF needs to have to avoid the interference of a packet  $j$  with the same or a different SF. To this end, the SIR at a given SF is computed as the ratio between the energy of  $i$  and the sum of the energies of the overlapping transmissions that share the same SF. The process is repeated for every possible combination of SFs. The MAC layer is modelled by *EndDeviceLoraMac* and *GatewayLoraMac*. The *EndDeviceLoraMac* class implements the functionalities of a class A ED, sets the address and the SF and opens the first and second receive windows. Duty cycle limitations are also enforced at this layer in the EDs. The *GatewayLoraMac* simply receives uplink packets from the PHY layer and pass downlink packets to the PHY layer. The module provides also a simple NS application that is able to receive MAC commands. Unfortunately, the NS node can only be connected to gateways by means of a point-to-point link, thus limiting more complex configurations adopting other technologies. Other limitations of the module are the lack of support for inter-protocol interference and the absence of ADR and of class B and class C devices.

Van den Abeele et al. [28] propose an ns-3 module similar to the one previously described. The main difference resides in the way in which inter-device interference is computed. In this case, instead of using a SIR matrix, the authors performed a series of complex Matlab baseband simulations to measure the Bit Error Rate (BER) of a set of LoRa PHY configurations over an Additive White Gaussian Noise (AWGN) channel. The BER points are then used to determine the BER curve that is then used to decide if a packet is lost for interference or not.

### 3.2.0.2 SimPy

SimPy [29] is a process-based general purpose discrete-event simulator written for Python. Unlike ns-3, SimPy is not a network simulator, but a framework that only provides the tools that are necessary to schedule events and to execute them sequentially. As a result, SimPy is often used to create custom-built simulators to simulate specific technologies or situations.

LoRaSim <sup>1</sup>, a custom built simulator based on SimPy, is one of the first attempts to build a simulator with the explicit purpose of studying the scalability of LoRaWAN by simulating packet collisions in a dense multi-gateway system. LoRaSim allows to place a set of  $N$  nodes and  $M$  sinks in a 2D space. Each node is characterized by its transmission power  $TP$ , the center frequency  $C_f$ , the SF, the bandwidth  $BW$ , the coding rate  $CR$ , the average transmission rate  $\lambda$  and the packet payload  $B$ . These parameters are used during the simulation to determine if a packet is received by a certain sink (if receive power is above a threshold that depends on SF and  $BW$ ) and, in case of time overlapping transmissions, if a collision occurs at the receiver. This circumstance happens whenever the  $C_f$  of two simultaneous transmissions with the same SF are separated by less than a threshold. The capture effect is also modelled: if a signal is received with a power that is significantly bigger than the other, that signal capture the other and no collision occurs. LoRaSim has a very limited scope and offers few degrees of personalization. In fact only a limited set of node configurations are available out of the box with no possibility of simulating mobile nodes and more complex systems that include other technologies.

### 3.2.0.3 OMNET++

OMNeT++ [30] is an object-oriented modular discrete event network simulation framework that, thanks to its generic architecture, allows to build simulations for a wide variety of applications. The architecture of OMNET++ is based on reusable components called *modules*. Modules can be connected with each others through input and output *gates* and they can exchange any type of data through *messages*. Related modules can be grouped together to form hierarchical compound modules. Modules at the lowest level of the hierarchy, called *simple modules*, implement the specific algorithms of the model and are written in C++, therefore taking advantage of the powerful features of the language. Simple modules are then nested and connected using the Network Description language (NED). Contrary to all the previously described network simulators, OMNET++ is bundled with an IDE, that provides a powerful visualization tool to visualize the topology of the network and to step through the simulation. Even if it is not part of the simulator itself, the INET framework is one of the most popular and used models for OMNET++, since it provides a wide variety of modules that can be used to simulate wired, wireless and mobile IP networks. The INET framework supports many protocols, including the IEEE 802.11 protocol and MANET routing protocols such as AODV, DSDV, DSR, OLSR and GPSR.

FLoRa <sup>2</sup> is the most recent attempt to create a LoRa simulation environment using OMNET++ and parts of the INET framework. Contrary to the modules mentioned in Sections 3.2.0.1 and 3.2.0.2, FLoRa supports ADR both at ED and at the NS and even the computation of energy consumption statistics

<sup>1</sup><http://www.lancaster.ac.uk/scc/sites/lora/lorasim.html>

<sup>2</sup><http://flora.aalto.fi/>

based on the state of the transmitter (idle, receive or transmit). Moreover, the backhaul network can be modelled by using any of the technologies supported by the INET framework. On the other side, FLoRa takes a more simplistic approach on the modelling of the physical layer by resorting to the model used in LoRaSim and already described in Section 3.2.0.2. In this case, a signal captures another overlapping signal only if the stronger signal is received with a power that is at least 6 dbm bigger than the weaker signal and at least five symbols of the PHY preamble are detected by the receiver.

### 3.3 UAVs Path Planning and Optimal Placement for Relaying

Establishing a relay wireless ad hoc network of UAVs comes with a set of non trivial problems: how to optimally place the UAVs in a 3D environment and how to plan the movement of UAVs to avoid collisions and at the same time reach the target objective. A problem that frequently arises is the connectivity versus coverage problem. In an ideal situation, the mesh network should provide the maximum possible coverage, while at the same time it should maintain the connectivity between its members. Unfortunately, due to the limited available resources (i.e. number of UAVs), achieving both requirements is typically not realistic and therefore one must give priority to one or the other. The algorithms that have been developed to tackle the aforementioned problems fall in two main categories: centralized and distributed. Centralized algorithms relies on one single entity having full knowledge and control of the nodes forming the network. This approach typically produces close to optimal results, but suffers from the single point of failure problem and, if the dimension of the problem is big, the amount of computations can generate substantial delays. Mixed integer programming techniques are typically used. On the other hand, distributed algorithms give more approximated solutions, but computations are typically smaller and distributed among the nodes, thus making the network more resilient in case of unexpected situations. Most of the existing distributed algorithms adapts concepts coming from physics or natural animal behaviour. In this section some of the existing approaches tackling the aforementioned problems are presented.

Chandrashekar et al. [31] study the problem of determining the minimum number of UAVs necessary to provide full coverage to clusters of isolated ground nodes and at the same time maintain connectivity between UAVs. All UAVs fly at a fixed altitude  $h$  and have a communication radius with ground nodes of  $MaxRadius$ . The solution is based on a centralized algorithm that runs every  $\Delta T$  s. First, the ground nodes are divided in clusters based on a connectivity matrix. For each cluster, the algorithm determines the set of neighbouring clusters, defined as the clusters in which the distance between the closest points is at most  $2 \cdot MaxRadius$ . Starting from the cluster with fewer neighbours, the subset of neighbours that are also each other's neighbours is selected. The position of the UAV that covers this subset of clusters (super cluster) is determined by means of a convex minimization problem:

$$\min_X R \quad s.t. \quad ||X - p_i|| \leq R \quad i = 1, \dots, M \quad (3.1)$$

where  $X$  is the position of the UAV,  $R$  is the minimum radius of the circle covering the super cluster and  $p_i$  is the position of the ground node of cluster  $i$  which is closer to  $C$ .  $C$  is determined by computing the center of gravity of the center points of each cluster belonging to the considered super cluster. If  $R \geq 2 \cdot MaxRadius$ , the furthest cluster is removed and the computation is performed again until an acceptable solution is found. The aforementioned steps are repeated until all clusters, grouped in super clusters, have an associated UAV. Finally, the UAV to send to the new computed location is chosen by using a metric that minimizes the total travel distance of UAVs.

Caillouet et al. [32] tackle the more challenging problem of optimally positioning the minimum number of UAVs, taken from the set  $\mathcal{U}$  of all available UAVs, in order to cover a set  $\mathcal{N}$  of targets on the ground,

while maintaining the connectivity of the aerial mesh to a ground base station. Targets have a fixed position in the 2D plane  $(x_n, y_n)$ , while UAVs can occupy any position  $p = (x_u, y_u, h_u)$  in the 3D space. The range of Air-to-Air (AtA) communication is a fixed distance  $R_u$ . The AtG range is instead a function of the altitude  $h_u$  and of the half beamwidth  $\theta$  of the directional antenna mounted on each UAV:

$$r_u^h \leq h_u \tan \frac{\theta}{2} \quad (3.2)$$

Each UAV has also an associated cost  $c_u$ , that can be used to model energy-related or cost-related constraints. The solution proposed by the authors is based on a multi-objective linear optimization problem, that minimizes, separately, the functions

$$\min \sum_{p \in P} \sum_{u \in \mathcal{U}} c_u z_p^u \quad (3.3)$$

$$\min \max_{u \in \mathcal{U}, p \in P} h_u(p) z_p^u \quad (3.4)$$

where  $z_p^u$  is a binary variable indicating if UAV  $u$  is deployed at position  $p$ . Intuitively, equation (3.3) tries to minimize the cost of deployment, while equation (3.4) tries to minimize the altitude in order to improve the AtG channel quality. Equations (3.3) and (3.4) are constrained by a set of nine inequalities, omitted for space reasons, modelling the positioning and connectivity constraints defined by the problem. An evaluation of the model performed by the authors proves that the algorithm always find an optimal solution in a reasonable amount of time (1000 s). The authors also state that the impact of the connectivity constraint more than doubles, in some circumstances, the number of deployed UAVs.

Goddemeier et al. [33] consider a distributed decision approach to maintain a coherent mesh network of UAVs with the objective of exploring a 3D area. The swarm has also the additional requirement of keeping the connectivity to a ground base station. Two scenarios are studied by the authors: one in which the connectivity to the base station is permanent (Bounded Relaying) and one in which some disconnections are allowed for the purpose of extending the exploration (Release and Return). Each UAV in the network has the ability of self-selecting, depending on the topology of the network, a different role: Scout Agents (SA) are assigned the task of exploration and sensing, Relay Nodes (RN) keep the communication between one or more UAVs and the base station, Articulation Points (AP) link two clusters that otherwise would be fragmented and Returnees (R) are drones that, in a Release and Return scenario, are coming back to regain connectivity after being detached from the network. In order to maintain mesh connectivity, the authors propose a communication-aware algorithm based on virtual potential fields called Communication Aware Potential Fields (CAPF). According to this algorithm, each UAV is subject to a virtual force

$$\vec{F}_{dir} = \vec{F}_{conn} + \vec{F}_{AP} + \vec{F}_{CA} \quad (3.5)$$

where  $\vec{F}_{conn}$  is the force that keeps a UAV connected to its neighbours,  $\vec{F}_{AP}$  is the force that keeps a SN in the range of an AP or a RN and  $\vec{F}_{CA}$  is the repelling force that keeps the UAV away from obstacles (i.e. other UAVs). The magnitude and direction of each force is determined by the strength of the communication between each pair of UAVs, measured by the RSSI. As an example

$$\vec{F}_{conn} = \sum_{k=1}^d \vec{F}_{conn_k} \quad (3.6)$$

$$\vec{F}_{conn_k} = q |\Delta RSSI| \vec{d}_{0k} \quad (3.7)$$

For each UAV, the  $d$  neighbouring UAVs with the best connections are chosen. For each pair,  $\vec{F}_{conn_k}$  is computed:  $\Delta RSSI$  measures how far is the current RSSI ( $RSSI_{curr}$ ) from the maximum ( $RSSI_{max}$ ) or minimum ( $RSSI_{min}$ ) RSSI threshold, while  $q$  determines if the force is attractive or repulsive. If  $RSSI_{curr}$  is above  $RSSI_{max}$ , the force is repulsive, while if it is below  $RSSI_{min}$ , the force is attractive. The force is zero only when  $RSSI_{curr}$  is between  $RSSI_{min}$  and  $RSSI_{max}$ . The resulting force  $\vec{F}_{dir}$  gives the direction in which the UAV has to move in the current time step.

Di Felice et al. [34] propose a distributed approach similar to [33], but based on VSFs. The UAV mesh network described in the paper is used to give coverage to a set of partitioned ground nodes. The mobility of UAVs is subject to a distributed algorithm that enforces three ground rules:

- I. connectivity of the aerial mesh must be preserved;
- II. QoS of air-to-ground and air-to-air links must be guaranteed;
- III. the number of covered ground nodes should be preserved and maximized.

The aforementioned requirements are obtained by moving each UAV in the mesh according to a virtual force  $\vec{R}$  given by the sum of the individual spring forces associated with each wireless link established by the UAV. In the most basic form, each force is computed as

$$\vec{F} = -k \cdot (\vec{x} - l_0) \quad (3.8)$$

where  $k$  is the stiffness of the virtual spring and  $(\vec{x} - l_0)$  is the displacement of the spring from the neutral position  $l_0$ . Three types of VSFs, with different parameters, are defined: one for air-to-air links, one for air-to-ground links and, if exploration is a requirement, one for attracting scout drones to new areas. The authors propose to compute the displacement  $\delta = (\vec{x} - l_0)$  as a function of the link budget  $LB(i, j)$  measuring the quality of the wireless link between each pair  $(i, j)$  of UAVs.  $LB(i, j)$  and  $\delta$  are defined as

$$LB(i, j) = Pr_j^i - RS_{thr}^i \quad (3.9)$$

$$\delta = \sqrt[\alpha]{\frac{\max(LB(i, j), LB_{req})}{\min(LB(i, j), LB_{req})}} - 1 \quad (3.10)$$

$LB(i, j)$  is simply the difference between the received power  $Pr_j^i$  and the receiver sensitivity  $RS_{thr}^i$ , while  $\delta$  expresses how much the measured link budget deviates from the required link budget  $LB_{req}$ .  $\alpha$  is the decay exponent. The stiffness  $k$  is defined differently depending on the type of link. For AtA links,  $k$  is a fixed parameter that determines how quickly the mesh will react to meet the desired QoS requirements. For AtG links,  $k$  is dynamic and defined as

$$k_{AtG} = \frac{n_i}{\max(n_j) \forall j \in Neigh_i} \quad (3.11)$$

where  $n_i$  is the number of ground nodes covered by the  $i$ -th UAV and  $n_j$  is the number of ground nodes covered by the best connected  $j$ -th UAV belonging to the neighbours of the  $i$ -th UAV. In this way, UAVs connecting more ground nodes will oppose more resistance to movements that might reduce the number of covered nodes. At each time step  $\Delta T$ , the resulting force  $\vec{R}$  is computed for each UAV and, at the end of the interval, the UAV moves with constant speed in the newly computed direction. This happens at the condition that the force magnitude is above a certain threshold. The paper also defines an exploration phase, which can be useful to find additional ground nodes not yet covered by the mesh. Each  $T_{scout}$  seconds, all the UAVs that do not have any other UAV in their visibility zone, become Scout

Nodes (SN) with probability  $p_{SCOUT}$ . The least visited cell in the visibility zone is then selected for exploration and an attractive spring force is computed between the UAV and the center of the cell. In this case the displacement  $\delta$  is fixed to a reference value, while the stiffness is dynamic and given by

$$k_{AtF} = \left(1 - \frac{v_i(j)}{v_i^{Max}}\right)^{v_i^{Min}+1} \quad (3.12)$$

where  $v_i^{Max}$  and  $v_i^{Min}$  are the maximum and minimum number of times the UAV has visited a cell, considering all the cells in the scenario.  $v_i(j)$  represents the number of visits of the  $i$ -th UAV to the  $j$ -th cell. Such definition allows for less visited cells to be explored with a higher priority.

### 3.4 Summary

LoRaWAN is a promising LPWAN protocol that is worth investigating as a core part of LoRaUAV, especially because it promises a good signal range and the recovery of highly degraded signals. Moreover, the possibility of taking advantage of multiple GWs to improve the chances of successful delivery of packets is of great interest in a multi-gateway system like LoRaUAV. The mobility of UAV and their optimal placement for relaying is the most complex problem that must be dealt with in LoRaUAV. Between the two approaches investigated in Section 3.3, the distributed approach seems to be the most simple and flexible, even if optimality can not be ensured. Centralized algorithms are not suitable for the application scenario investigated in this thesis, since they assume, unrealistically, that the value of many parameters is available at all times. Moreover, the presence of mobile nodes on the ground requires a flexible mechanism to rearrange the aerial mesh, even if this means disconnecting one or more drones from time to time. Following the previous considerations, the VSF approach (or in general approaches based on virtual forces) seems promising to solve the problem at hand. In order to simulate such a complex system, ns-3 and OMNET++ are the best available choices, since they are very extensible platforms and they natively provide support for all the network technologies and features needed by LoRaUAV, including LoRaWAN. Both FLoRa for OMNET++ and the LoRaWAN module for ns-3 are actively maintained and they implement the necessary features. The choice of ns-3 over OMNET++ is therefore arbitrary and it comes from the familiarity of the author with the ns-3 simulation environment.

## Chapter 4

# LoRaUAV Design and Simulation

In this chapter, the LoRaUAV system components and the main design choices are presented and described. The main objective of the LoRaUAV system is to provide network coverage to a set of mobile ground nodes using a fleet of UAVs, whose purpose is to promptly relay the collected data to a base station through an ad hoc network established between its members. In order to meet these requirements, each UAV must be able to automatically adjust its position to reflect the continuously changing topology of the ground nodes and at the same time it must be able to maintain a communication path with the base station. Disconnections, although difficult to avoid, must be minimized and a recovery technique must be set up to recover the connectivity whenever it is necessary. LoRaUAV provides mechanisms to deal with all the aforementioned problems. This kind of system might be useful in a wide variety of situations, especially when there is the need to rapidly deploy a network to collect data generated by moving targets in remote or disrupted areas, where other networks might be unavailable, damaged or highly congested. After an introductory overview of the architecture of the system (Section 4.1), the chosen topology algorithm running on UAVs forming the mesh network is described (Section 4.3). The envisioned solution is tested by means of a simulation using ns-3. The developed modules and the main modelling choices and parameters are described in Section 4.4.

### 4.1 Architecture

The architecture of LoRaUAV consists of a two-layer system, in which the first layer is composed of GNs that transmit data using LoRaWAN and the second layer is composed of a swarm of drones communicating over a WiFi ad hoc network and acting as relays between the GNs and a BS. The LoRaUAV main components are listed and described below:

- **GNs:** ground nodes are equipped with devices that are able to aggregate data coming from various sensors and to transmit it using a LoRa module implementing the LoRaWAN protocol. Commercially available LoRa chips, like the Semtech SX1272/73, can be used for this purpose;
- **UAVs:** drones are the most complex elements in the system, given the variety of tasks they need to perform. First of all, each UAV must be equipped with all the necessary sensors, controls and software for navigation and stabilization. Typical sensors include altimeters, speed meters, accelerometers and tilt sensors. The details of the navigation and control system are outside the scope of the thesis and it is simply assumed, realistically, that drones already come with all the necessary components and with a software API to interact with them. The UAVs considered in this work are multi-rotors and thus able to hover over a specific location or target. The type of engine

and fuel depends on the application and on the desired degree of flight autonomy and is outside the scope of this work. From the networking point of view, UAVs carry as a payload a module that integrates a LoRaWAN GW chip, like the Semtech SX1301, and a WiFi chip implementing any of the physical layer standards described in Section 2.3.3 and supporting ad hoc communications. End-to-end packet delivery over WiFi is accomplished by resorting to a traditional TCP/IP stack, with the assistance of a MANET routing protocol. Even though any MANET routing protocol can be used, in the current implementation of the system some mechanisms rely on the knowledge of the whole topology of the network, therefore a table-driven link state protocol like OLSR becomes necessary for the correct operation of LoRaUAV. The module contains also a LoRa-to-WiFi relay application, responsible for adapting uplink LoRa packets received by the GW and send them over the WiFi ad hoc network to the intended destination. The most important and complex part of the whole system is the VSF distributed mobility algorithm. This algorithm is responsible for planning the movements of the drone according to virtual spring forces computed on the basis of parameters collected from positioning sensors, routing tables and messages exchanged with GNs and other UAVs. Given the amount of information needed, the algorithm needs to interface with almost all the previously described components to work properly. A thorough and detailed description of the algorithm is given in Section 4.3.1.

- **BS:** in its most minimal configuration, the BS contains the same WiFi equipment of a UAV, runs the same MANET routing protocol and integrates a LoRaWAN NS that is used to manage the LoRaWAN network as described in more detail in Section 2.2. The ASs and the JS are outside the scope of this thesis and might be placed in any location close or far from the BS provided that another form of network connectivity is available.

A scheme of the aforementioned elements and their components is shown in Figures 4.1, 4.2 and 4.3.

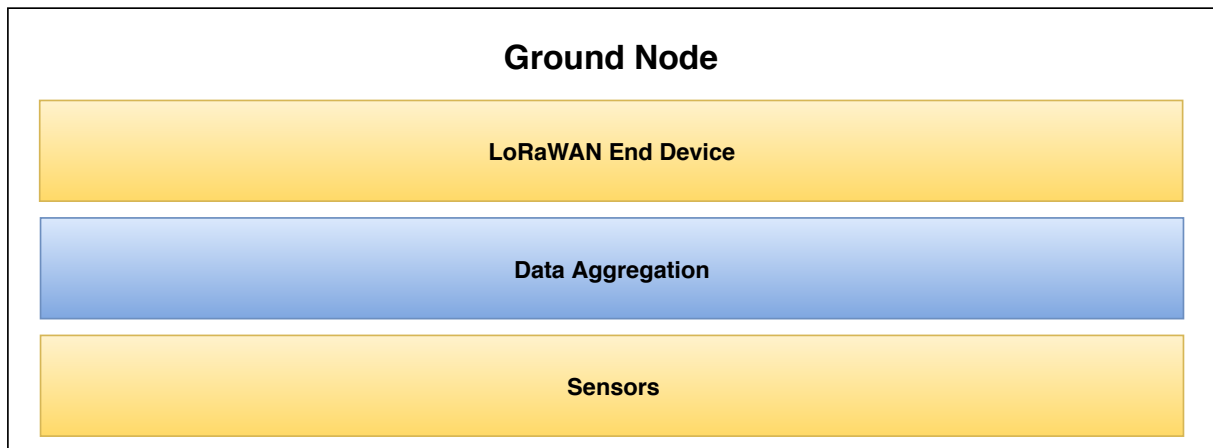


Figure 4.1: System stack of GNs.



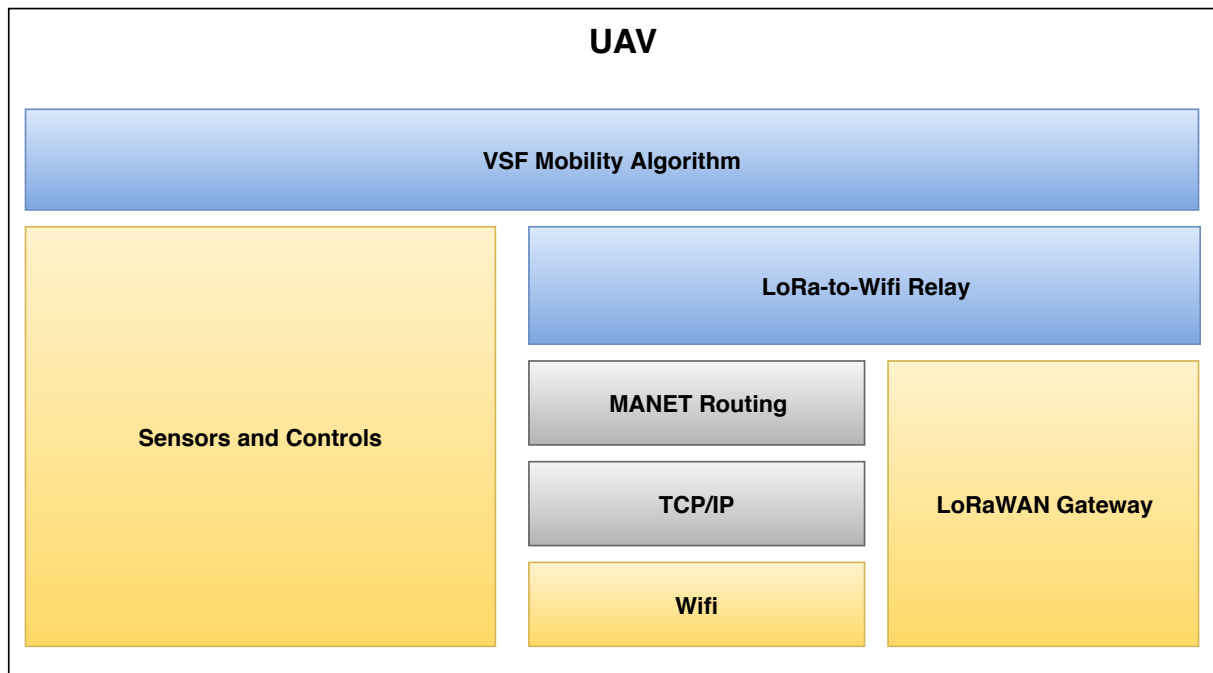


Figure 4.2: System stack of UAVs.

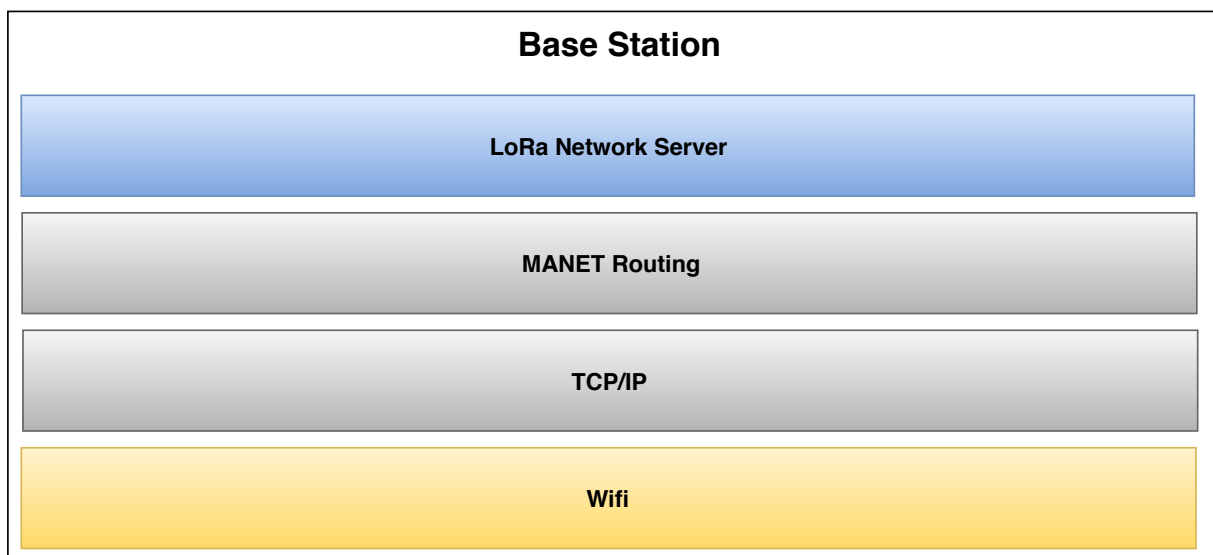


Figure 4.3: System stack of the BS.

## 4.2 Assumptions

Some assumption will hold throughout the thesis:

- I. **All UAVs fly at the same altitude in a 2D plane.** A three dimensional configuration of UAVs is still allowed in the developed system, but the benefits in terms of extended coverage to GNs and collision avoidance are not investigated as part of this work. This means that even if UAVs have different altitudes, the mobility algorithm behaves like if they are on the same plane. An altitude optimization procedure may be introduced seamlessly on top of the system at a later time;
- II. **All UAVs can move with constant speed in any direction.** Drones are complex systems that require complex controllers to manage speed, acceleration, stability and heading along six axis. The present work focuses on the performance of the networks and on the ability of the UAVs mobility algorithm to optimize/recover coverage of GNs, therefore modelling complex drone dynamics is left for future evolutions of the present work;
- III. **UAVs periodically exchange their position and the list of covered GNs with their neighbours.** This can be easily achieved by embedding such information in the periodic updates issued by the routing algorithm or by standalone messages. Spatial coordinates only require latitude and longitude and can be stored and sent as two 4 bytes integers to support a precision comparable to the one provided by typical commercial GPS systems <sup>1</sup>. Assuming that the number of GNs is known beforehand and each one of them has an assigned progressive ID starting from 0, the list of covered GNs can be sent as a list of bits in which the bit in position  $i$  is 0 if the GN  $i$  is not covered and 1 if it is covered. Using this technique, the message size becomes very small even when a large number of GNs is covered;
- IV. **UAVs have access to the received power of neighbouring UAVs and GNs or can estimate it based on their position.** Routing messages can be used to collect the received power, which is necessary to determine the link budget of the links with neighbouring UAVs. The received power of GNs can be collected when an uplink message is received. Eventually, the received power can be estimated by resorting to propagation models and the positions of the nodes;
- V. **The position of the BS is known by all UAVs.** The position of the BS is fixed at all times and therefore can be sent to all drones at the beginning of the operations or set beforehand.

## 4.3 LoRaUAV Mobility

The mobility of the drones is essential to keep the system functional and to adapt the formation to the movements of GNs. LoRaUAV implements three mechanisms that together control the mobility behaviour of the UAVs: the VSF mobility algorithm, the Connection Recovery and Maintenance (CRM) algorithm and the Movement Prediction (MP) algorithm. These mechanisms are described in detail in the following sections.

### 4.3.1 The VSF Mobility Algorithm

LoRaUAV takes advantage of a distributed mobility algorithm through which the direction of movement is determined at each time step by a virtual spring force that can push or pull a UAV closer or further to other UAVs or GNs. In this way, UAVs are kept in the range of each others, possible collisions are

---

<sup>1</sup><https://stackoverflow.com/questions/1220377/latitude-longitude-storage-and-compression-in-c>

avoided and the objective of covering the GNs is pursued. The implemented VSF algorithm is a partially adapted version of the algorithm described by Di Felice et al. in [34] and already presented in Section 3.3. The bulk modifications are related with the criteria for determining the weights of AtA and AtG forces.

In its most simple form, the algorithm runs at fixed regular intervals  $\Delta T$ . At each time step, each UAV generates a set of virtual springs, having one end attached to the UAV that generated them and the other end attached to either a GN (AtG springs) or another UAV (AtA springs). For each spring, the attractive or repulsive force  $\vec{F}_{ij}$  between node  $i$  and node  $j$  is given by

$$\vec{F}_{ij} = K \cdot (LB_{ij} - LB_{req}) \vec{x} \quad (4.1)$$

where  $K$  is a constant that depends on the nature of the spring,  $LB_{ij}$  is the real or estimated link budget between node  $i$  and node  $j$  and  $LB_{req}$  is the required link budget. Choosing link budget measurements instead of physical distances is more meaningful for a communication system. In fact, the link budget gives a more direct indication of the quality of the link and allows for an easier choice of the parameters, e.g  $LB_{req}$ , on the basis of application-specific QoS requirements. The total force  $\vec{F}_{tot}^i$  applied to UAV  $i$  is given by

$$\vec{F}_{tot}^i = \sum_{j=0}^N \vec{F}_{ij} \quad (4.2)$$

where  $N$  is the number of springs originating from  $i$ . AtG springs are established with all the GNs that have sent to the UAV  $i$  at least one message containing their position in the last  $t_{AtG}$  seconds. AtA springs, instead, are established with any one-hop neighbouring UAV. AtA and AtG springs are therefore only established with direct neighbours, thus limiting the complexity of the interactions and the amount of computations.

The  $K$  parameter is fundamental to determine the weight, and therefore the priority, of each spring force  $\vec{F}_{ij}$ . For AtA springs,  $K$  is updated at each time step according to the following expression:

$$K_{AtA} = K_p \left( \frac{N_{neighs}^{max}}{n_{neighs}} \right) \quad (4.3)$$

where  $K_p > 0$  is a scaling factor,  $N_{neighs}^{max}$  is the maximum expected number of neighbouring UAVs and  $n_{neighs}$  is the current number of neighbouring UAVs. Equation 4.3 allows to scale the magnitude of the attractive/repulsive AtA forces on the basis of the relative number of neighbouring UAVs. In this way a UAV with few neighbours will have a higher tendency to stick to the formation, while a UAV with many neighbours will exhibit the opposite behaviour. This gives more importance to AtG springs when a UAV has a high number of neighbouring UAVs. A good value of  $N_{neighs}^{max}$  resides in most of the circumstances in the interval [6-8], since UAVs tend to arrange themselves in a hexagonal grid formation when they are subject only to AtA forces. If the number of drones in the system is small,  $N_{neighs}^{max}$  can be approximated with the total number of UAVs currently operating in the system.

The  $K$  parameter takes a different value at each time step for each AtG spring connected to GN  $j$  according to the expression

$$K_{AtG}^j = \frac{u_{max}}{u_j} \quad (4.4)$$

where  $u_{max}$  is the highest number of neighbouring UAVs covering one or more GNs that are also covered by the current UAV and  $u_j$  is the current number of UAVs covering GN  $j$ . This relation allows to give more priority to the GNs that are covered by less UAVs. Drones are therefore expected to distribute

themselves uniformly between the GNs that are in range. This is especially useful when a group of GNs splits and the covering UAVs must decide which GNs to follow.

### 4.3.2 Connection Recovery and Maintenance (CRM) Algorithm

Even though attractive forces already provide a cohesion mechanism that helps to keep all the members of the swarm in a compact formation, in a system like LoRaUAV clustering is difficult to avoid. Disconnections might be caused by a series of factors, like variable environmental conditions or simply by a UAV or a group of UAVs that goes too far from nearby UAVs due to a particularly strong attractive force. In these circumstances, some mechanisms need to be integrated in the system to recover the connectivity with the UAV formation and therefore with the BS. When the connection to the BS is recovered, a connection maintenance procedure must be triggered in order to avoid further disconnections. The complete CRM algorithm implemented in LoRaUAV is shown in Figure 4.4.

Three mobility behaviors are implemented in the system: the VSF mobility, based on the dynamic computation of forces (see Section 4.3.1), the Network Recovery Mobility (NRM), that consists in moving the drone in the direction of the BS in an attempt to re-establish a connection and the Stationary Mobility (SM) that forces the drone to hover over a specific location. The selection of the right mobility is based on the value of two parameters: *pause* and *persist*. The *pause* parameter is a positive integer number that specifies the number of  $\Delta T$  time intervals during which the UAV must remain stationary. When a connection with the BS is re-established thanks to the NRM, the SM is activated and kept active until, after  $pause \cdot \Delta T$  time intervals, the *pause* parameter reaches the value zero. This mechanism has the purpose of keeping the UAV in a stable position for some time and avoid further disconnections. For the purpose of our work *pause* is set, for each UAV, using the following expression

$$pause = \left\lfloor P_{max} \frac{R_{max}}{D_{BS}} \right\rfloor \quad (4.5)$$

where  $P_{max}$  is the maximum number of pause intervals,  $R_{max}$  is the maximum range of the drone and  $D_{BS}$  is the current distance from the BS.  $R_{max}$  is progressively computed by resorting to the position information periodically exchanged with the neighbouring UAVs. Equation 4.5 assigns a more conservative behaviour to drones that are closer to the BS and therefore the ones that are more likely to cause the disconnection of a big part of the network. On the contrary, peripheral UAVs can move more freely, since it is less likely that their disconnection causes the partition of other parts of the network. The *pause* parameter is ignored only in one circumstance, that is when the *load* parameter is zero. This parameter measures the number of relayed packets during a time window of fixed length  $t_{load}$  and it can therefore be used to select those UAVs that have been inactive for some time. Since those UAVs are not useful for the operation of the system, they can keep moving according to the VSF mobility without the constraints imposed by the CRM algorithm.

The *persist* parameter is an integer number expressing the number of  $\Delta T$  time intervals during which the drone must keep using the VSF mobility even if a connection to the BS is not available. This behaviour is necessary to relax the conditions that trigger the NRM, in the hope that a path to the BS can be re-established thanks to the rearrangement of other UAVs, typically located closer to the BS. In the present work, the *persist* parameter is set according to the following expression

$$persist = \left\lfloor (hops - 1) \frac{D_{BS}}{R_{max}} \right\rfloor \quad (4.6)$$

where *hops* is the number of hops that were separating the UAV from the BS the last time the path was active. Similarly to Equation 4.5, Equation 4.6 allows peripheral UAVs to be dragged by VSF forces for a longer time, even if a path to the BS is not available. On the contrary, UAVs closer to the BS are

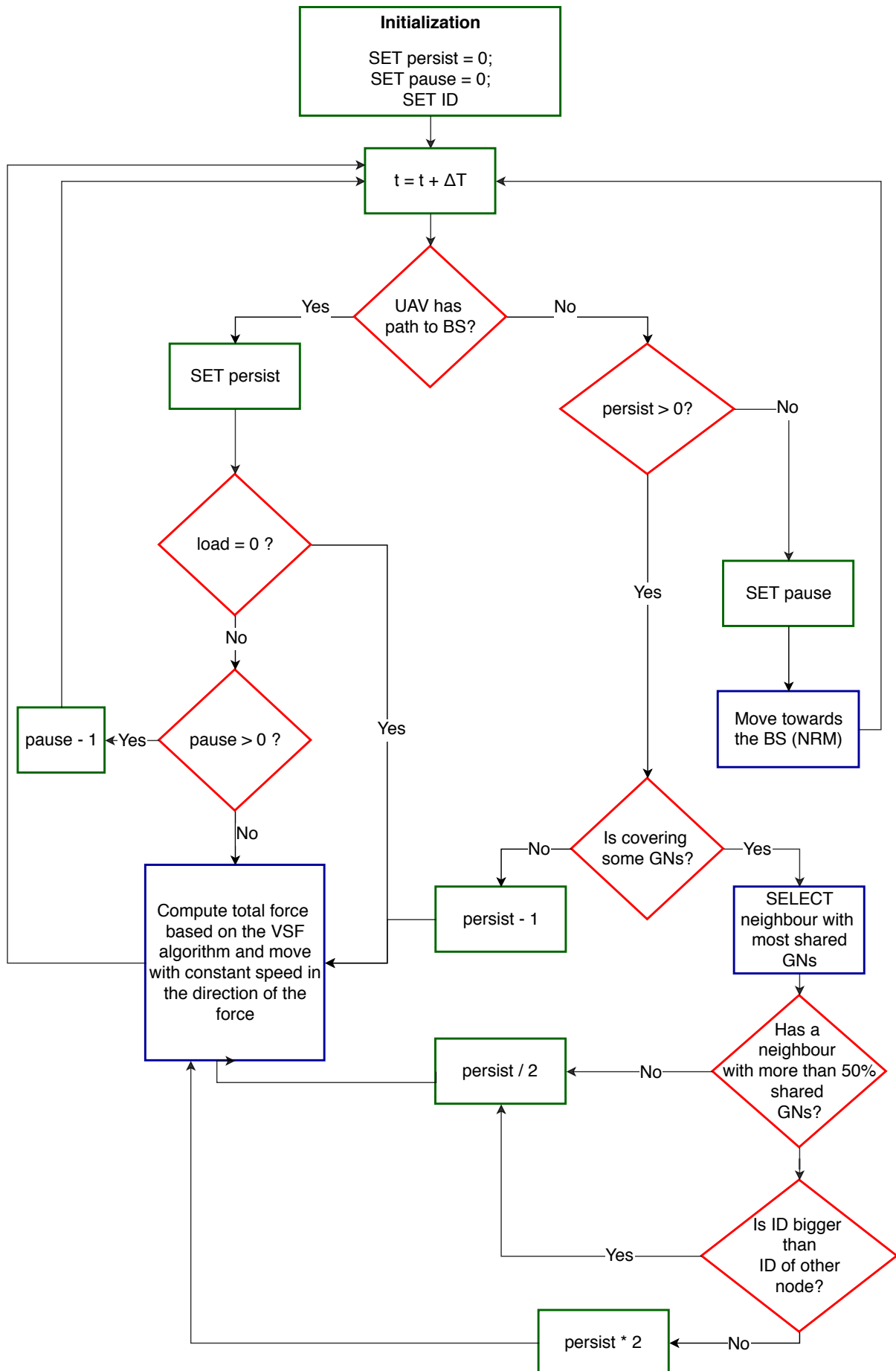


Figure 4.4: CRM algorithm.

less tolerant to disconnections and try to recover a connection sooner than their peripheral counterparts. In normal circumstances, *persist* is decremented by one unit at each time step, until zero is reached. However, there is a situation in which the *persist* counter is updated in a different way. It is possible that a group of UAVs, covering almost the same set of GNs, gets separated from the rest of the swarm. The typical *persist* behaviour would not, in this case, exploit the redundant UAVs to recover the connectivity. Instead, all the UAVs would trigger the NRM almost at the same time and all would loose the connection with the covered GNs. To avoid this kind of situation, UAVs that are covering a set of GNs determine, immediately after loosing the connection to the BS, the neighbouring UAV with which they share the highest percentage of GNs. If this percentage is above a certain threshold  $p_{shared}$ , one of the two UAVs must be redundant. Every UAV has a fixed ID that has been assigned before the operation. The UAV with the lowest ID among the two proceed to cut by half its *persist* intervals, while the other UAV double its *persist* intervals. During the next time steps, one of the redundant UAVs, the one with the lowest value of *persist*, triggers the NRM before the others and reposition itself closer to the BS. This return, in some cases, helps to re-establish a connection between partitioned UAVs and the BS.

### 4.3.3 Movement Prediction (MP) Algorithm

During the operations, some GNs might go out of the range of the UAV swarm and become totally isolated. A recovery mechanism must therefore be implemented. The solution presented in this section is based on a combination of virtual forces and movement prediction. Using the same virtual forces that are used for the normal UAVs mobility, as described in Section 4.3.1, allows to easily integrate the solution in the system and at the same time it allows to preserve the simplicity of the distributed approach that distinguish LoRaUAV.

The recovery mechanism is centred around elements called *holograms*, whose purpose is to signal to the members of the swarm the presumed position of a lost GN or group of GNs. In practice, an hologram is simply a piece of information that in its most simple form contains a pair of spatial coordinates that can be used as a reference to compute a virtual AtG force between the hologram itself and one or more UAVs. For the system to be effective, holograms must be placed in convenient positions. Two alternatives have been considered:

- I. Placing holograms in correspondence of the last known position of a GN or a group of GNs or
- II. Placing holograms in the predicted position of a GN or a group of GNs;

Since GNs are continuously moving, having only the information about their last known position is not enough to generate a useful hologram. In fact, while drones adjust their position, GNs in the meanwhile may have moved somewhere else. If we can assume that GNs keep a regular direction and speed for some time, it is possible to predict the future position of GNs from their previous recorded positions and update the hologram location accordingly at each time step. The prediction implemented in LoRaUAV is based on simple kinematic equations. If we consider a GN  $i$  at time  $t$ , its position  $x_i$  and  $y_i$  at time  $t + 1$  is given by

$$x_{i(t+1)}^p = x_{it} + v_{it}^x + \frac{1}{2} \hat{a}_{it}^x \cdot \Delta t \quad (4.7)$$

$$y_{i(t+1)}^p = y_{it} + v_{it}^y + \frac{1}{2} \hat{a}_{it}^y \cdot \Delta t \quad (4.8)$$

where  $v_{it}$  and  $a_{it}$  are respectively the velocity at time  $t$  and the expected variation of velocity between  $t$  and  $t + 1$  (acceleration). The computation of all the necessary variables only requires the knowledge

of the three most recent positions of a node, so that

$$v_{it}^x = x_{it} - x_{i(t-1)} \quad v_{i(t-1)}^x = x_{i(t-1)} - x_{i(t-2)} \quad v_{i(t-2)}^x = x_{i(t-2)} - x_{i(t-3)} \quad (4.9)$$

$$\begin{aligned} \hat{a}_{it}^x &= a_{i(t-1)}^x + \Delta a_{i(t-1)}^x \cdot \Delta t \quad \Delta a_{i(t-1)}^x = \frac{a_{i(t-1)}^x - a_{i(t-2)}^x}{\Delta t} \quad a_{i(t-2)}^x = \frac{v_{i(t-1)}^x - v_{i(t-2)}^x}{\Delta t} \\ a_{i(t-1)}^x &= \frac{v_{it}^x - v_{i(t-1)}^x}{\Delta t} \end{aligned} \quad (4.10)$$

$$v_{i(t+1)}^x = v_{it}^x + \hat{a}_{it}^x \cdot \Delta t \quad (4.11)$$

and the same applies to  $y$  coordinates. To simplify the problem, in the present work only GNs with constant velocity are considered, so that in the previous equations the acceleration can be set to zero and only the two most recent positions are needed to perform a prediction.

Each UAV is responsible for keeping track of the GNs it has lost communication with and to send to the UAVs in proximity the information needed to predict the positions of the holograms. This information is included in a message called *token* and sent in unicast to all the UAVs that have an entry in the routing table. A *token* includes necessarily the last known position of a GN, its velocity vector and the time at which the last known position was recorded. Upon reception of a *token*, a hologram is created and an expiration time  $t_{exp}$  is attached to it. To avoid the disruption of local AtA and AtG forces, only UAVs that are inactive ( $load = 0$ ) are influenced by forces produced by holograms.

To reduce the amount of exchanged *tokens*, lost GNs can be first clustered in groups based on their position using a clustering algorithm. In the present work, *k-means* is used for clustering and *k-means++* is used to better initialize the centroids and speed up the convergence. The *k-means* algorithm accepts as input the vector data that has to be clustered and the number  $k$  of desired clusters.  $k$  centroids are initialized and all data points are then associated to the closest centroid using a suitable distance measure (the Euclidean distance in our case). The mean of all the points belonging to a centroid is used as a new centroid in the next iteration of the algorithm and the aforementioned process is repeated until convergence. One of the main disadvantages of *k-means* is that it needs to know from the beginning the number  $k$  of clusters. The selection of  $k$  is a non trivial problem that is still at the center of many researches. In LoRaUAV, the *k-means* algorithm is run multiple times with different values of  $k$ , with  $k$  ranging from 2 to  $k_{max}$  and the quality of the clustering is then assessed by using the *average silhouette index*. This index measures the similarity among members of the same cluster and their dissimilarity from the members of other clusters. In formulas, the silhouette  $s(i)$  of the data point  $i$  is

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \quad (4.12)$$

where  $a(i)$  is the average distance of  $i$  from the members of its own cluster and  $b(i)$  is the lowest average distance of  $i$  from the members of neighbouring clusters. The index gets values in the interval  $[-1, 1]$ , with values close to 1 indicating that the data point  $i$  belongs to a good cluster and values close to -1 indicating that the data point  $i$  probably should belong to another cluster. To avoid having too many clusters, data points belonging to a cluster with only one point have a silhouette value of 0. The average  $s_{avg}$  of all  $s(i)$  is used to assess the overall quality of the clustering. In fact a particular clustering is only accepted if  $s_{avg}$  is above a threshold  $s_{thr}$ . If no clustering configuration that satisfies the criteria is found, then all lost GNs are considered as members of a single cluster.

After the appropriate clustering is found, a *token* is created for each cluster and sent to the other UAVs. The token contains the center of mass computed from the last recorded positions of the GNs

belonging to the cluster, the average velocity of the cluster computed as the average of the velocity vectors associated with each GN belonging to the cluster and the time associated with the most recent recorded position among the GNs in the cluster.

## 4.4 Simulation Model

The LoRaUAV system performance is evaluated through a series of simulations performed with the ns-3 network simulator. ns-3 has been chosen over other network simulators for its modular architecture, its speed, its active community and, more importantly, for the support of all the protocols that are necessary for LoRaUAV, including a WiFi module, MANET modules and external modules for simulating LoRaWAN networks. Of the two LoRaWAN modules developed for ns-3 and presented in Section 3.2, the one developed by Magrin et al. has been chosen for its support of the required LoRaWAN features (class A EDs, GWs, inter-device interference, SF configuration) and because it is the most recent and more actively maintained by the authors. Additional modules, utility classes and applications have been developed specifically for implementing the missing features, such as the VSF based mobility algorithm. The complete list of used modules is presented in Table 4.1. In the following sections, the main design choices, configurations, parameters and custom-built classes are justified and described.

Module	Description	Source
Core	Contains core classes for managing times, scheduling events and making callbacks	built-in ns-3.27
Network	Classes to model basic network concepts, such as packets, queues, channels and network devices	built-in ns-3.27
Applications	Contains applications performing a wide variety of tasks, e.g. an UDP client	built-in ns-3.27
Internet	Helpers and classes to set up and configure IPv4/IPv6 networks. Includes also TCP/UDP and routing.	built-in ns-3.27
Mobility	Classes for configuring the mobility of nodes	built-in ns-3.27
OLSR Routing	Models OLSR specific features	built-in ns-3.27
Propagation Models	Offers helpers and classes that model some common propagation models	built-in ns-3.27
WiFi Models	Classes modelling different aspects of a WiFi network PHY and MAC layers	built-in ns-3.27
Lorawan	Classes, helpers and applications to set up a LoRaWAN channel and network	[35]
loravsf	The VSF mobility model and a set of utility classes to manage aspects of the mobility	custom-built

Table 4.1: ns-3 modules used to simulate LoRaUAV.



#### 4.4.1 Simulation Scenario

The performances of the LoRaUAV system are evaluated in a particularly challenging situation: a wildfire disaster scenario in which teams of firefighters intervene to extinguish the fire. A CP, containing a BS, is established in a convenient location and a GCS, responsible for managing the UAVs and process the data, is placed in proximity. A certain amount of UAVs is launched and immediately an ad hoc wireless network is established between them. Fire fighters are divided in teams. Starting from the CP, each team moves to the objective of its assigned mission without any restriction imposed by the aerial mesh network. Each fire fighter is equipped with a Body Area Network (BAN) of sensors, responsible for collecting important measurements from the surrounding environment, like the position, noxious gases levels, heat, heart rate, blood pressure and body temperature. The relevant information is aggregated and sent to the UAVs through a LoRa module and then relayed to the GCS. The scenario is characterized by the presence of a thick vegetation, which complicates the AtG communications due to the significant signal attenuation introduced by trunks and foliage and the absence of LOS between the ground nodes and the aerial mesh.

#### 4.4.2 Channel Propagation Models

The developed simulation consists of two separate channels: an AtA channel for the communications between UAVs over WiFi and an AtG channel for the communications between GNs and UAVs through LoRaWAN. Both channels are configured to use the *ConstantSpeedPropagationDelayModel* class, so that signals propagate with a constant speed equal to the speed of light. This is a reasonable assumption, considering that signals propagate in a low density air medium. A different path loss  $PL$  has instead been chosen for each channel, in order to better characterize two extremely different propagation conditions: AtA links are modelled with a Friis propagation model and AtG links with a Log Distance propagation model. The following sections present the parameters of the models and a detailed description of the motives that led to their choice.

##### 4.4.2.1 Air-to-Ground Model

In the scenario described in Section 4.4.1, GNs move in an environment rich of trees and vegetation. In these conditions, the LOS between GNs and UAVs highly depends on the type and density of the vegetation and is therefore not guaranteed at all times. This produces considerable absorption, scattering and diffraction. Multiple analytical and empirical path loss models have been proposed to properly characterize the effects of trees and foliage. The Modified Exponential Decay model (MED), well known for its simplicity, is described by

$$PL = X f^Y d^Z \quad (4.13)$$

where  $X$ ,  $Y$  and  $Z$  are parameters dependent on the type and density of the vegetation,  $f$  is the frequency in hertz and  $d$  the depth of the foliage obstructing the LOS in meters. Other similar models have been proposed (see [36]). In the context of the system proposed in this work, the results of Iova et al. [20], already described in Section 3.1, show that it is reasonable to expect a significant range drop, from few kilometers to few hundred meters, when LoRaWAN is used in environments with thick vegetation, even if high SFs are used. Keeping such considerations as a reference to test the goodness of a model, several of the aforementioned empirical path loss models have been implemented and tested in ns-3. However, they all failed to give good approximations of the losses incurred by a LoRa signal, thus producing too optimistic and therefore unrealistic ranges. For this reason, the RSSI metrics collected

in [20] in a forest environment have been used to estimate the parameters of a Log Distance path loss model described by the equation

$$PL = L_0 + 10n \log_{10} \left( \frac{d}{d_0} \right) \quad (4.14)$$

where  $n$  is the path loss exponent,  $d$  the distance in meters from the receiver and  $d_0$  and  $L_0$  are respectively the reference distance and the reference loss for  $d_0$ . The estimated parameters are reported in Table 4.2 and the trend of the path loss is shown in Figure 4.5. This model does not claim to be extremely realistic, but it gives a worst-case estimation of the LoRaWAN expected range that agrees with real experimental measures. The built-in ns-3 *LogDistancePropagationLossModel* class is used to implement the model.

Parameter	Value
$d_0$	1 m
$L_0$	31.22 dBm
$n$	5.2

Table 4.2: Values of the parameters of the Log Distance model.

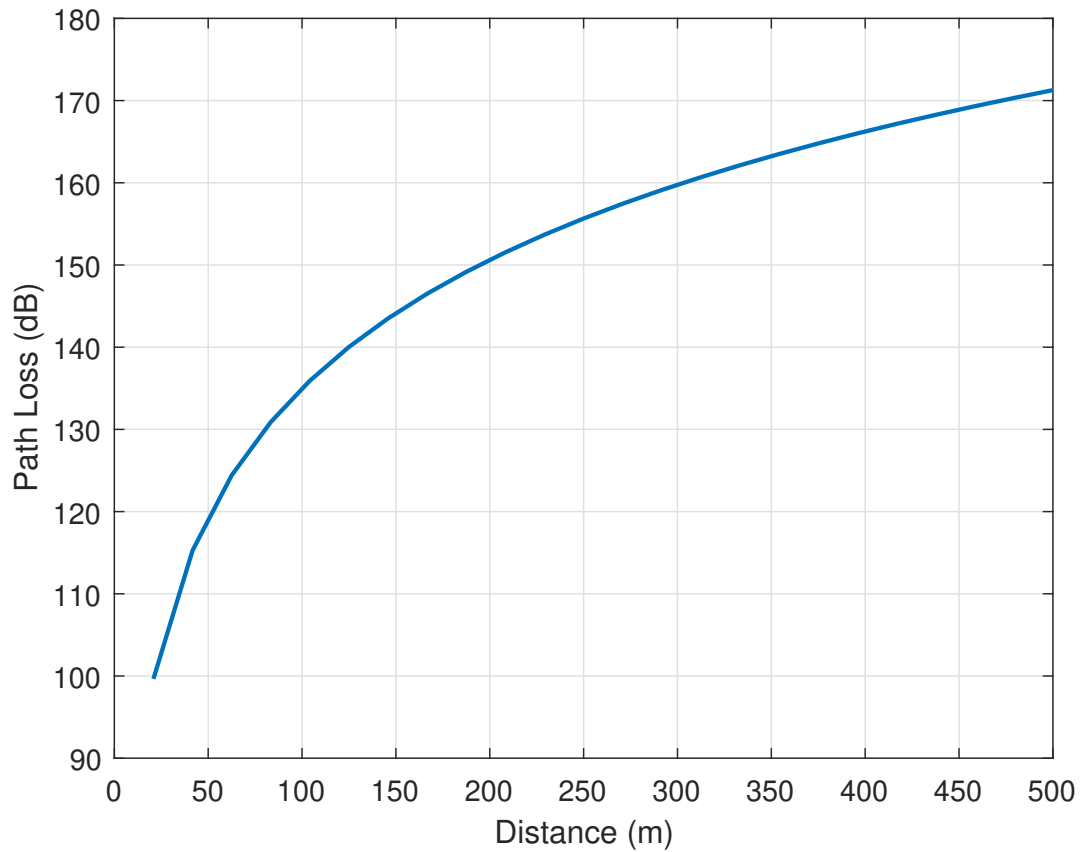


Figure 4.5: Log Distance path loss trend for AtG links.

#### 4.4.2.2 Air-to-Air Model

In our simulations it is assumed that the area of the operations is flat and no obstacles are present in the LOS of the UAVs at any moment. Since UAVs operates several meters above the ground, the LOS assumption is acceptable. Under these conditions, the Friis propagation model represents a reasonable choice, as it is also stated in [37]. The path loss  $PL$  in dB is therefore computed as

$$PL = G_t + G_r - 10 \log \left( \frac{(4\pi d)^2 L}{\lambda^2} \right) \quad (4.15)$$

where  $G_t$  and  $G_r$  are the gains of the antennas (considered isotropic) in dBi,  $\lambda$  is the wavelength in meters,  $d$  is the distance between the transmitter and the receiver in meters and  $L$  is the system loss. In the used model,  $G_t$  and  $G_r$  are set to 0 and  $L$  is set to 1.  $\lambda$  is set according to the frequency used by the WiFi physical layer, which corresponds to 2.4 GHz for the IEEE 802.11g protocol used in our simulations. The built-in ns-3 *FriisPropagationLossModel* class is used to implement the model.

#### 4.4.3 Mobility Model of UAVs

The mobility model of UAVs, as described in Sections 4.3.1, 4.3.2 and 4.3.3 is implemented in a new ns-3 module called *loravsf*. The *VirtualSprings2dMobilityModel* class, a subclass of the *MobilityModel* class, is the entry point for all the mobility decisions and is responsible for:

- Setting the initial attributes, for example the duration of the time step  $\Delta T$ , the speed of the drones and the required link budgets for AtA and AtG links;
- Initializing and updating the *pause* and *persist* parameters at each time step and deciding when CRM behaviours must be triggered;
- Computing the attractive/repulsive force as the sum of AtA and AtG forces and move the UAV in the force direction;
- Generate *tokens* when GNs are lost and send them to the other UAVs.

Additionally, some specialized classes have been developed to manage some aspects of the mobility algorithm. An instance of each one of them is used inside the *VirtualSpring2dMobilityModel* class. These classes are:

- *LinkBudgetEstimator*: utility class used to estimate the link budget based on AtA and AtG propagation models, the sensitivity of the receivers and the position of the nodes at the ends of the link. Any instance of the class *PropagationLossModel* can be plugged in, so that specialized propagation models can be used for different types of environments;
- *LoRaEdsMonitor*: class responsible for monitoring the incoming LoRaWAN packets and compile a list containing their time of arrival and the position of the GNs that sent them. A tolerance parameter  $t_{AtG}$  is used to filter the list, so that it contains only the GNs that sent a packet in the last  $t_{AtG}$  seconds. The list is then used by the mobility algorithm to determine the covered GNs and to compute the AtG forces. The class is also responsible for compiling the list of lost GNs, saving their most recent positions, performing the clustering using *k-means* and finally creating *tokens* as described in Section 4.3.3. A GN is considered to be lost if no message has been received in the last  $t_{lost}$  seconds. To avoid using stale information, the list of lost GNs is always filtered based on the time that separates the two last recorded positions, which can not be larger than  $t_r$  seconds.

For clustering, the class uses a partially modified version of a *k-means++* source file found on the internet <sup>2</sup>;

- *LoadMonitor*: utility class that takes care of monitoring the IP packets relayed by the UAV in a time window of length  $t_{load}$ , with the purpose of estimating the experienced load. This information is then used by the mobility algorithm to select the most convenient mobility behaviour;
- *HologramManager*: class responsible for collecting and updating the received tokens, to delete them when they expire and to compute, upon request, the current position of the *holograms* based on the information contained in the tokens.

#### 4.4.4 Mobility Model of Firefighters

The VSF algorithm adapts the topology of the UAV swarm based on the movements of the GNs. This means that to validate the performance of the developed algorithm, it is necessary to assign a suitable mobility model to the GNs. Given the scenario that is being investigated, GNs must act similarly to firefighters during a wildfire operation. Unfortunately, to the best of our knowledge, no mobility model has ever been investigated or developed for this kind of situation and therefore a new one has been developed based on information collected from [38] and [39]. When a wildfire is detected, a CP is placed in a safe area and teams of fire fighters are then sent to one or multiple intervention areas, that are typically close to natural fire barriers like roads, rivers or already burnt areas. Once arrived to the target location, fire fighters can adopt several techniques to stop or slow down the advancement of the fire, such as building artificial fire lines or performing a direct attack if the fire is small. An approximation of this behaviour has been implemented in the *FiremenMobilityModel* ns-3 class, which is a specialization of the *MobilityModel* class. According to this model, each GN  $i$  is assigned to a team  $j$ , with  $j = 0, 1, 2, \dots, N$ . Teams can initially be placed anywhere in the simulation area, which is assumed to be an  $S \times S$  square. The whole area is then divided vertically in  $N$  equal width columns, one for each team, and each column is in turn split horizontally in  $R_j$  equal width rows, where

$$R_j = \lfloor S/d_{rj} \rfloor \quad (4.16)$$

where  $d_{rj}^j$  is the retreat distance of team  $j$ .  $R_j$  can be set differently for each team. After this process, every team has an assigned vertical stripe with  $R_j$  cells. At  $t = 0$ , each team elects a random point in the furthest cell from the origin of the scenario in its assigned stripe and move towards that direction with constant speed until the destination is reached. At this point the team members start moving independently and randomly for a predefined amount of time  $t_{rw}$  (deterministic or random) in a rectangular operation area with dimensions  $s_j^x \times s_j^y$ . When time is over, teams elect a new point in the second furthest cell and repeat the previously described process. Teams stop moving when the simulation is over or when all the cells of their assigned column have been visited. The idea behind this model is to mimic the advancement of a wildfire and the consequent retreat of firefighters to fallback areas where new attacks to the fire may be attempted. A simplified representation of the model is shown in Figure 4.6.

<sup>2</sup>[https://rosettacode.org/wiki/K-means%2B%2B\\_clustering#C](https://rosettacode.org/wiki/K-means%2B%2B_clustering#C)

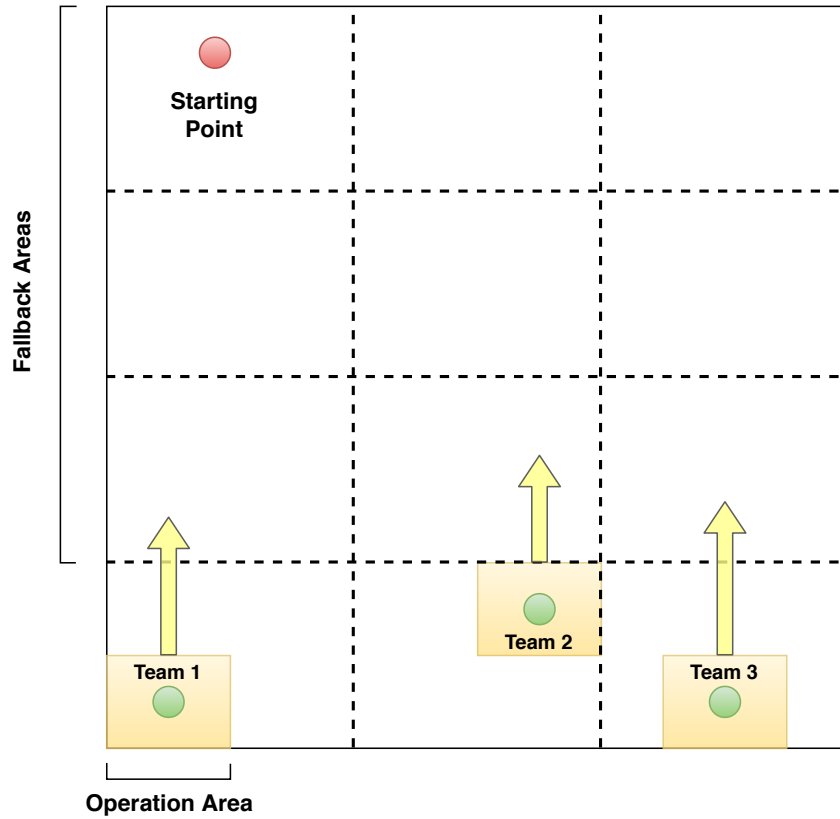


Figure 4.6: Firemen mobility model. All teams start from a single starting point and begin their operation in the furthest cell from the origin in their assigned column.

#### 4.4.5 Applications

The behaviour of the nodes taking part in the simulation is implemented in ns-3 with applications, which are instances of the general *Application* class. The following applications are used in the simulation:

- *PacketSink*: this application is installed in the BS and is responsible for consuming and reporting to the main script the packets sent by the aerial mesh. Since we are not interested in the actual processing of packets, there is no need of a more complex logic. *PacketSink* is a built-in ns-3 application;
- *PeriodicSender*: this application is part of the *lorawan* module. It allows EDs to send LoRaWAN packets at fixed periodic intervals. To avoid the superposition of transmissions, each application is started with a random delay of a few milliseconds;
- *UdpRelay*: custom built application installed in every UAV. It is responsible for collecting the incoming packets arriving at the LoRaWAN gateway interface and adapt them into UDP packets ready to be sent over WiFi to the BS. The *UdpRelay* application takes also care of buffering packets when no route to the BS is available and to empty the buffer whenever there is the opportunity.

#### 4.4.6 Simulation Script

The simulation scenario is instantiated and configured in the simulation script. The script accepts the following input parameters from the command line:

- I. *NumUAVs*: the number of UAVs in the simulation;

- II. *NumTeams*: the number of teams in the scenario. The maximum allowed number is 5;
- III. *NumMembers*: the number of nodes per team;
- IV. *Size*: the size of the simulation area;
- V. *Time*: the duration of the simulation in seconds;
- VI. *Run*: integer number that is used to seed the Random Number Generator (RNG) of the simulator;
- VII. *Trace*: if true an XML trace of the movements of the nodes is produced. The file is saved with the name *trace-r<Run>-t<Time>-s<Size>-u<NumUAVs>-tm<NumTeams>-mb<NumMembers>.xml*. The trace can be imported and visualized in *NetAnim*.

The *main* function then instantiates and configures channel objects, helpers and nodes and installs on the latter ones the required network devices, routing protocols, internet stacks, mobility models and applications. Some callback functions are used to collect useful data from the simulation. This data is later used to compute some statistics of interest through Python scripts. In particular, at the end of the simulation, the following files are produced:

- *nodes-stats.out.txt*: each line reports, in order, the id of the GN, the number of sent packets, the number of unique packets received by the GW, the number of unique packets received by the BS;
- *nodes-avg-stats.out.txt*: contains the average PRR at the GW, the average packet delay to the GW, the average delay to the BS;
- *mesh-stats.out.txt*: each line reports, in order, the number of unique packets that have been relayed by UAVs and the number of unique packets relayed by UAVs and successfully received by the BS;
- *coverage-stats.out.txt*: reports for each time step, the percentage of GNs covered by at least one UAV;
- *coverage-avg-stats.out.txt*: contains the average percentage of covered GNs.

Occasionally the simulation script has been adapted to collect other data relevant to the evaluation of the proposed solutions.

## 4.5 Summary

In this chapter, the architecture of LoRaUAV, its core algorithms and its simulation model have been presented in detail. LoRaUAV is a network system based on UAV LoRaWAN GWs forming a WiFi ad hoc network and acting as relays between mobile LoRaWAN GNs and a fixed remote BS. The mobility algorithm, the core of LoRaUAV, is based on the computation of virtual spring forces that can attract or repel a UAV from its neighbouring UAVs or GNs depending on the required link budget. The basic VSF algorithm is extended with the CRM algorithm and the MP algorithm. The first uses pause and persist behaviours to better distribute the available UAVs in the monitored area, while the latter uses k-means clustering and movement prediction as a last attempt to restore a connection with out-of-range GNs. LoRaUAV and its mobility algorithms have been modelled in the ns-3 simulation environment through already existing modules and a new *loravsf* module. A particular attention has been given to the reproduction of the conditions of a wildfire situation, including a suitable AtG path loss model and a realistic mobility model for firefighters. Results are collected through callback functions and manipulated through appropriate scripts to extract relevant QoS metrics.

## Chapter 5

# Results and Analysis

In this chapter, the performance of the LoRaUAV mobility algorithm, as described in Section 4.3, is assessed in a wildfire disaster scenario through a series of simulations in ns-3. The basic VSF algorithm, described in Section 4.3.1, is tested first and an optimal range of values for  $K_p$  is determined. A comparison between the LoRaUAV VSF algorithm and the VSF algorithm proposed by Di Felice et al. in [34] is then performed. It is proven that the performance gain in terms of PRR is small and limited to just some values of  $K_p$ , but inexpensive. A significant improvement of the performance is then shown after activating the CRM algorithm. Next, the impact and significance of the MP algorithm on the LoRaUAV system is measured and analysed. Finally, the frequency and duration of disconnection periods of GNs is investigated and commented.

### 5.1 Simulation Parameters

The main simulation parameters, fixed in all the simulations performed in this chapter, are listed in Table 5.1. LoRaWAN PHY layer parameters correspond to the default values set in the *lorawan* module. The SF is set to 7 to minimize the delay and the collisions. Higher SFs have been considered, but some simulation tests showed negligible increase in terms of range due to the harsh AtG path loss parameters that have been chosen (see Section 4.4.2.1). In the context of this work, GNs are only required to send their position, therefore the packet size is set to 10 bytes, which is enough to hold a pair of GPS coordinates. Packets are sent every 30 seconds.

The WiFi ad hoc network operates using IEEE 802.11g, a protocol that is broadly supported and implemented in cheap hardware. The modulation is ERP-OFDM with a maximum throughput of 12 Mbps, which is high enough to support the traffic generated by LoRaWAN GNs. Moreover, the selected modulation is a good compromise in terms of signal range and throughput. The transmission power and receiver sensitivity are set to the default ns-3 values, which correspond to common WiFi settings.

The MANET routing protocol is OLSR. Proactively building the routing tables is in fact essential for the correct functioning of the LoRaUAV mobility algorithm, since the information they contain is used to compute various important parameters. The impact of the routing protocol on the performance of the system has not been investigated as part of this work, but it is reasonable to expect that, at least in highly mobile networks, any protocol that provides fast convergence of routes and frequent periodic table updates would benefit the LoRaUAV system.

In the LoRaUAV mobility algorithm, the numerous timeout timers and time thresholds have been set to reasonable values according to the packet transmission period of GNs. To perform movement prediction, *k-means* is run with  $k$  ranging from 2 to  $k_{max} = 3$  in order to reduce the complexity of the

LoRaWAN	
EDs Tx Power	14 dBm
EDs Class	Class A
Bandwidth	125 kHz
Frequency	868.1 MHz
SF	7
GW Sensitivity (dBm)	-124 (SF 7), -127 (SF 8), -130 (SF 9), -133 (SF 10), -135 (SF 11), -137 (SF 12)
Packet size	10 bytes
Packet period	30 s
WiFi	
PHY layer protocol	IEEE 802.11g
Modulation	ERP-OFDM 12 Mbps
Frequency	2.4 GHz
Tx Power	16.02 dBm
Receiver Sensitivity	-99 dBm
Routing	
Protocol	OLSR
VSF algorithm	
$\Delta T$	10 s
$t_{AtG}$	40 s
$LB_{req}$	20 dbm
CRM algorithm	
$P_{max}$	30
$p_{shared}$	0.5
$t_{load}$	3 min
MP algorithm	
$t_{lost}$	3 min
$t_r$	40 s
$t_{exp}$	10 min
$k_{max}$	3
$s_{thr}$	0.6
Others	
UAV altitude	60 m
GN speed	1.39 m/s

Table 5.1: List of fixed simulation parameters.



computations. Higher values can be used if enough computational power is available. Only cluster configurations with an average silhouette bigger than  $s_{thr} = 0.6$  are accepted.

The altitude of UAVs is set to 60 m above the ground, which corresponds to the maximum height of trees in temperate climates. The terrain is considered flat. GNs moves at a normal walking speed of 1.39 m/s ( $\simeq 5$  km/h), a value that has been taken from the research of Bohannon [40].

## 5.2 QoS Metrics

The performance of the algorithms is determined by computing and comparing some QoS metrics averaged over a certain number of simulation runs. In particular, the main considered metrics are:

- **Average End-to-End Packet Reception Rate (AE-PRR):** the average percentage of unique packets sent by all GN and correctly received by the BS through the mesh network;
- **Average Total Delay (ATD):** average delay of packets sent by GNs to the BS through UAV gateways. It is the sum of the delay experienced in the LoRaWAN network segment and in the WiFi network segment. It includes the delay caused by buffered packets;

Other QoS metrics have been collected, but they play a secondary role in the analysis:

- **Average LoRaWAN Packet Reception Rate (AL-PRR):** the average percentage of packets sent by all GNs and received by at least one UAV gateway during the simulation time;
- **Average Mesh Packet Reception Rate (AM-PRR):** the average percentage of unique packets successfully relayed by UAVs through the mesh network and correctly received by the BS during the simulation time;
- **Average Covered GNs (ACGN):** average percentage of GNs covered by at least one UAV during the simulation time;

All the averages are reported with their 95 % confidence interval, computed using the standard deviation estimated from the samples. It is assumed that the population has a Student's t-distribution.

## 5.3 Generation of Scenarios

The LoRaUAV system is evaluated in wildfire scenarios generated according to the model presented in Section 4.4.4. In all the simulations performed in this chapter, the parameters of the mobility model are set according to table 5.2. The BS is always positioned at coordinates  $(S/2 \text{ m}, 300 \text{ m})$ , where  $S$  is the dimension of one side of the squared simulation area. The initial position of teams of GNs is chosen uniformly at random in a circle of radius 100 m around the BS. UAVs are also positioned uniformly in a circle around the BS, but with a higher radius of 300 m. A schematic visualization is given in Figure 5.1. For reasons related to the available computational time, the number of firefighter teams is limited to the range  $[1, 5]$  with each team having 20 members. This last number is not casual: to conform as much as possible with the application at hand, the 20-units wildland firefighter teams operating in the United States, called hotshots crews [41], are taken as a realistic model of GN deployment in a wildfire scenario. In order to obtain statistically significant results, each simulation test is run 100 times with progressive seeds in the range  $[0, 99]$ .

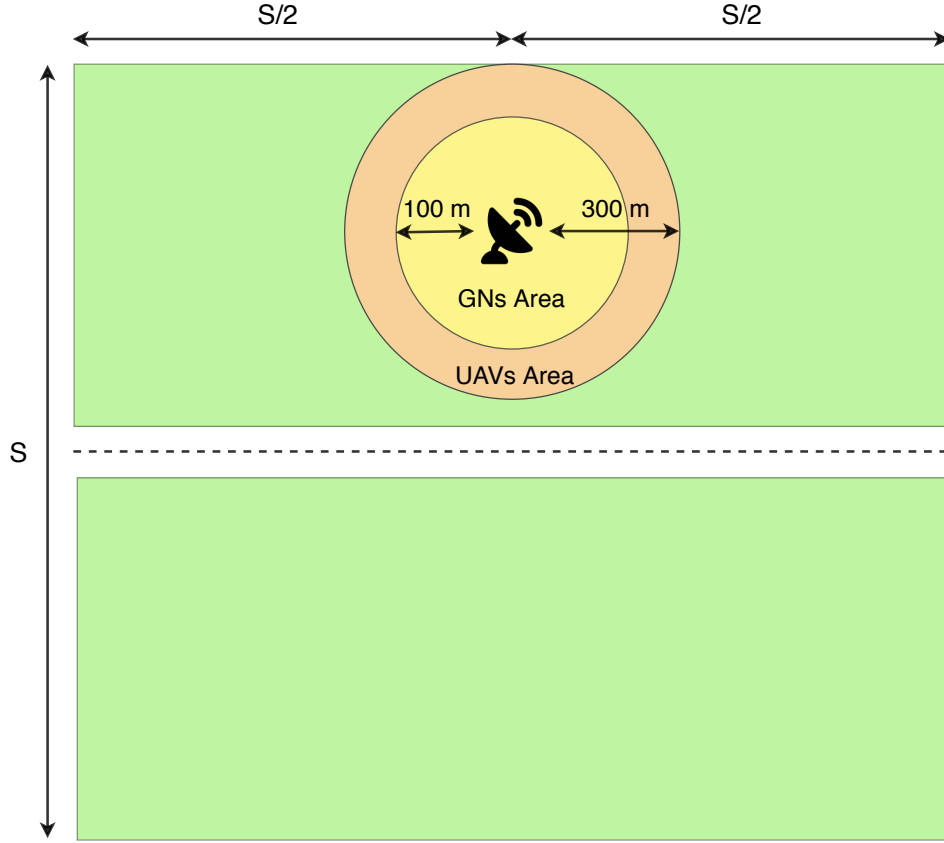


Figure 5.1: Initial configuration of simulation scenarios.

Parameter	Value
$s_j^x \forall j$	300 m
$s_j^y \forall j$	100 m
$d_r^j$	$\mathcal{U}(50, 250)$ m
$t_{rw}$	5 min

Table 5.2: Firefighters mobility model parameters.

## 5.4 Behaviour and Performance of the VSF Algorithm

In this section, the performance of the basic VSF algorithm, as described in Section 4.3.1, is evaluated without the presence of the CRM algorithm and the MP algorithm. Disconnections from the BS are simply solved by using the NRM. As discussed in Section 4.3.1,  $K_{AtA}$  is computed as follows:

$$K_{AtA} = K_p \left( \frac{N_{neighs}^{max}}{n_{neighs}} \right) \quad (5.1)$$

As a preliminary step, the trend of some relevant QoS metrics as a function of  $K_p$ , the number of GNs and the number of UAVs is determined. This step is necessary to determine how to select the values of  $K_p$  in subsequent simulations. The AE-PRR has been selected as the main metric because it better represents the performance of the overall system, since it takes into account both the coverage

provided to GNs and the cohesion of the UAV mesh network. A comparison is then performed with the algorithm proposed by Di Felice et al. (from now on called DF algorithm) already described in Section 3.3.

#### 5.4.1 Study of the Impact of $K_p$ on the Coverage of GNs

In this section, the impact of  $K_p$  on the coverage of GNs is assessed. The objective is to determine if and how much different values of the parameter affect QoS metrics in scenarios containing a different number of GNs and UAVs. The parameters used in the simulations are listed in Table 5.3.

Parameter	Values
Number of UAVs	(4, 6, 8, 10, 12)
Number of teams	(1, 2, 3, 4, 5)
GNs per team	20
Total simulation time	2000 s
Simulation area	$2000 \times 2000$ m

Table 5.3: Parameters used for simulations described in Section 5.4.1 and Section 5.4.2.

The total simulation time is big enough to allow the GNs to reach the opposite side of the scenario and sufficiently small to perform the simulations in a reasonable amount of time. Considering the dimension of the simulation area, the maximum number of UAVs has been set to 12 to avoid a saturation which would have produced non-indicative results. The values of  $K_p$  to be tested have been chosen in a relatively wide range, since there was no indication of any good range of values before the tests were performed, apart from sparse and not statistically significant simulations performed during the development of the system model. Finally, tests have been run with an increasing number of teams, from one to five, to understand if a relation exists between the number of GNs and  $K_p$ . The trend of the AE-PRR with three teams on the ground is shown in Figure 5.2. Some observations can be made:

- I. As expected, the AE-PRR increases with the number of UAVs. In fact, the addition of 2 UAVs to the system produces an  $\approx 10\%$  increase of the AE-PRR;
- II. The impact of different values of  $K_p$  on the AE-PRR is strong, especially when a larger number of UAVs is deployed. In fact, when 12 UAVs are used,  $K_p = 25$  produces a result which is  $\approx 30\%$  higher than the one produced by  $K_p = 1$ ;
- III. The AE-PRR increases until a certain value of  $K_p$  and then it stays almost constant before starting decreasing again. This behaviour is absolutely normal. In fact, after  $K_p$  reaches the optimal value, which represents the optimal equilibrium of AtA and AtG forces, further increases become useless or even counterproductive. If the value of  $K_p$  is too big, cohesive forces generated by the UAVs that do not cover any GN become too strong and therefore the cohesion of the mesh prevails over the coverage. This also explains why such behaviour is more noticeable when 12 UAVs are used. In fact, in this situation, most of the UAVs behave as relays and are not actually covering any GN. The flat part of the plot defines the range of optimal  $K_p$  values;
- IV. The AE-PRR remains almost constant for all values of  $K_p$  when 4 or 6 UAVs are used. This comes from the fact that the number of UAVs is too small for offering good coverage to the scenario and therefore, apart from a small period of time at the beginning of the simulation, all GNs remain

isolated from the UAV mesh for most of the simulation time. When this happens,  $K_p$  does not really make a difference anymore;

Changing the number of teams on the ground does not change the aforementioned observations, as is possible to see in Figure 5.3, Figure 5.4, Figure 5.5 and Figure 5.6. However it is possible to notice that the number of teams influences the range of  $K_p$  values that gives the best results in terms of AE-PRR. As an example, with only one team and 12 UAVs, the best results are obtained with  $K_p$  in the range  $[10, 15]$ , while with two teams and 12 UAVs the best range becomes  $[15, 20]$ . In general we can observe that a bigger number of GNs requires bigger  $K_p$  values. Another immediate and obvious consequence of adding more GNs is the decrease of the maximum achievable AE-PRR, which passes from  $\approx 0.9$  with one team to  $\approx 0.65$  with five teams. This is one of the most critical aspects of the basic LoRaUAV VSF algorithm, since the high packet loss might be too penalizing for a critical application like the one that is being investigated. In fact, even when just 60 GNs divided in three teams are deployed and 12 UAVs are used, the packet loss is above 20 %. This is the drive that brought to the development of the CRM and MP algorithms.  $K_p$  is not only influenced by the number of teams, but also by the number of GNs per team, as it is possible to observe in Figure 5.7. The figure shows the AE-PRR of two simulation scenarios: one having 3 teams of 10 GNs and the other having 3 teams of 40 GNs. Compared with the results reported in Figure 5.2, in the scenario with only 10 GNs per team, the system behaves better with a lower  $K_p$  (range  $[5, 10]$ ). On the contrary, in the scenario with 40 GNs per team, the system performs better with a higher  $K_p$  (range  $[45, 50]$ ). This shows that a higher density of GNs moving in the same area requires bigger values of  $K_p$  to obtain the best AE-PRR results. Other interesting observations can be made by studying the trend of the ATD. A plot is shown in Figure 5.8 for three teams of GNs. The ATD decreases with the number of UAVs and with bigger values of  $K_p$ . In order to really understand this behaviour, it is necessary to remember that the ATD is computed only on packets that are successfully received by the BS and it takes into account also the time that a packet eventually spends in the buffer. Therefore, in scenarios with few UAVs, the mesh is more subject to disconnections and the bigger number of buffered packets contributes in large part to the big delay. Disconnections and buffering time also explains why lower delays are associated with bigger values of  $K_p$ : a big  $K_p$  favours the compactness of the aerial mesh to the detriment of the coverage. This is confirmed by the decreasing trend of the average percentage of buffered packets shown in Figure 5.9, which is as an indication of the probability of disconnections in the mesh.  $K_p$  must therefore be chosen as a compromise between the desired coverage and delay. The analysis of the other QoS metrics produces some minor observations. The AM-PRR is almost constant in all the situations, ranging from 96 % to 98 %. In fact, only a small percentage of packets is lost in intermediate relay nodes. AL-PRR and ACGN show similar trends. This can be explained by the fact that only few LoRaWAN packets are lost for collisions or GW saturation, instead most packets are lost because the node is not in the range of any UAV. AM-PRR, AL-PRR and ACGN plots are reported in Section A.1. In conclusion,  $K_p$  proved to be an important parameter that greatly influences the performance of the system. It has therefore to be chosen properly depending on the number of UAVs, the number of teams and the number of GNs per team. Configuring  $K_p$  might be complex in a real system, due to the numerous variables that must be taken into account. A future study of a  $K_p$  optimization routine is therefore necessary to simplify the configuration process.

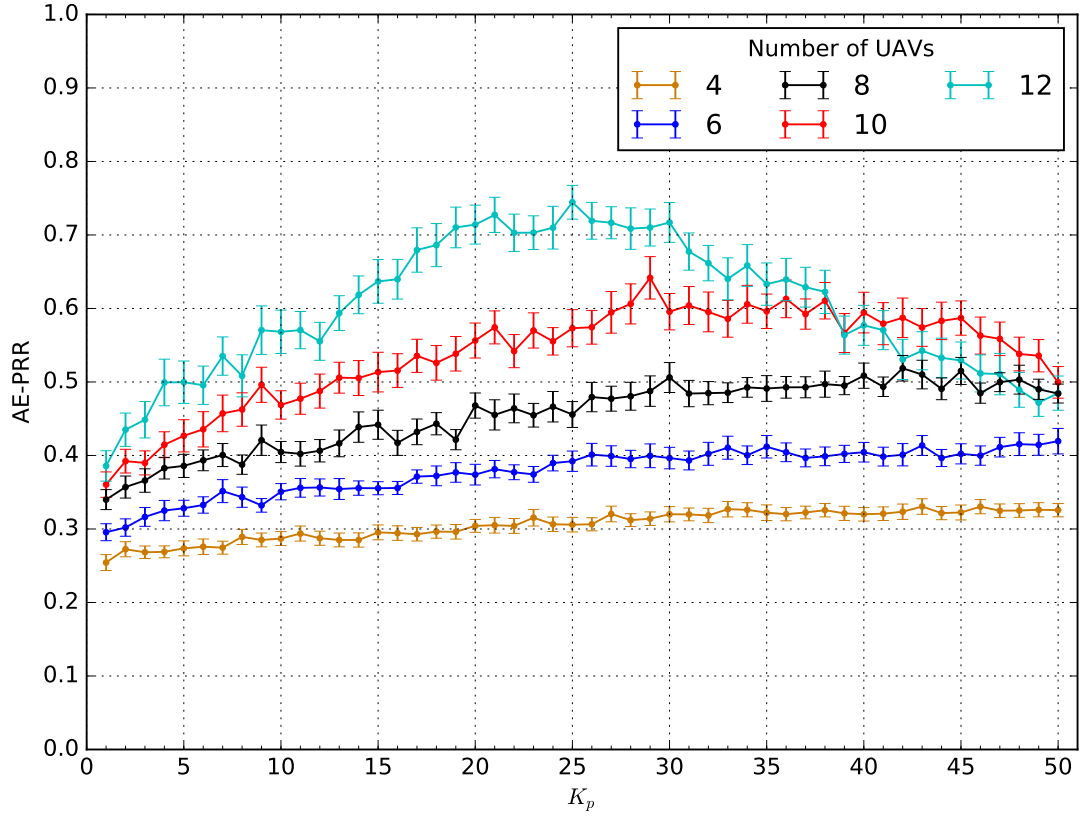


Figure 5.2: AE-PRR trend with 3 teams of 20 GNs in a scenario with parameters listed in Table 5.3.

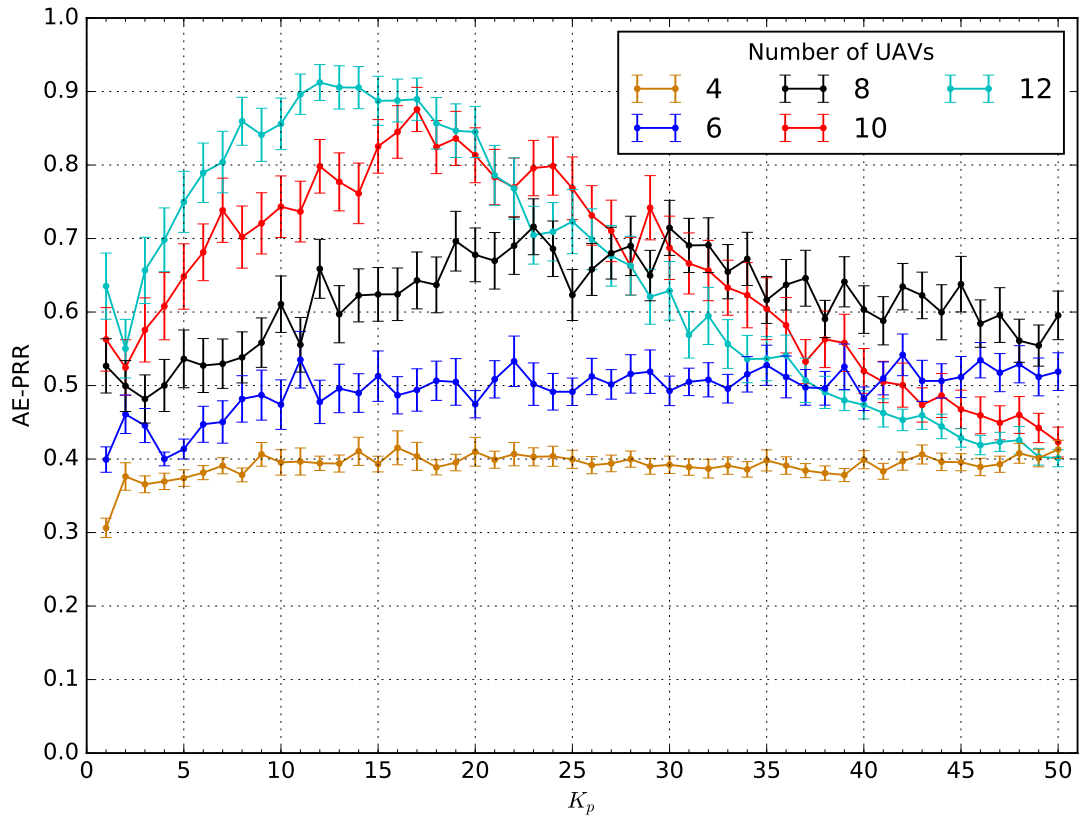


Figure 5.3: AE-PRR trend with 1 team of 20 GNs in a scenario with parameters listed in Table 5.3.

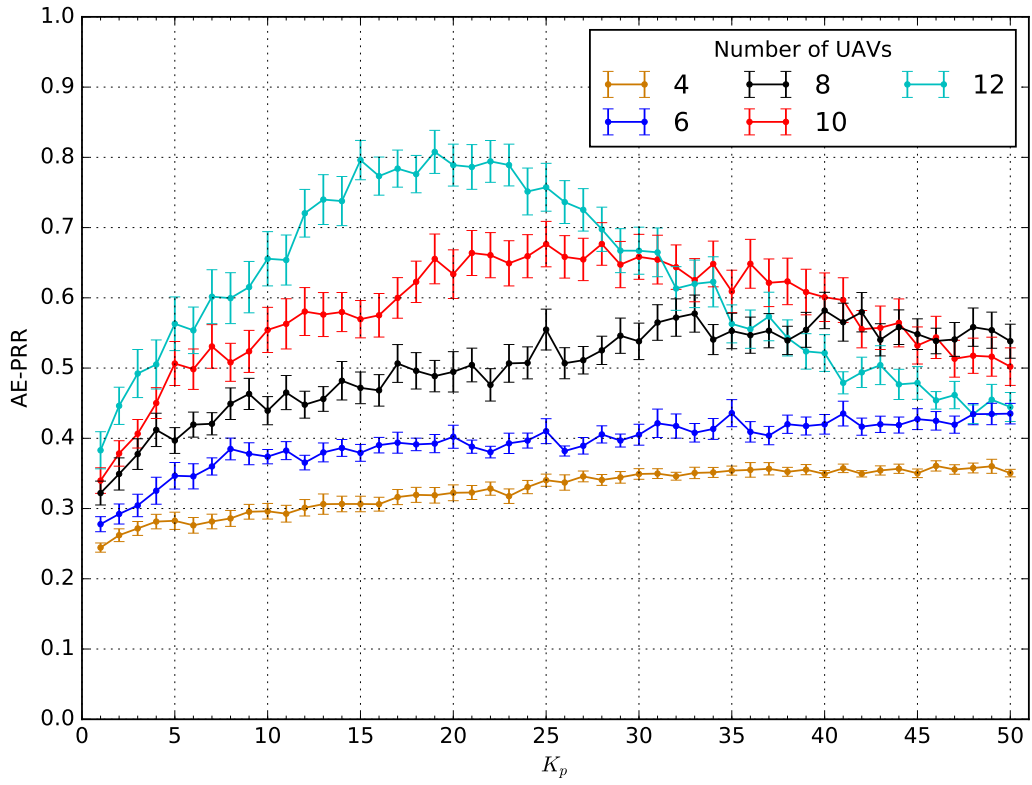


Figure 5.4: AE-PRR trend with 2 teams of 20 GNs in a scenario with parameters listed in Table 5.3.

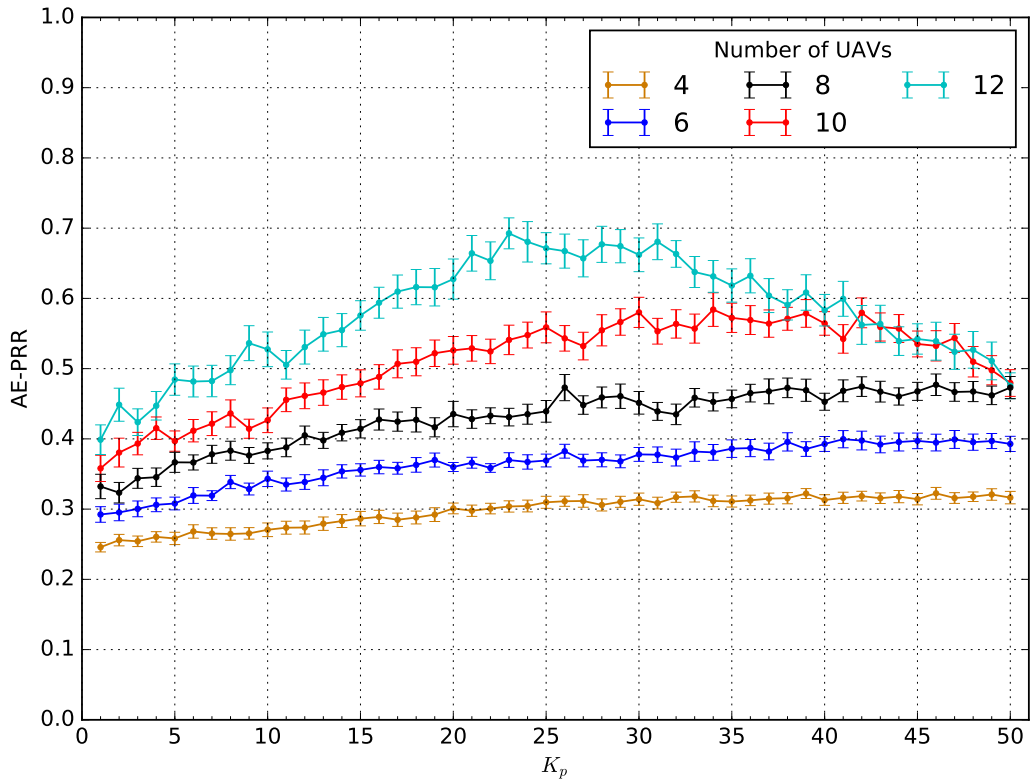


Figure 5.5: AE-PRR trend with 4 teams of 20 GNs in a scenario with parameters listed in Table 5.3.

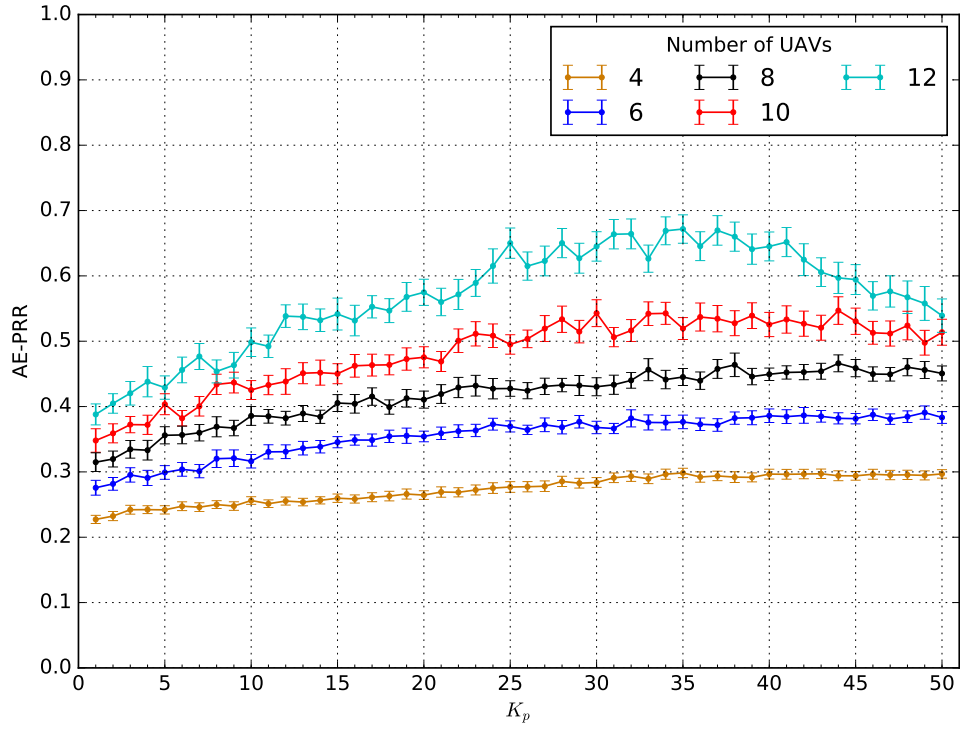


Figure 5.6: AE-PRR trend with 5 teams of 20 GNs in a scenario with parameters listed in Table 5.3.

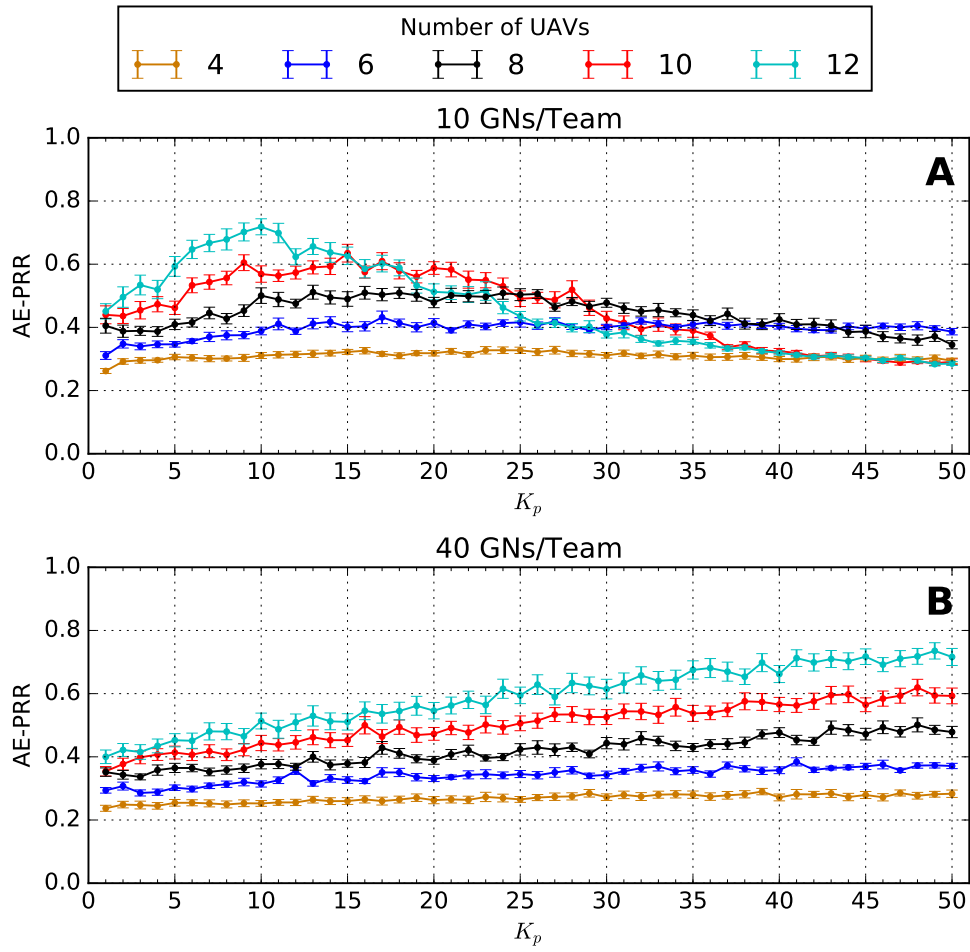


Figure 5.7: AE-PRR trend with 3 teams composed of different numbers of GNs in a scenario with parameters listed in Table 5.3.

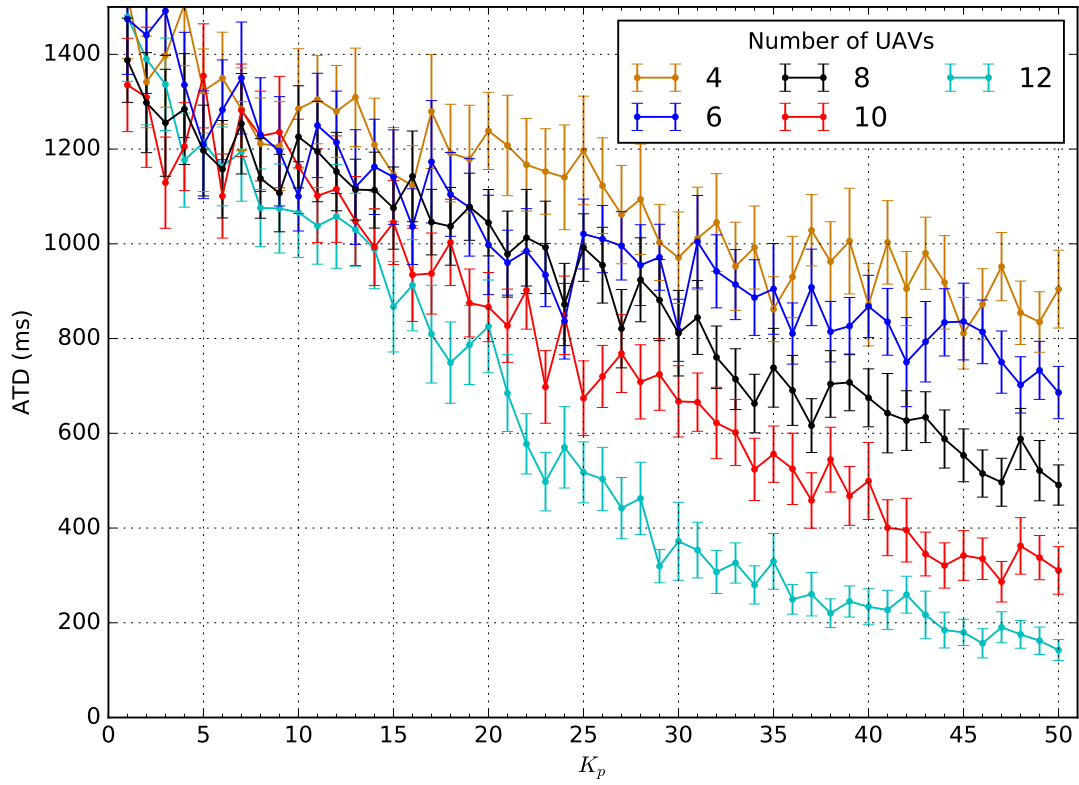


Figure 5.8: ATD trend with 3 teams of 20 GNs in a scenario with parameters listed in Table 5.3.

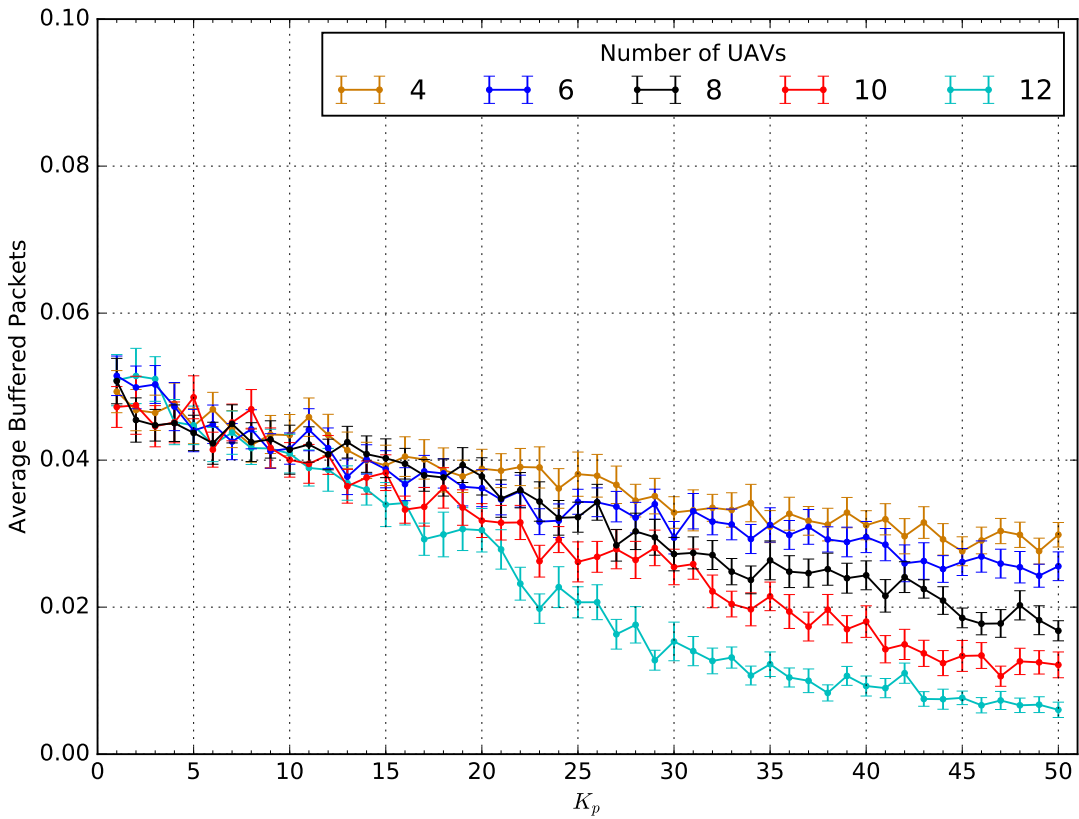


Figure 5.9: Average percentage of buffered packets in a scenario with 3 teams of 20 GNs and the parameters listed in Table 5.3.



## 5.4.2 Comparison between LoRaUAV VSF and DF VSF algorithms

This section is dedicated to the comparison between the LoRaUAV VSF algorithm developed as part of this thesis (see Section 4.3.1) and the DF VSF algorithm described in Section 3.3. Before starting the analysis, it is important to clarify that the  $K_p$  parameter coincides with the proportionality factor constituting  $K_{AtA}$  in the LoRaUAV VSF algorithm and it coincides, instead, with  $K_{AtA}$  itself in the DF VSF algorithm. The simulations have been performed using the same parameters used in Section 5.4.1 and reported in Table 5.3. A comparison between the two algorithms, in terms of AE-PRR, is shown in Figure 5.10, Figure 5.11 and Figure 5.12 for a scenario with three, four and five teams of GNs respectively. Simply by observing the plots it is possible to notice that the AE-PRR of both algorithms follows approximately the same trend for each configuration of UAVs. In fact it presents the same regions of increase and decrease and the same range of optimal values. However, the AE-PRR produced by the LoRaUAV VSF algorithm seems to be better, for most  $K_p$  values, than the one generated by the DF VSF algorithm. In order to verify this hypothesis, the curves generated by the two algorithms have been compared by computing the point by point euclidean distance between them for each simulated scenario. The minimum, maximum and average distances between the curve generated by the DF algorithm and the curve generated by the LoRaUAV algorithm, considering the whole range of  $K_p$  values, are reported in Table 5.4. It is possible to notice that, on average, LoRaUAV produces an increment of the AE-PRR which lies between 0.03 % and 4.6 % independently of the value of  $K_p$  that is used. The most relevant performance increments are experienced in scenarios with 4 or 5 teams, thus showing that the developed VSF algorithm produces more benefits when a larger number of teams is deployed. However, for most  $K_p$  values, the improvements are small and they fall inside the confidence interval of the DF VSF algorithm. Therefore, it is not possible to assert that the LoRaUAV VSF algorithm performs better in general. Small improvements can be observed locally in correspondence of some UAV configurations and for some values of  $K_p$ , but with no recognizable pattern. This is evident, for example, in Figure 5.12 for 12 UAVs and  $K_p \in [30, 40]$  where the improvement lies between 4.2 % and 7.9 % with a confidence interval of  $\approx \pm 2$  %. The effects of the algorithms have been investigated also on the ATD metric. In particular, we want to determine how the ATD behaves in the range of  $K_p$  values where the AE-PRR remains almost constant and takes values close to the observed maximum. In order to simplify the choice of the  $K_p$  range to test, only the configurations with 3, 4 and 5 teams are considered. The chosen interval is  $[25, 30]$ . In this interval the average value of the ATD is computed for the two algorithms and the percentage variation is then determined. The results are reported in Table 5.5. According to this table, in almost all the circumstances, the LoRaUAV VSF algorithm performs worse than the DF VSF algorithm. However, the ATD alone is only useful to understand the trend of the delay, but it must always be contextualized and put in perspective, since it takes into account also the time that packets spend in the buffer. In all cases, buffered packets, in the given  $K_p$  interval, are  $\approx 3\%$  of the total received packets and are responsible for all the delays bigger than a few hundred milliseconds. Buffering time is therefore responsible for the observed ATD variations, meaning that, by using the LoRaUAV algorithm, buffered packets have a higher chance of staying longer in the buffer.

As expected, the LoRaUAV VSF algorithm does not significantly outperform the DF algorithm. In fact the two algorithms perform almost identically in most of the situations. The conclusion is that changing just the weights of the forces is not sufficient, alone, to achieve a significant AE-PRR increase. However, albeit small and limited to particular configurations, the improvements introduced by our algorithm comes with no additional cost, apart from a delay penalty, correlated with the higher achieved AE-PRR, that only affects a small percentage of the packets in some situations.

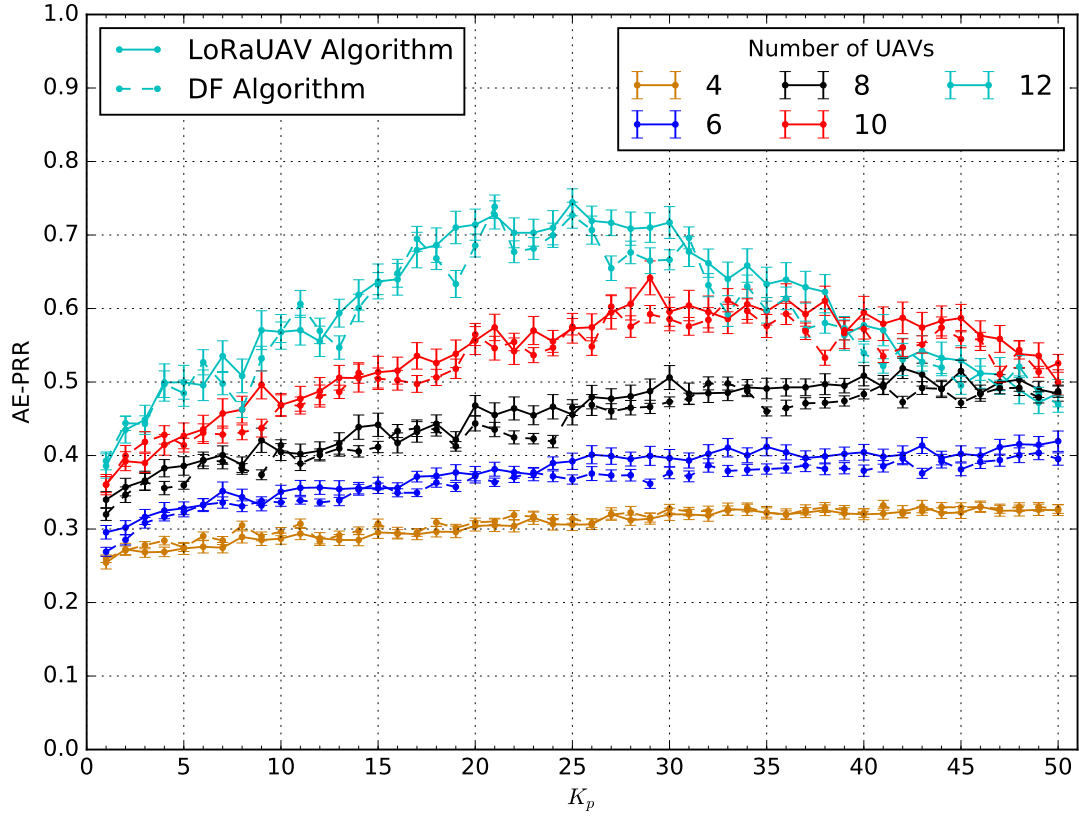


Figure 5.10: AE-PRR comparison when the LoRaUAV VSF algorithm and the DF VSF algorithm are used in a scenario with 3 teams of 20 GNs. Parameters are listed in Table 5.3.

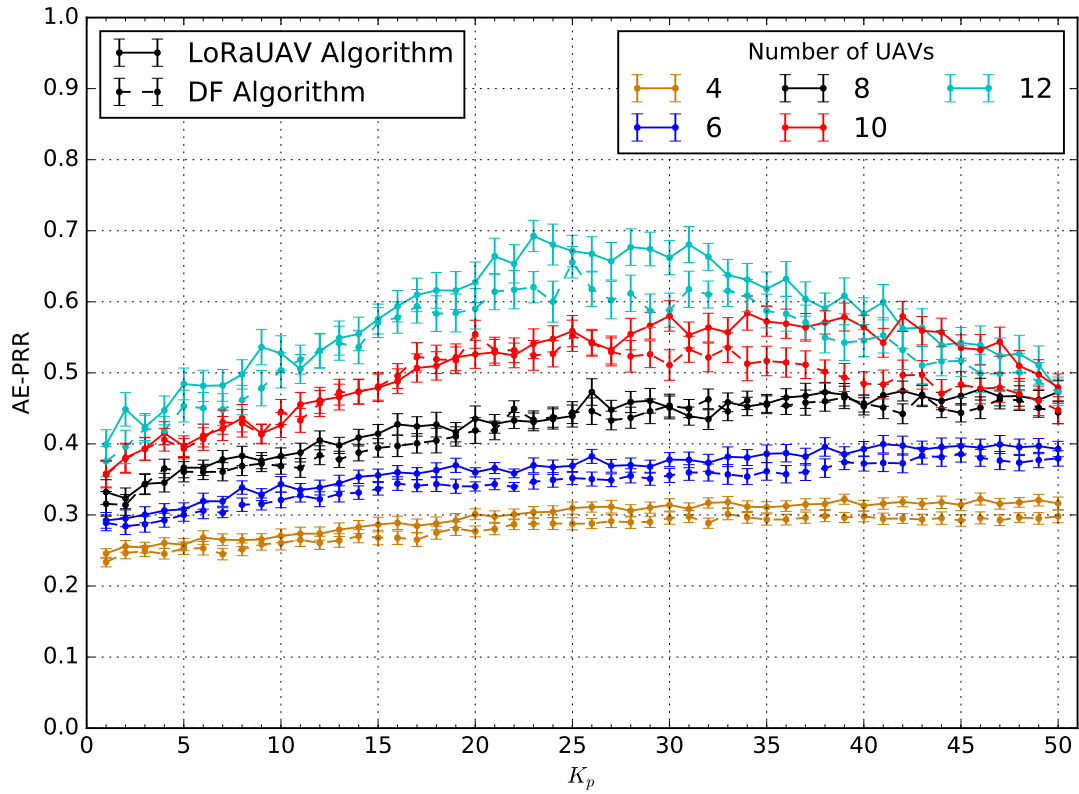


Figure 5.11: AE-PRR comparison when the LoRaUAV VSF algorithm and the DF VSF algorithm are used in a scenario with 4 teams of 20 GNs. Parameters are listed in Table 5.3.

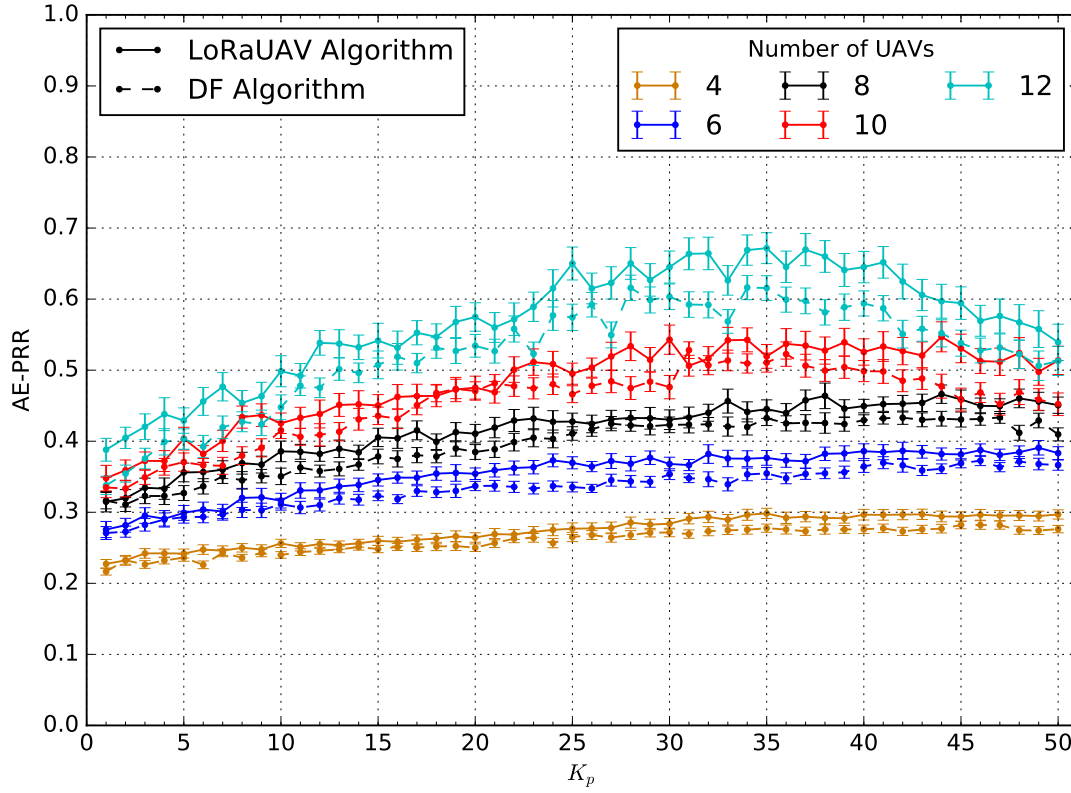


Figure 5.12: AE-PRR comparison when the LoRaUAV VSF algorithm and the DF VSF algorithm are used in a scenario with 5 teams of 20 GNs. Parameters are listed in Table 5.3.

UAVs	Teams								
	1			2			3		
	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg
4	+0.004 %	+5.1 %	+2.6 %	+0.48 %	+2.7 %	+1.7 %	-1.5 %	+0.40 %	-0.58 %
6	-2.8 %	+6.8 %	+2.2 %	+0.58 %	+4.8 %	+2.4 %	-0.43 %	+2.7 %	+1.2 %
8	-7.4 %	+11 %	+0.03 %	-1.8 %	+5.7 %	+1.4 %	-0.80 %	+3.9 %	+1.2 %
10	-2.1 %	+14 %	+3.2 %	-3.3 %	+6.0 %	+1.5 %	-2.7 %	+7.2 %	+1.7 %
12	-4.6 %	+18 %	+3.2 %	-1.1 %	+9.4 %	+2.3 %	-3.0 %	+6.4 %	+2.01 %

UAVs	Teams					
	4			5		
	Min	Max	Avg	Min	Max	Avg
4	+0.5 %	+2.8 %	+1.7 %	+0.01 %	+2.4 %	+1.4 %
6	+0.4 %	+3.2 %	+1.8 %	+0.1 %	+2.8 %	+3.6 %
8	-2.8 %	+3.2 %	+1.0 %	-0.1 %	+4.8 %	+2.2 %
10	-2.8 %	+8.6 %	+2.5 %	-2.1 %	+7.2 %	+2.9 %
12	-1.3 %	+8.7 %	+3.5 %	+1.3 %	+7.9 %	+4.6 %

Table 5.4: AE-PRR point by point minimum, maximum and average variation between the LoRaUAV VSF algorithm and the DF VSF algorithm.

UAVs	Teams								
	3			4			5		
	DF	LoRaUAV	Var	DF	LoRaUAV	Var	DF	LoRaUav	Var
4	1011	1075	+6.3 %	1154	942	-18 %	1301	992	-24 %
6	929	962	+3.6 %	969	894	-7.7 %	1020	857	-16 %
8	869	897	+3,2 %	768	818	+6.5 %	745	800	+7.4 %
10	637	710	+11 %	539	673	+25 %	552	721	+31 %
12	448	436	-2.7 %	442	509	+15 %	504	645	+28 %

Table 5.5: Average value of the ATD in milliseconds for the LoRaUAV VSF algorithm and the DF algorithm and relative variation obtained for  $K_p \in [25 - 30]$ .

## 5.5 Study of the impact of the CRM algorithm

One of the main additions to the system is the CRM algorithm detailed in Section 4.3.2. This algorithm has been developed with the explicit purpose of incrementing the coverage of the UAV mesh network, without increasing the number of UAVs. In fact, the AE-PRR obtained with just the VSF algorithm, as shown in Figures 5.2 to 5.6, is affected by an high percentage of lost packets (more than 20 %) in all the considered situations, mainly because of the absence of covering UAVs. Depending on the QoS requirements imposed by the mission, this might be a major drawback of the LoRaUAV VSF algorithm operating alone. In the absence of other mechanisms, this might only be mitigated by reducing the monitored area or by adding more UAVs to the system. The CRM algorithm tries to solve the aforementioned issues through a better spacial distribution of the available UAVs thanks to the combination of a pause behaviour and a persist behaviour. In this section, it is shown that, in terms of relevant QoS metrics, the CRM algorithm effectively improves the performance of the system in comparison with the basic LoRaUAV VSF algorithm operating alone, but with some potential drawbacks. Simulations have been performed according to the settings reported in Table 5.6.

Parameter	Values
Number of UAVs	(4, 6, 8, 10, 12)
Number of teams	(1, 2, 3, 4, 5)
GNs per team	20
Total simulation time	3000 s
Simulation area	2500 × 2500 m

Table 5.6: Parameters used for simulations described in Section 5.5.

With respect to the simulation performed in Section 5.4.1, the simulation area has been incremented to better observe the performance increment introduced by the CRM algorithm. As a consequence, also the simulation time has been increased to 3000 s. Only a subset of  $K_p$  values is tested. This subset coincides with the range of  $K_p$  values that approximately maximizes the AE-PRR according to the results obtained in Section 5.4.1. A visual comparison of a sample scenario with three teams of GNs is plotted in Figure 5.13. Table 5.7 reports the difference between the AE-PRR with and without CRM for some values of  $K_p$  and for all the possible configurations. In all the situations, the CRM algorithm significantly improves the AE-PRR, with most of the increments lying between 14 % and 26 % with peaks of more

than 50 %. In absolute terms, the AE-PRR achieved in the considered scenarios is low. In Figure 5.13, it is possible to see that more than 30 % of the packets are lost. It has to be taken into account, though, that the size of the monitored area has been chosen in order to stress the tested algorithms. If we consider more appropriate simulation settings, like the ones used in Section 5.4.1, the results are more encouraging. Figure 5.14 shows the trend of the AE-PRR with the same configuration used in Section 5.4.1 with 60 GNs divided in three teams. In this scenario, the AE-PRR with 12 UAVs reaches almost 0.9 and good results are obtained even with 8 or 10 UAVs, with a considerable performance increment if compared with the results, obtained under the same conditions, shown in Figure 5.2. Selecting the number of UAVs to optimize the coverage of a certain area is a non trivial problem and deserves a more in-depth study in future evolutions of this work.

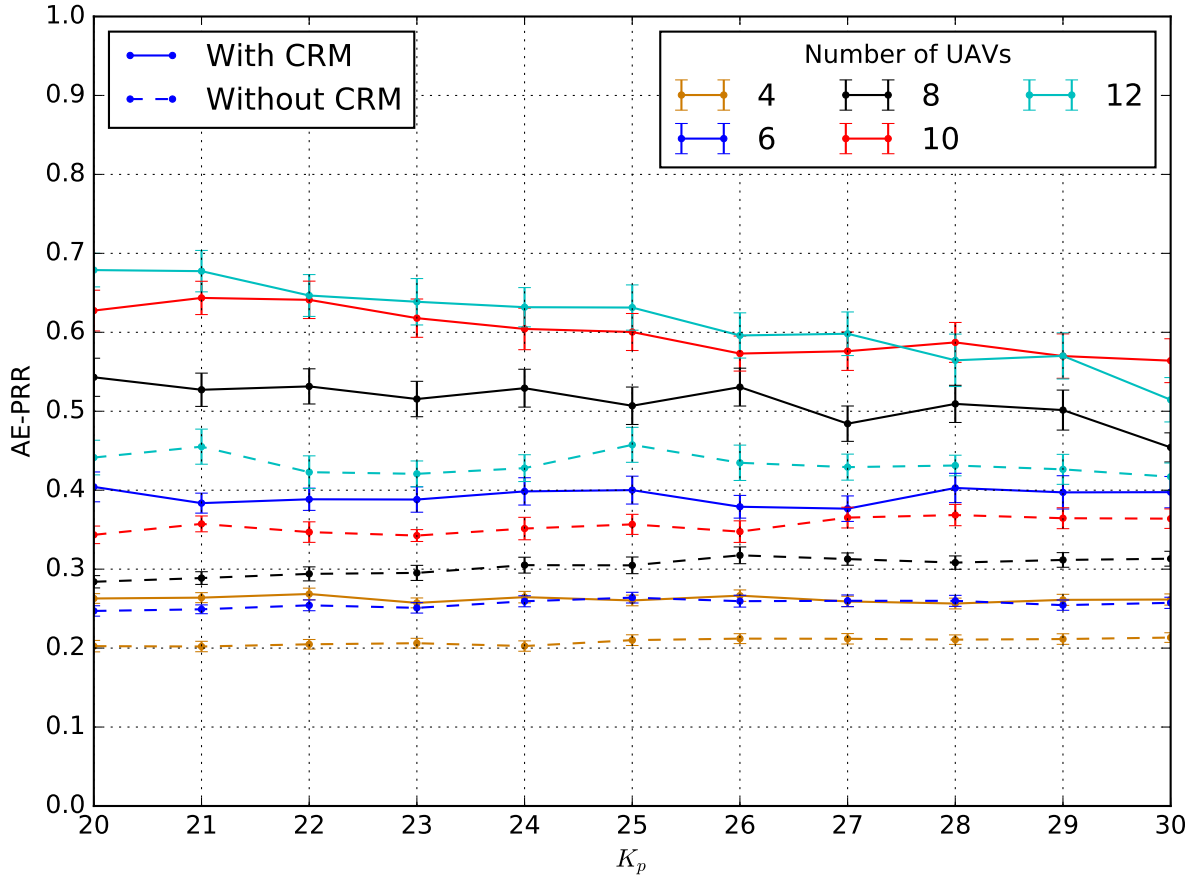


Figure 5.13: AE-PRR comparison when CRM algorithm is activated in a scenario with 3 teams of 20 GNs and parameters listed in Table 5.6.

The ATD seems to be the QoS metric that is most negatively affected by the CRM algorithm, as it is possible to see in Figure 5.15. However, as stated also in Section 5.4.2, a deeper analysis shows that the ATD metric is greatly influenced by a small number of packets with very large delays. In fact, the CRM algorithm allows by design for more and longer disconnections in order to achieve a higher PRR. Some packets may therefore stay in the buffer for a very long time before being sent. To prove this hypothesis and to evaluate its impact, 10 individual simulations have been run for a scenario having 12 UAVs, 3 teams of GNs and  $K_p = 25$  and the total individual delays of all the successfully received packets have been recorded. Results are reported in Table 5.8. It is immediately clear that the percentage of buffered packets more than doubles when the CRM algorithm is active, meaning that disconnections are longer and/or more frequent. Also the permanence in the buffer drastically increases with the CRM algorithm,

thus proving that huge delays are only caused by buffered packets. In fact, if buffered packets are not considered, the average experienced delay is  $\approx 55$  ms, a value that is far lower than the reported ATD values. In conclusion, in the face of a double digit rise of the AE-PRR, the increase of the buffer time for a reduced percentage of packets seems to be a good trade-off. This shows the goodness of the CRM algorithm and demonstrates its importance in the LoRaUAV system.

UAVs	Teams				
	$K_p = 13$	$K_p = 14$	$K_p = 15$	$K_p = 16$	$K_p = 17$
4	$0.21 \pm 0.01$	$0.21 \pm 0.01$	$0.20 \pm 0.01$	$0.21 \pm 0.01$	$0.19 \pm 0.01$
6	$0.54 \pm 0.03$	$0.52 \pm 0.03$	$0.50 \pm 0.03$	$0.50 \pm 0.03$	$0.48 \pm 0.03$
8	$0.58 \pm 0.01$	$0.56 \pm 0.02$	$0.52 \pm 0.02$	$0.54 \pm 0.02$	$0.54 \pm 0.02$
10	$0.43 \pm 0.04$	$0.40 \pm 0.04$	$0.45 \pm 0.03$	$0.42 \pm 0.03$	$0.37 \pm 0.04$
12	$0.29 \pm 0.04$	$0.28 \pm 0.05$	$0.26 \pm 0.05$	$0.32 \pm 0.04$	$0.19 \pm 0.05$
2					
	$K_p = 20$	$K_p = 21$	$K_p = 22$	$K_p = 23$	$K_p = 24$
4	$0.09 \pm 0.01$	$0.07 \pm 0.01$	$0.08 \pm 0.01$	$0.08 \pm 0.01$	$0.08 \pm 0.01$
6	$0.17 \pm 0.02$	$0.18 \pm 0.02$	$0.17 \pm 0.02$	$0.19 \pm 0.02$	$0.16 \pm 0.02$
8	$0.30 \pm 0.03$	$0.27 \pm 0.03$	$0.27 \pm 0.03$	$0.29 \pm 0.03$	$0.26 \pm 0.03$
10	$0.34 \pm 0.03$	$0.30 \pm 0.03$	$0.28 \pm 0.03$	$0.26 \pm 0.03$	$0.24 \pm 0.03$
12	$0.22 \pm 0.04$	$0.21 \pm 0.03$	$0.14 \pm 0.03$	$0.15 \pm 0.04$	$0.11 \pm 0.03$
3					
	$K_p = 23$	$K_p = 24$	$K_p = 25$	$K_p = 26$	$K_p = 27$
4	$0.051 \pm 0.007$	$0.062 \pm 0.008$	$0.050 \pm 0.008$	$0.054 \pm 0.008$	$0.047 \pm 0.008$
6	$0.14 \pm 0.01$	$0.14 \pm 0.02$	$0.14 \pm 0.02$	$0.12 \pm 0.01$	$0.12 \pm 0.02$
8	$0.22 \pm 0.02$	$0.22 \pm 0.02$	$0.20 \pm 0.02$	$0.21 \pm 0.02$	$0.17 \pm 0.02$
10	$0.28 \pm 0.02$	$0.25 \pm 0.02$	$0.24 \pm 0.02$	$0.23 \pm 0.02$	$0.21 \pm 0.02$
12	$0.22 \pm 0.03$	$0.20 \pm 0.03$	$0.17 \pm 0.03$	$0.16 \pm 0.03$	$0.17 \pm 0.03$
4					
	$K_p = 20$	$K_p = 21$	$K_p = 22$	$K_p = 23$	$K_p = 24$
4	$0.06 \pm 0.01$	$0.06 \pm 0.01$	$0.06 \pm 0.01$	$0.06 \pm 0.01$	$0.06 \pm 0.01$
6	$0.13 \pm 0.01$	$0.14 \pm 0.01$	$0.12 \pm 0.01$	$0.12 \pm 0.01$	$0.11 \pm 0.01$
8	$0.19 \pm 0.01$	$0.18 \pm 0.02$	$0.17 \pm 0.02$	$0.19 \pm 0.02$	$0.20 \pm 0.02$
10	$0.27 \pm 0.02$	$0.28 \pm 0.02$	$0.25 \pm 0.02$	$0.22 \pm 0.02$	$0.25 \pm 0.02$
12	$0.31 \pm 0.02$	$0.31 \pm 0.02$	$0.28 \pm 0.02$	$0.28 \pm 0.02$	$0.27 \pm 0.02$
5					
	$K_p = 30$	$K_p = 31$	$K_p = 32$	$K_p = 33$	$K_p = 34$
4	$0.03 \pm 0.01$	$0.04 \pm 0.01$	$0.04 \pm 0.01$	$0.03 \pm 0.01$	$0.03 \pm 0.01$
6	$0.07 \pm 0.01$	$0.08 \pm 0.01$	$0.08 \pm 0.01$	$0.07 \pm 0.01$	$0.07 \pm 0.01$
8	$0.12 \pm 0.01$	$0.13 \pm 0.02$	$0.13 \pm 0.02$	$0.12 \pm 0.02$	$0.10 \pm 0.02$
10	$0.17 \pm 0.02$	$0.16 \pm 0.02$	$0.16 \pm 0.02$	$0.15 \pm 0.02$	$0.14 \pm 0.02$
12	$0.15 \pm 0.02$	$0.16 \pm 0.02$	$0.15 \pm 0.02$	$0.14 \pm 0.02$	$0.16 \pm 0.02$

Table 5.7: Difference between AE-PRR with and without CRM and relative confidence interval for some values of  $K_p$ .

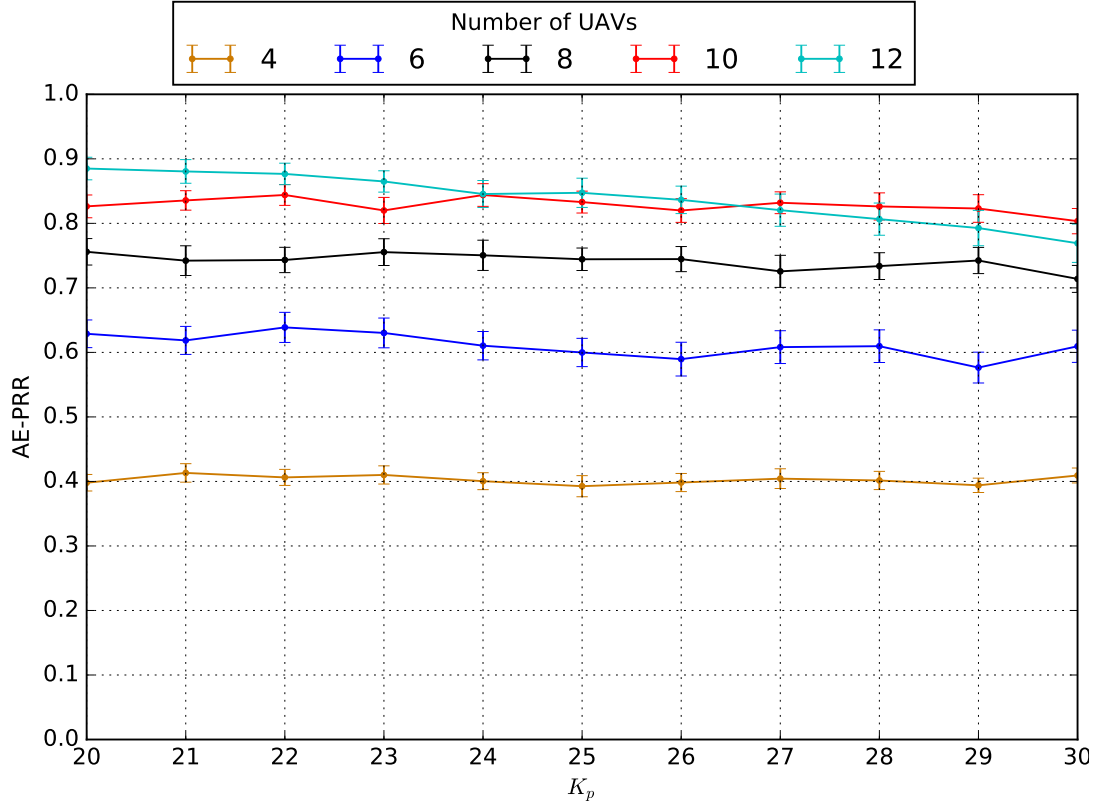


Figure 5.14: AE-PRR obtained with VSF + CRM in scenario configured as reported in Table 5.3.

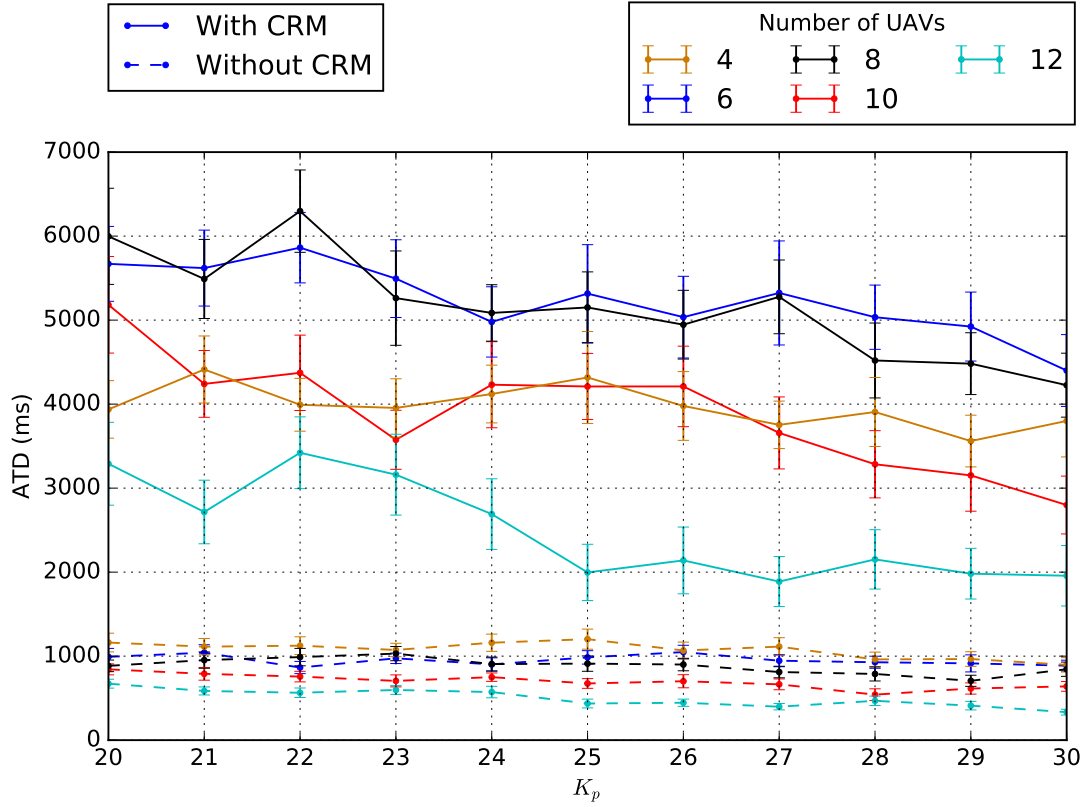


Figure 5.15: ATD comparison when CRM algorithm is activated in a scenario with 3 teams of 20 GNs and parameters listed in Table 5.6.

Run	Buffered Packets		Avg Buffer Delay (s)		Run	Buffered Packets		Avg Buffer Delay (s)	
	w/o CRM	w/ CRM	w/o CRM	w/ CRM		w/o CRM	w/ CRM	w/o CRM	w/ CRM
0	2.1 %	4.1 %	34	30	5	2.7 %	3.5 %	18	83
1	1.8 %	11 %	24	54	6	2.5 %	6.4 %	13	40
2	2.8 %	6.7 %	20	37	7	2.3 %	5.5 %	13	131
3	4.2 %	4.2 %	15	34	8	3.1 %	8.2 %	20	37
4	3.9 %	11 %	20	50	9	1.2 %	7.5 %	12	62

Table 5.8: Buffered Packets and Average Buffer Delay with and without CRM algorithm for a scenario with 12 UAVs, 3 teams of GNs and  $K_p = 25$ .

## 5.6 Study of the impact of the MP algorithm

This section has the objective of determining if and how the MP algorithm, described in detail in Section 4.3.3, affects LoRaUAV. In particular, the interest is all in the PRR, since the MP algorithm represents the last attempt of the system to re-establish the connection between the aerial mesh and GNs that went out of range. However, it is worth noticing that the algorithm, given its distributed nature, is triggered whenever a group of GNs goes out of range of every single UAV, even if the very same group happens to be in the range of a neighbouring UAV. This characteristic may prove to be useful not only when a team of GNs becomes isolated from the mesh, but even to better distribute the UAVs inside the mesh. Testing the impact of the MP algorithm is particularly challenging, mostly because the mechanism, given its own nature, might not be fully exploited in all the situations. In order to increase the triggering chances, the firemen mobility has been extended to support the splitting of teams. With this modification, a team can be configured to split in half during the simulation. In this way, a UAV that is covering alone a team of GNs is forced to create at least one hologram when half of the team eventually goes out of range. In order to avoid to separate a team before the MP algorithm has collected enough positioning data, the split happens only halfway through the simulation. Given this modification, it is now of particular interest to observe the behaviour of the MP algorithm in two situations: a more conventional setting, where team splitting is not active, and in a more challenging setting where teams can split. Two comparisons are made:

- I. LoRaUAV with VSF + MP algorithms and LoRaUAV with just the basic VSF algorithm;
- II. LoRaUAV with VSF + CRM + MP algorithms and LoRaUAV with just VSF + CRM algorithms.

Unlike the other sections of this chapter, the end-to-end packet reception rate is not averaged over multiple runs and it is instead collected individually for all the 100 runs of the same scenario. A direct comparison between corresponding runs of different scenarios is then performed and significant summary metrics are extracted. The used simulation parameters are reported in Table 5.9. The results of the comparison are reported in Table 5.10, Table 5.11, Table 5.12 and Table 5.13. The tables report, in the first column, the percentage of runs that performed better with the MP algorithm and the percentage of runs that performed better without the MP algorithm. The second column reports the average gain obtained in those runs where the MP algorithm improved the PRR and the average loss obtained in those runs where the MP algorithm worsened the PRR.

The first observation that can be made is that the MP algorithm is not always beneficial to the system. In fact, a relevant number of runs shows a PRR that is, on average, even 8 % lower than the one measured when the same scenarios are run without the MP algorithm. The loss is more marked in scenarios in which teams do not split. The reasons of these losses have been investigated by visual inspection of those scenarios that present the most marked performance loss. The conclusion is that



Parameter	Values
Number of UAVs	(4, 6, 8, 10, 12)
Number of teams	3
GNs per team	20
$K_p$	25
Total simulation time	3000 s
Simulation area	2500 × 2500 m

Table 5.9: Parameters used for simulations described in Section 5.6.

UAVs	Runs		Avg MP PRR gain/loss	
	Better w/ MP	Better w/o MP	Better w/ MP	Better w/o MP
4	40 %	60 %	+1.15 %	-0.9 %
6	38 %	62 %	+2.5 %	-1.4 %
8	42 %	58 %	+4.3 %	-2.7 %
10	45 %	55 %	+5.2 %	-5.5 %
12	47 %	53 %	+11.0 %	-8.0 %

Table 5.10: Comparison between runs with VSF + MP and runs with only VSF when team splitting is not active.

UAVs	Runs		Avg MP PRR gain/loss	
	Better w/ MP	Better w/o MP	Better w/ MP	Better w/o MP
4	57 %	43 %	+1.1 %	-1.9 %
6	51 %	49 %	+3.7 %	-3.9 %
8	53 %	47 %	+5.5 %	-4.3 %
10	56 %	44 %	+6.2 %	-4.9 %
12	64 %	36 %	+8.9 %	-5.8 %

Table 5.11: Comparison between runs with VSF + MP and runs with only VSF when team splitting is active.

UAVs	Runs		Avg MP PRR gain/loss	
	Better w/ MP	Better w/o MP	Better w/ MP	Better w/o MP
4	58 %	42 %	+2.1 %	-1.5 %
6	53 %	47 %	+9.0 %	-7.1 %
8	66 %	34 %	+13 %	-6.6 %
10	77 %	23 %	+15 %	-7.0 %
12	86 %	14 %	+18 %	-4.0 %

Table 5.12: Comparison between runs with VSF + CRM + MP and runs with only VSF + CRM when team splitting is not active.

UAVs	Runs		Avg MP PRR gain/loss	
	Better w/ MP	Better w/o MP	Better w/ MP	Better w/o MP
4	78 %	22 %	+4.3 %	-2.4 %
6	71 %	29 %	+8.9 %	-5.9 %
8	69 %	31 %	+11 %	-5.8 %
10	78 %	22 %	+11 %	-5.2 %
12	91 %	9 %	+15 %	-1.9 %

Table 5.13: Comparison between runs with VSF + CRM + MP and runs with only VSF + CRM when team splitting is active.

the strategy of recovering the connectivity with lost GNs is not always beneficial to the UAV formation since it does not follow any global optimality criteria. For example, some UAVs may steer toward a region where there are some lost GNs, not knowing in advance if such connection can be restored, even if it would be more beneficial to just ignore those GNs and proceed without using the MP algorithm. Considered this general observation, it is immediately noticeable that using the MP algorithm without the CRM algorithm does not bring any advantage to the system, since in all the cases approximately half of the runs show an improvement while the other half show a loss of almost the same magnitude. On the other side, setting aside some inevitable losses for the reason explained before, the simultaneous activation of the MP algorithm and of the CRM algorithm is beneficial to the system in most of the circumstances, except when 4 or 6 UAVs are used in scenarios with no team splitting. In fact, the average PRR undergoes a considerable increment in at least 66 % of the runs when teams do not split and 71 % of the runs when teams split. In some situations, the gain that is obtained is more than double of the loss observed in analogous scenarios without MP algorithm. If we take into consideration the simulation parameters reported in Table 5.3 in Section 5.4.1, the activation of the MP algorithm and of the CRM algorithm produces the results shown in Figure 5.16. Even if small, the improvements to the AE-PRR allow to achieve levels above 0.9 with 12 UAVs and between 0.8 and 0.9 with 8 or 10 UAVs, thus further improving the results shown in Figure 5.14. Following all the aforementioned considerations, it is possible to conclude that the synergy between the CRM algorithm and the MP algorithm greatly benefits the system, especially in more challenging situations where the formations of GNs may change over time.

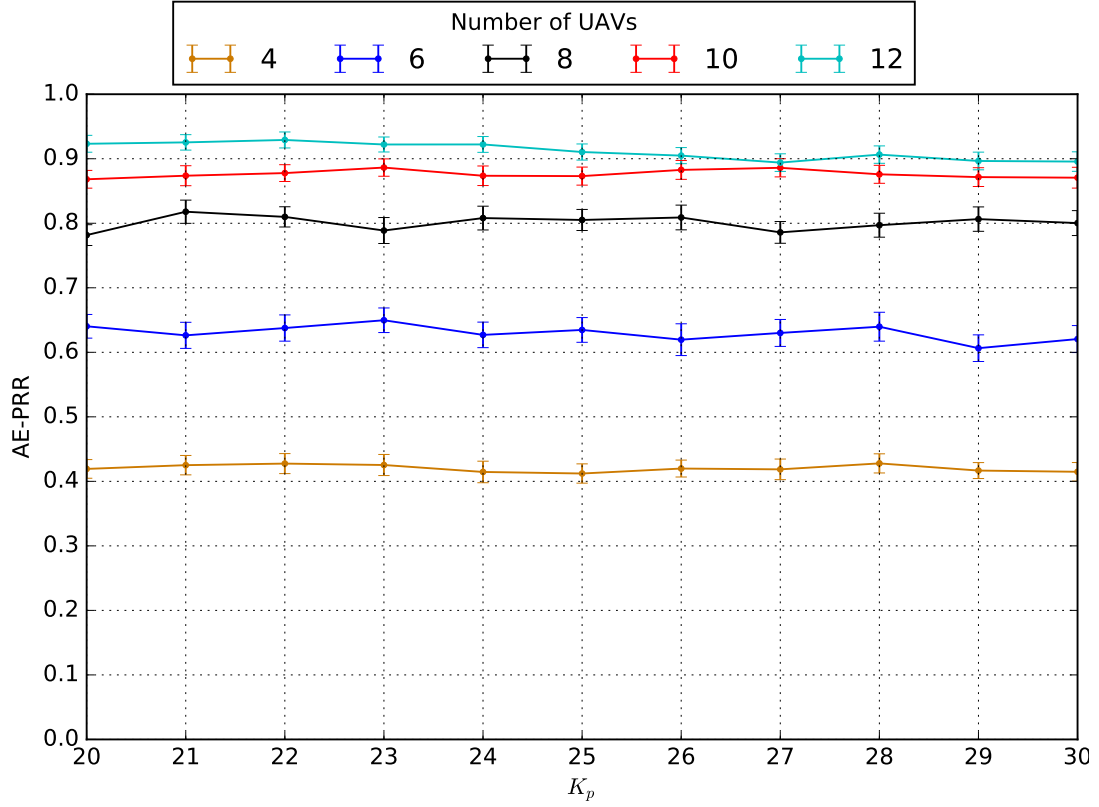


Figure 5.16: AE-PRR with VSF+CRM+MP (no team splits) in scenario with parameters of Table 5.3.

## 5.7 Frequency and Duration of Disconnections

LoRaUAV is subject to the inevitable disconnection of some GNs. In this section, the frequency and duration of such disconnections is investigated and analysed. The simulation parameters are reported in Table 5.14. The number of UAVs and the number of teams are chosen in continuity with the reference parameters used in the previous sections of this chapter.

Parameter	Values
Number of UAVs	12
Number of teams	3
GNs per team	20
$K_p$	25
Total simulation time	3000 s
Simulation area	2000 × 2000 m

Table 5.14: Parameters used for simulations described in Section 5.7.

The simulation script has been adapted to periodically check the GNs that are covered by each UAV. This information is then used to compute, for each GN, the number and duration of disconnection periods. The results of 100 runs are aggregated and reported in Figures 5.17 to 5.20 for different combinations of algorithms. In order to better visualize the results, disconnection periods are grouped in bins of different lengths and colors are used to distinguish between different teams. The first observation that can be made is that all the algorithms are subject to partitions, so that GNs might get

isolated even for long periods of time. This is particularly evident when the VSF algorithms are used alone. In fact, in both Figure 5.17 and Figure 5.18, the number of disconnections above 500 seconds constitutes a big share of the total experienced disconnections. In this regard, the DF VSF algorithm seems to experience less disconnections than the LoRaUAV VSF algorithm. This can be explained by the reduced coverage offered by the DF algorithm and the consequent presence of fewer but longer disconnection periods. In fact, a more in depth analysis of the disconnection periods larger than 500 seconds of the DF VSF algorithm reveals that several GNs remain isolated for more than 2500 seconds, while, with the LoRaUAV VSF, isolation periods are no longer than 2000 seconds, meaning that the mesh is able to cover the GNs for a longer time before losing them definitely. Nonetheless, both VSF algorithms show low performances and a high chance of almost permanent isolation of GNs. This claim is supported by the low AE-PRR results obtained by the VSF algorithms in all the previous sections of this chapter. The CRM algorithm improves the situation. One first observation is that the algorithm is fairer towards teams. In fact, disconnection periods of the same duration are approximately uniformly distributed among teams. This is caused by a better average distribution of the UAV mesh. The second observation is that disconnections are more frequent if compared with the LoRaUAV VSF algorithm operating alone. However, disconnections are shorter and the number of disconnection periods above 500 seconds is reduced by  $\approx 1000$  units overall, with team 0 and team 1 experiencing the most marked improvements. This means that, albeit partitions are still possible, they either happen much later in the simulation or they are recovered sooner by the system. The introduction of the MP algorithm on top of the CRM algorithm gives some interesting results. First, the frequency of shorter disconnection periods increases in comparison with the results obtained with just the CRM algorithm, especially in the interval  $[0, 30]$  seconds. This increase can be explained by the numerous connection recovery attempts, some of them successful, others only partially successful because the connection is lost again after some time. The disconnection periods above 500 seconds remains approximately the same obtained with the CRM algorithm, but with a huge disproportion towards team 2 members. A visual inspection of the runs where long disconnection periods are experienced showed that team 2 tends toward a region of the map that is more difficult to access for the aerial mesh. Disconnections of the other teams are solved, thanks to the MP algorithm, earlier in the simulation. This leads the UAV mesh to steer towards the location of team 0 and team 1, thus making recovery of team 2 less likely. As stated also in Section 5.6, the MP algorithm suffers from the lack of global optimality criteria that can better direct the connection recovery efforts.

All the aforementioned observations are confirmed by the total average disconnection time, computed as the average of the sums of all disconnection periods of all the performed runs. Values are reported in Table 5.15 for all the algorithms. It can be noted that, on average, the activation of the CRM and of the MP algorithms effectively reduces almost by half the total experienced disconnection time in comparison with both the VSF algorithms. The VSF + CRM and the VSF + CRM + MP configurations perform almost identically. However, if we separate the total average disconnection times of the different teams, it can be seen that, while in the configuration with just the CRM the average disconnection time is split almost equally among teams ( $\approx 4000$  s), the configuration with the MP algorithm favours team 0 and team 1 ( $\approx 2200$  s) at the expense of team 2 ( $\approx 7500$  s) for the reasons already explained before in this section.

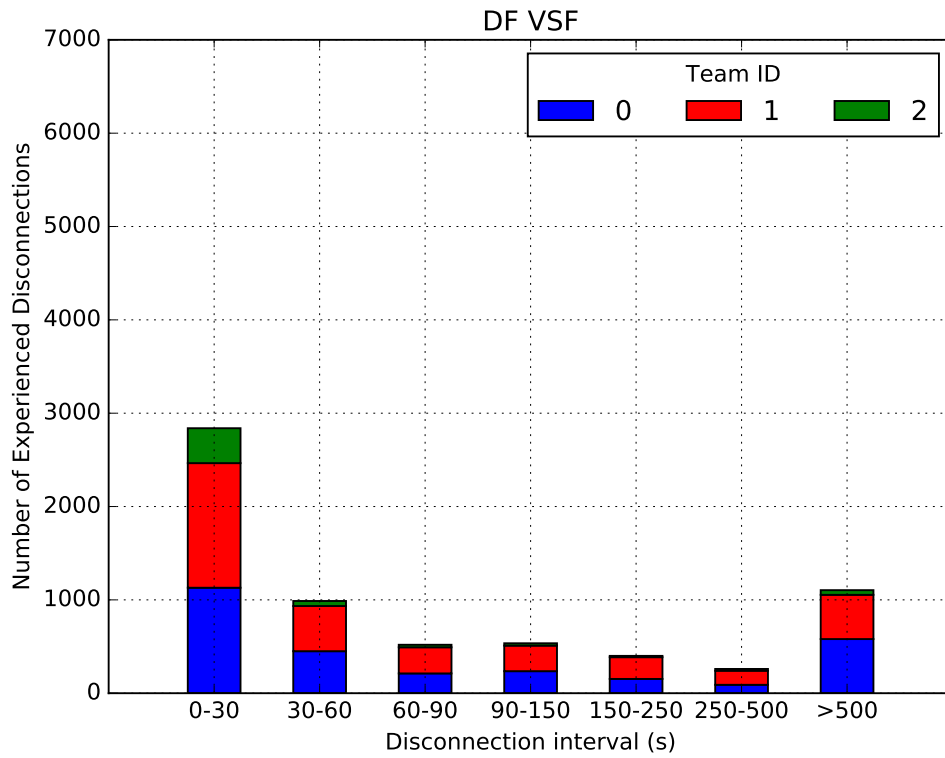


Figure 5.17: Frequency and duration of disconnections of 100 runs of a scenario using the DF VSF algorithm and with parameters listed in Table 5.14.

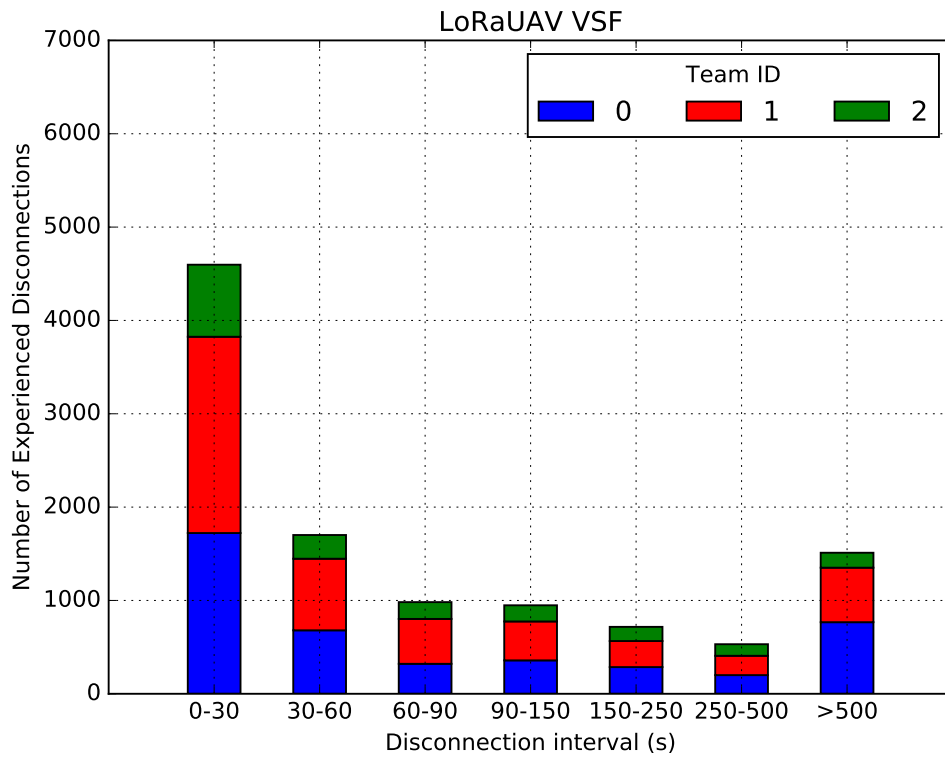


Figure 5.18: Frequency and duration of disconnections of 100 runs of a scenario using the LoRaUAV VSF algorithm and with parameters listed in Table 5.14.

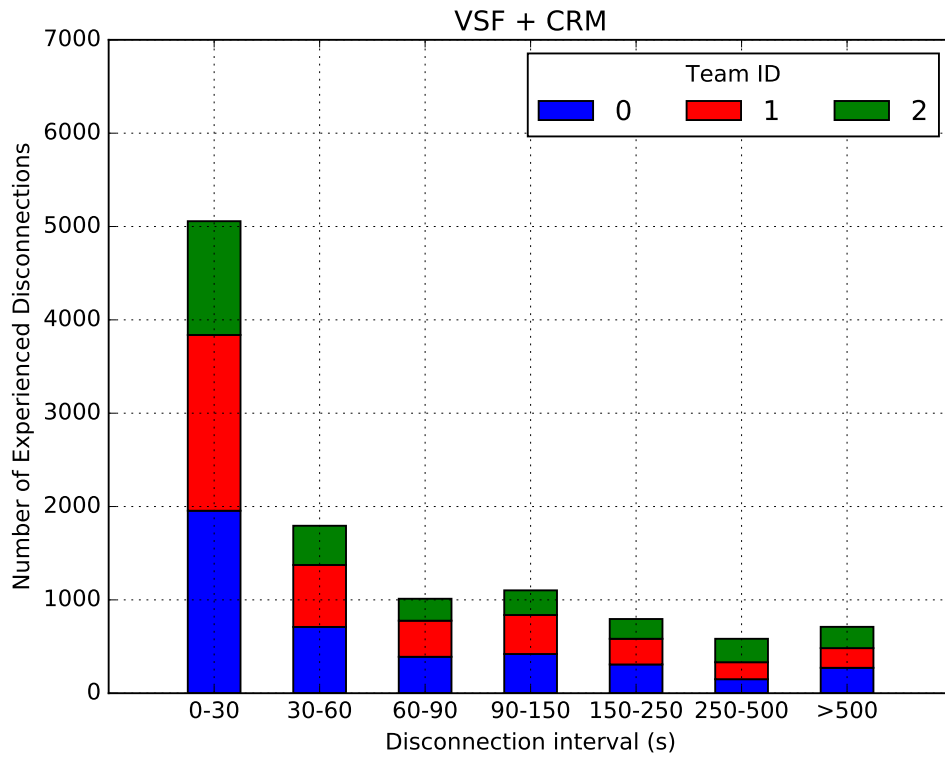


Figure 5.19: Frequency and duration of disconnections of 100 runs of a scenario using the LoRaUAV VSF + CRM algorithms and with parameters listed in Table 5.14.

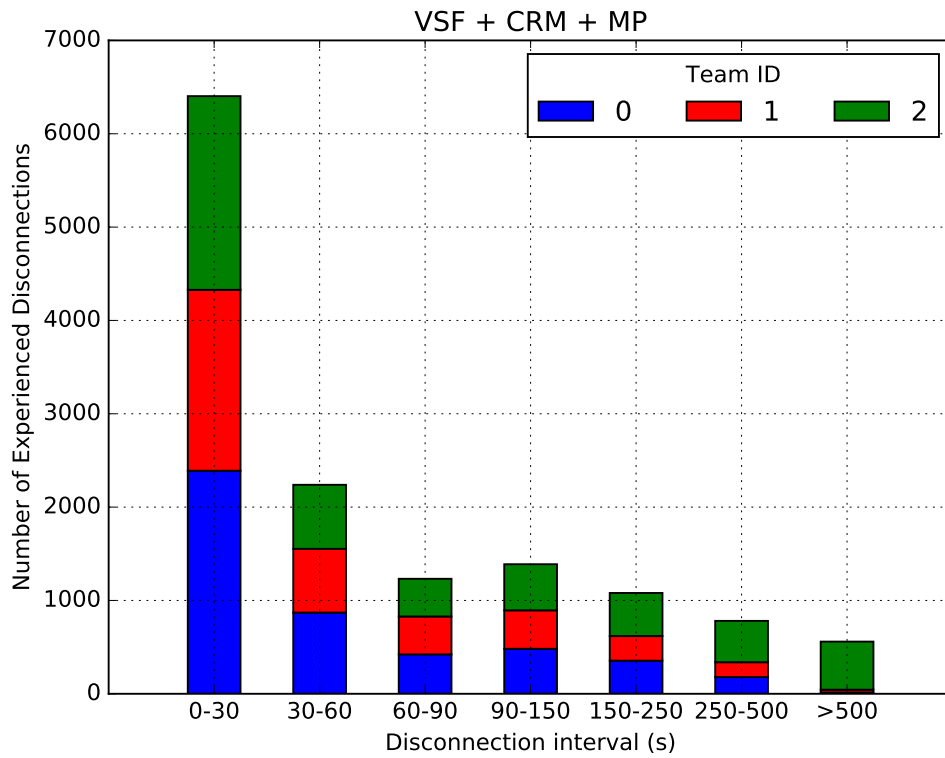


Figure 5.20: Frequency and duration of disconnections of 100 runs of a scenario using the LoRaUAV VSF + CRM + MP algorithms and with parameters listed in Table 5.14.

Algorithm	Total Average Disconnection Time (s)
DF VSF	$23199 \pm 3350$
LoRaUAV VSF	$21826 \pm 2976$
LoRaUAV VSF + CRM	$12365 \pm 2258$
LoRaUAV VSF + CRM + MP	$11876 \pm 1600$

Table 5.15: Total average disconnection time and relative confidence interval obtained with different algorithm combinations in a scenario with parameters listed in Table 5.14.

## 5.8 Summary

In this chapter, the performance of LoRaUAV in terms of PRR and packet delay has been assessed through simulations in scenarios reproducing a simplified model of a wildfire intervention area. The considered scenarios consist of a different number of firefighter teams in the range  $[1, 5]$ , each one composed of 20 GNs, operating in a squared area of side  $2000\text{ m}$  or  $2500\text{ m}$  for a total time of, respectively,  $2000\text{ s}$  and  $3000\text{ s}$ . The modular structure of the LoRaUAV mobility allowed to perform an incremental analysis of the various sub-algorithms. At first, the basic LoRaUAV VSF algorithm has been tested and its relation with  $K_p$ , the proportional parameter of  $K_{AtA}$ , has been determined. Results show that the choice of  $K_p$ , with the purpose of maximizing the AE-PRR, depends on many factors: the total number of deployed UAVs, the number of teams of GNs and the number of GNs per team. It is also shown that the delay introduced by the buffer, in some cases, drastically affects a small percentage of packets. A subsequent comparison between the LoRaUAV VSF algorithm and the VSF algorithm proposed by Di Felice et al. [34] (DF algorithm) showed only a small improvement of the AE-PRR limited to certain  $K_p$  intervals. Increments are more marked when a large number of teams (4 or 5) is deployed. Even though the improvements are limited, the VSF algorithm proposed in this thesis can be implemented with no particular additional cost and promises to behave identically or even better than the DF VSF algorithm. The most significant improvements of the AE-PRR are obtained by enabling the CRM and MP algorithms. The CRM algorithm causes the most noticeable AE-PRR increments, with peaks of  $+50\%$ , at the expense of a considerable buffer delay affecting between  $3.5\%$  and  $11\%$  of successfully delivered packets. The MP algorithm, exclusively in combination with the CRM algorithm, further improves the AE-PRR, especially in more complex scenarios where the composition of teams changes over time. A more in-depth study of the frequency and duration of disconnections has also been performed for different combination of algorithms. It is shown that both VSF algorithms are subject to long and frequent disconnection periods caused by the early isolation of whole teams of GNs. Both the CRM and the MP algorithms effectively help in reducing such occurrence. This is proven by the total average disconnection time, that is reduced approximately by half (from  $\approx 20000\text{ s}$  to  $\approx 12000\text{ s}$ ) when the CRM and MP algorithms are used. This means that isolation of GNs either happens much later in time or that isolated GNs are recovered sooner. These claims are confirmed by the substantial AE-PRR improvements observed when CRM and MP algorithms are activated.

The algorithms implemented in LoRaUAV tend to maximize the GN coverage to the detriment of the packet delay. This might be undesirable in some applications where the collected data have strict delay requirements, possibly including also some firefighting applications. If the PRR is instead the main desired QoS metric, the usage of VSF, CRM and MP algorithms allow to achieve good AE-PRR levels, as it has been shown with the results depicted in Figure 5.16. It is also stated that the system would benefit from a  $K_p$  selection routine and a mechanism to select the number of UAVs to deploy according to the desired QoS requirements. These additions can be considered for future evolutions of this work.





## Chapter 6

# Conclusions

In this thesis, a new network system called LoRaUAV has been designed. The main objective of the system is to extend the network coverage offered by a BS to a set of mobile GNs by means of relaying gateways installed on UAVs. This objective is achieved thanks to a double-layer network that consists of a LoRaWAN segment for the communication between the GNs and the GWs and a IEEE 802.11 (WiFi) ad hoc network for all the communications between the GWs and the BS. The core of the system is the LoRaUAV mobility algorithm, specifically designed to adapt as much as possible to the ever changing position of the GNs. The mobility backbone consists of a completely distributed algorithm based on the computation of attractive/repulsive virtual spring forces. The basic mobility is then expanded by the CRM and MP algorithms, designed, respectively, to increase the GN coverage through a better UAV distribution and to restore the connection of out-of-range GNs through movement prediction. The performance evaluation of LoRaUAV is achieved through simulations performed in ns-3, using a custom-built *loravsf* module integrated with other built-in and external modules. All tests are performed on a simplified model of a wildfire disaster scenario, a situation in which LoRaUAV may prove to be particularly useful. The performance impact of each component of the system was assessed through some relevant QoS metrics, in particular the average end-to-end PRR (AE-PRR) and the average total delay (ATD). As a preliminary step, the trend of the AE-PRR and the ATD as a function of  $K_p$  and the number of UAVs and GNs has been assessed without the influence of the CRM and MP algorithms. It is proven that the selection of  $K_p$  is extremely important for achieving good performances, since there is a considerable disparity between the AE-PRR obtained with different values of  $K_p$ . Some particularly good values of  $K_p$  might in fact produce a significant increase of the AE-PRR. The ATD is also affected by  $K_p$ , the principal factor being the permanence time of packets in the GW buffer. It is shown that big values of  $K_p$  favour the compactness of the aerial mesh and diminish the buffered packets at the expense of the provided coverage. Furthermore, the need for a  $K_p$  optimization routine is underlined, since it might be too difficult to select a good value of the parameter based on the number of UAVs, the number of teams and the number of GNs per team. Next, the LoRaUAV VSF algorithm has been compared with another similar VSF algorithm developed by Di Felice et al. [34]. It has been proved that LoRaUAV achieves an average AE-PRR performance improvement which lies between 0.03 % and 4.6 % depending on the particular scenario configuration. However, if the confidence interval of the two algorithms is considered, the improvement affects only some values of  $K_p$ , without any recognizable pattern. Even if small and limited to some  $K_p$  values, the improvement comes with no significant additional cost. The impact of the activation of the CRM algorithm has been assessed relatively to a system operating only with the basic LoRaUAV VSF algorithm. The AE-PRR drastically improved: from a minimum average increment of 5.4 % to a maximum average increment of 50 %, with most of the configurations experiencing at least a double-digit performance increment. The delay is the metric that is more negatively affected by the

CRM algorithm: in the tested scenarios, between 3.5 % and 11 % of the packets are buffered and they experience an average buffer delay which can be even ten times bigger than the one experienced in a system with no CRM algorithm. This is caused by a bigger UAV disconnection chance which is inherent in the operation of the CRM algorithm. Given the significant AE-PRR improvement, the additional delay affecting just a small portion of packets can be considered as an acceptable compromise. By activating the MP algorithm, the PRR achieves another consistent improvement, but only in system configurations where the MP algorithm works in synergy with the CRM algorithm. When teams of GNs are not allowed to split during the simulation, between 66 % and 86 % of the simulated scenarios achieved an improvement of 13 % – 18 %, except when too few UAVs are used. If teams are allowed to split, the increment has the same magnitude, but now between 78 % and 91 % of the simulated scenarios showed an improvement. It is also shown that the activation of all the LoRaUAV algorithms (VSF + CRM + MP) allows to obtain an AE-PRR above 0.9 in a representative scenario with three teams of GNs and 12 UAVs. An acceptable AE-PRR, between 0.8 and 0.9, is obtained with 8 or 10 UAVs. A study of the frequency and duration of disconnection periods experienced by GNs has also been performed for different combination of algorithms. It is shown that both VSF algorithms are subject to long and frequent disconnection periods caused by the early isolation of whole teams of GNs. Both the CRM and the MP algorithms effectively help in reducing such occurrence. This is proven by the total average disconnection time, that is reduced approximately by half (from  $\approx 20000$  s to  $\approx 12000$  s) when the CRM and MP algorithms are used. In conclusion, through the performed simulations, it is given a first indication that the mechanisms implemented in LoRaUAV effectively provide a good solution to improve the coverage of mobile ground nodes by means of a flexible and adapting UAV mesh network. The obtained results might be confirmed by a future development of a small scale prototype.

## 6.1 Future Work

There are some additional points that can be explored to continue and possibly upgrade what was accomplished in the present work:

- The movement of UAV has been modelled with simple cinematic rules that do not realistically portray the real behaviour of UAVs. A more complex simulation model can be integrated to see if and how motion constraints limit the effectiveness of the system;
- The propagation model used for the LoRa AtG channel may not reflect the real propagation conditions of the signal in a forest environment. The available literature regarding this aspect is still limited, therefore the results of a more complex study can be integrated in the simulation model to obtain more realistic results;
- The developed mobility algorithm is completely distributed. This allows to create a flexible system based on simple movement rules. However, the integration with a centralized algorithm might be beneficial in some circumstances to optimize the deployment of UAVs;
- UAVs are subject to strict energy constraints. The impact of such constraints is of great practical interest. In fact, the developed algorithm may need to be modified to limit as much as possible unnecessary movements and transmissions. A future iteration of the system might also consider the usage of fixed-wings drones to increase the maximum air-time. This also comes with additional mobility constraints imposed by the architecture of fixed-wings drones;
- The algorithm can be expanded to consider also the case in which more than one BS is available. Moreover, some BSs may have the ability to move to adapt to the aerial mesh position;

- In the developed system, GNs are only able to send uplink messages to the BS. In real scenarios, some sort of downlink message delivery might be desirable, even if limited. A future extension to support the delivery of LoRaWAN downlink messages through the UAV mesh would be of great practical interest;
- Setting the right value of  $K_p$  is particularly complex, since it depends on many conditions. A  $K_p$  optimization routine is therefore necessary and should be considered in future extensions of the present work;
- Prediction techniques, such as the ones implemented in the MP algorithm, seem to be effective. More sophisticated techniques are therefore worth being investigated.



# Bibliography

- [1] N. Shahid and S. Aneja, "Internet of things: Vision, application areas and research challenges," in *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pp. 583–587, Feb 2017.
- [2] D. G.-S. P. H. P. Wallemacq and R. Below, "Annual disaster statistical review 2016," tech. rep., EM-DAT, 2017.
- [3] S. Doerr and C. Santín, "Global trends in wildfire and its impacts: Perceptions versus realities in a changing world," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 371, Jun 2016.
- [4] "ERC recommendation 70-03." [https://www.ecodocdb.dk/download/25c41779-cd6e/attachments/2016\\_REC7003E.pdf](https://www.ecodocdb.dk/download/25c41779-cd6e/attachments/2016_REC7003E.pdf). Accessed: 1-6-2018.
- [5] "SX1276/77/78/79 datasheet." [https://www.semtech.com/uploads/documents/DS\\_SX1276-7-8-9\\_W\\_APP\\_V5.pdf](https://www.semtech.com/uploads/documents/DS_SX1276-7-8-9_W_APP_V5.pdf). Accessed: 3-6-2018.
- [6] "LoRaWAN 1.1 specification." [https://loro-alliance.org/sites/default/files/2018-04/lorawantm\\_specification\\_v1.1.pdf](https://loro-alliance.org/sites/default/files/2018-04/lorawantm_specification_v1.1.pdf). Accessed: 3-6-2018.
- [7] "LoRa modulation basics." <https://www.semtech.com/uploads/documents/an1200.22.pdf>. Accessed: 3-6-2018.
- [8] M. Knight, "Reversing LoRa." [https://github.com/matt-knight/research/blob/master/2016\\_05\\_20\\_jailbreak/Reversing-Lora-Knight.pdf](https://github.com/matt-knight/research/blob/master/2016_05_20_jailbreak/Reversing-Lora-Knight.pdf). Accessed: 3-6-2018.
- [9] M. Knight and B. Seeber, "Decoding lora: Realizing a modern lpwan with sdr," *Proceedings of the GNU Radio Conference*, vol. 1, no. 1, 2016.
- [10] Tektronix, "Wi-Fi: Overview of the 802.11 physical layer and transmitter measurements." [https://public.cnrood.com/public/docs/WiFi\\_Physical\\_Layer\\_and\\_Transm\\_Meas.pdf](https://public.cnrood.com/public/docs/WiFi_Physical_Layer_and_Transm_Meas.pdf). Accessed: 18-6-2018.
- [11] G. He, "Destination-Sequenced Distance Vector (DSDV) Protocol," tech. rep., Helsinki University of Technology, Jun 2002.
- [12] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks," in *Proceedings. IEEE International Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century.*, pp. 62–68, Dec 2001.
- [13] D. B. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," *Mobile Computing*, vol. 353, May 1999.

- [14] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings WM-CSA'99. Second IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90–100, Feb 1999.
- [15] B. Karp and H. T. Kung, "Gpsr: Greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, Mobi-Com '00*, pp. 243–254, 2000.
- [16] P. G. Fahlstrom and T. J. Gleason, *Introduction to UAV Systems, Fourth Edition*. Wiley, 2012.
- [17] J. Petajajarvi, K. Mikhaylov, A. Roivainen, T. Hanninen, and M. Pettissalo, "On the coverage of lpwans: range evaluation and channel attenuation model for lora technology," in *2015 14th International Conference on ITS Telecommunications (ITST)*, pp. 55–59, Dec 2015.
- [18] P. Jörke, S. Böcker, F. Liedmann, and C. Wietfeld, "Urban channel models for smart city iot-networks based on empirical measurements of lora-links at 433 and 868 mhz," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–6, Oct 2017.
- [19] S. Y. Wang, Y. R. Chen, T. Y. Chen, C. H. Chang, Y. H. Cheng, C. C. Hsu, and Y. B. Lin, "Performance of lora-based iot applications on campus," in *2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*, pp. 1–6, Sept 2017.
- [20] O. Iova, A. L. Murphy, G. P. Picco, L. Ghio, D. Molteni, F. Ossi, and F. Cagnacci, "Lora from the city to the mountains: Exploration of hardware and environmental factors," in *Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks, EWSN &#8217;17, (USA)*, pp. 317–322, Junction Publishing, 2017.
- [21] L. Angrisani, P. Arpaia, F. Bonavolontà, M. Conti, and A. Liccardo, "Lora protocol performance assessment in critical noise conditions," in *2017 IEEE 3rd International Forum on Research and Technologies for Society and Industry (RTSI)*, pp. 1–5, Sept 2017.
- [22] K. Mikhaylov, . J. Petaejaejaervi, and T. Haenninen, "Analysis of capacity and scalability of the lora low power wide area network technology," in *European Wireless 2016; 22th European Wireless Conference*, pp. 1–6, May 2016.
- [23] A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, "A study of lora: Long range and low power networks for the internet of things," *Sensors*, vol. 16, no. 9, 2016.
- [24] M. C. Bor, U. Roedig, T. Voigt, and J. M. Alonso, "Do lora low-power wide-area networks scale?," in *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, MSWiM '16, (New York, NY, USA)*, pp. 59–67, ACM, 2016.
- [25] "ns-3." <https://www.nsnam.org/>. Accessed: 21-9-2018.
- [26] E. Weingartner, H. vom Lehn, and K. Wehrle, "A performance comparison of recent network simulators," in *2009 IEEE International Conference on Communications*, pp. 1–5, June 2009.
- [27] D. Magrin, M. Centenaro, and L. Vangelista, "Performance evaluation of lora networks in a smart city scenario," in *2017 IEEE International Conference on Communications (ICC)*, pp. 1–7, May 2017.
- [28] F. V. den Abeele, J. Haxhibeqiri, I. Moerman, and J. Hoebeke, "Scalability analysis of large-scale lorawan networks in ns-3," *IEEE Internet of Things Journal*, vol. 4, pp. 2186–2198, Dec 2017.

- [29] "Simpy." <https://simpy.readthedocs.io/en/latest/>. Accessed: 21-9-2018.
- [30] "Omnet++." <https://www.omnetpp.org/>. Accessed: 21-9-2018.
- [31] K. Chandrashekar, M. R. Dekhordi, and J. S. Baras, "Providing full connectivity in large ad-hoc networks by dynamic placement of aerial platforms," in *IEEE MILCOM 2004. Military Communications Conference, 2004.*, vol. 3, pp. 1429–1436 Vol. 3, Oct 2004.
- [32] C. Caillouet and T. Razafindralambo, "Efficient deployment of connected unmanned aerial vehicles for optimal target coverage," in *2017 Global Information Infrastructure and Networking Symposium (GIIS)*, pp. 1–8, Oct 2017.
- [33] N. Goddemeier, K. Daniel, and C. Wietfeld, "Role-based connectivity management with realistic air-to-ground channels for cooperative uavs," *IEEE Journal on Selected Areas in Communications*, vol. 30, pp. 951–963, June 2012.
- [34] M. D. Felice, A. Trotta, L. Bedogni, K. R. Chowdhury, and L. Bononi, "Self-organizing aerial mesh networks for emergency communication," in *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*, pp. 1631–1636, Sept 2014.
- [35] "lorawan for ns-3." <https://github.com/signetlabdei/lorawan>. Accessed: 12-6-2018.
- [36] Y. S. Meng, Y. H. Lee, and B. Chong Ng, "Study of propagation loss prediction in forest environment," *Progress in Electromagnetics Research B*, vol. 17, pp. 117–133, Jan 2009.
- [37] A. A. Khuwaja, Y. Chen, N. Zhao, M. Alouini, and P. Dobbins, "A survey of channel modeling for uav communications," *IEEE Communications Surveys Tutorials*, Jan 2018.
- [38] National Wildfire Coordinating Group, *Wildland Fire suppression tactics reference guide*, 1996.
- [39] C. Tymstra and M. Flannigan, "Living with wildland fire: What we learned from the 2016 horse river wildfire." [https://www.frames.gov/files/9615/1190/3222/2017\\_November\\_21\\_Alaska\\_Fire\\_Science\\_Consortium\\_webinar\\_Tymstra\\_Flannigan.pdf](https://www.frames.gov/files/9615/1190/3222/2017_November_21_Alaska_Fire_Science_Consortium_webinar_Tymstra_Flannigan.pdf). Accessed: 11-7-2018.
- [40] R. W. Bohannon, "Comfortable and maximum walking speed of adults aged 20—79 years: reference values and determinants," *Age and Ageing*, vol. 26, no. 1, pp. 15–19, 1997.
- [41] "Hothot crews." <https://www.fs.fed.us/science-technology/fire/people/hotshots>. Accessed: 27-9-2018.





## Appendix A

### Additional Plots

#### A.1 Trend of some QoS metrics with the basic LoRaUAV VSF algorithm for 3 teams of GNs

