Study and conception of adaptive algorithms and mechan in the Internet of Mobile Things (IoMT)

Philippe Fabian

Université Gustave Eiffel

UNIVERSITÉ PARIS-EST

UNIVERSITÉ PARIS-EST

ÉCOLE DOCTORALE MSTIC

Mathématiques et Sciences et Technologies de l'Information et de la Communication

Thèse de doctorat

Discipline : Informatique

Présentée par

M. Philippe FABIAN

Study and conception of adaptive algorithms and mechanisms for data transportation in the Internet of Mobile Things (IoMT)

Thèse dirigée par Abderrezak RACHEDI

Soutenue le 14 décembre 2020

Jury :

Lyes KHOUKHI	Rapporteur	ENSICAEN
Samuel PIERRE	Rapporteur	Polytechnique Montréal
Marcelo Dias DE AMORIM	Président du jury	Université Pierre et Marie Curie
Samia BOUZEFRANE	Examinatrice	Conservatoire National des Arts et Métiers
Abbas BRADAI	Examinateur	Université de Poitiers
Abderrezak RACHEDI	Directeur de thèse	Université Gustave Eiffel
Cédric GUÉGUEN	Co-Encadrant	IRISA Rennes

Remerciements

Pour commencer, je tiens à remercier les membres du jury pour leur rôle essentiel quant à l'évaluation de cette thèse. Merci à Lyes Khoukhi et Samuel Pierre d'avoir rapporté cette thèse.

Un grand merci à Abderrezak, mon directeur de thèse, pour son encadrement exceptionnel. Tu m'as beaucoup aidé, tant du point de vue technique mais aussi - et surtout - du point de vue moral. Tu as toujours été constructif et ce, peu importe la situation. Je crois bien que cet aspect de ta personnalité a déteint sur moi, et j'en suis très content !

Cédric, mon co-encadrant, ancien maître de stage et surtout très bon ami ;) Merci de ton implication dans mon parcours et d'avoir cru en moi durant une période qui m'était très difficile. Merci aussi de toujours être "sans pitié" !

Merci à Patrice du CRI, à Corinne et aux "filles" du LIGM pour leur support durant toute ma thèse.

Merci beaucoup à Sylvie Cach de l'ÉD pour ses réponses très rapides et son aide précieuse pour toutes les questions relatives au doctorat, surtout sur la fin !

À Mounty, Rémi, Titi et les autres pour ces bonnes petites soirées au centre-ville de Paris :P

Merci à mes parents ! À mon père pour m'avoir appris à remettre les choses en question et pour m'avoir transmis ce goût pour l'informatique. À ma mère, toujours dédiée et qui m'a poussé à faire des études. Merci à François pour son soutien et sa bonne humeur constante !

Merci à Hampelie ! C'est pas facile de trouver des fans de films d'horreur et de Dove !

Merci à Xavier, Dr. Fabian I^{er} ! Pas seulement pour les conseils judicieux au sujet de la thèse et du LaTeX pour le manuscrit mais pour tout le reste aussi. Tu m'as rendu la pareille avec les séances "décompressions nécessaires" ;)

Merci également à Amandible pour ces moments sur Paris qui me changeaient le quotidien (découverte d'arrondissement et kimchi plus ou moins réussis (ratés ?).

Merci à Aymeric ! Bien que rares, les séjours à Rennes m'ont toujours bien boostés. À chaque fois, je savais que ce serait serein et posé. Prions pour n'avoir que des burgers gratuits !

Дякую Даша ! It seems PhD is not just about pure research but also meeting new awesome (and Precious) people. I was a coffee purist. Зараз, the question is кофе или чай ? Пум пурум пум пум...

At last, special thanks to the open source community : Pixabay for hosting such a nice collection of copyright free images. All the images in this manuscript that I did not create are from Pixabay. SUMO [1], which was used to model mobility in simulations. *Danke schön Uwe Drechsel für* VYM, the mind mapping tool I used extensively to organize ideas and tasks.

List of publications

International journals

- P. Fabian, A. Rachedi, and C. Gueguen, "Programmable objective function for data transportation in the Internet of Vehicles," Transactions on emerging telecommunications technologies, Jan. 2020 (DOI: 10.1002/ett.3882) [published];
- P. Fabian, A. Rachedi, and C. Gueguen, "Selection of mobile relays based on classification of mobility and network metrics in a highly dynamic context," Transactions on emerging telecommunications technologies, Mar. 2021 (DOI: 10.1002/ett.4246) [published];
- T. Abar, A. Rachedi, A. ben Letaifa, P. Fabian, and S. El Asmi, "Fellowme cache: Fog computing approach to enhance (qoe) in internet of vehicles," Future Generation Computer Systems, 2020 (DOI: 10.1016/j.future.2020.06.026) [published].

International conferences

- P. Fabian, A. Rachedi, C. Gueguen, and S. Lohier, "Fuzzy-based Objective Function for Routing Protocol in the Internet of Things," in IEEE Global Communications Conference (Globecom), Abu-Dhabi, United Arab Emirates, Dec. 2018 (DOI: 10.1109/GLOCOM.2018.8647969) [published];
- P. Fabian and A. Rachedi, "Dynamic selection of relays based on classification of mobility profile in a highly mobile context," in 2020 IEEE International Conferences on Communications (ICC). IEEE, 2020, pp. 1–6 (DOI: 10.1109/ICC40277.2020.9149423) [published];
- P. Fabian, A. Rachedi, and C. Gueguen, "Placement of UAV relays to improve connectivity using position prediction in the Internet of Vehicles," [to be submitted].

Résumé français

La quantité de données produite par l'internet des objets augmente de plus en plus rapidement. Qu'il s'agisse du déploiement à grande échelle de capteurs dans une ville ou du nombre croissant d'utilisateurs de téléphones mobiles (*smartphones*), le nombre d'appareils connectés n'a jamais été aussi important. L'avènement des véhicules autonomes et des nouvelles technologies telles les villes jumelles numériques requièrent une qualité de service élevée. En effet, nous avons d'une part les véhicules autonomes et plus généralement les appareils et utilisateurs mobiles qui affectent la topologie du réseau par leurs déplacements dans le temps et, d'autre part, nous avons la navigation en temps réel dans une ville avec de la réalité augmentée ou dans un jumeau numérique de ville en réalité virtuelle qui nécessitent un grand débit, un faible délai ainsi qu'une consommation d'énergie réduite. Une panne d'équipement, un événement social causant une soudaine augmentation du trafic réseau ou une catastrophe naturelle qui endommage l'infrastructure réseau sont des exemples de situations compliquées à gérer.

L'objectif de cette thèse est de développer de nouvelles approches à la fois flexibles et globales en vue d'améliorer et de garantir la qualité de service au sein de réseaux sans infrastructure et très dynamiques. Elles s'appuient sur divers aspects au niveau du réseau, du système et des applications. D'un point de vue théorique, ceci est accompli par la résolution d'un problème d'optimisation multi-contraintes où tous les aspects sont considérés à la fois. L'algorithme est distribué à cause du manque d'infrastructure et la nécessité de répondre aux besoins locaux. Un processus d'aide à la décision assisté par apprentissage artificiel est mis en place pour sélectionner l'élément optimal d'un ensemble de candidats potentiels.

Nous commençons par établir le contexte des communications sans-fil dans l'internet des objets et présentons les différents types de réseaux pertinents pour ce travail. Ensuite, nous expliquons en détail le concept des métriques reliées au réseau, au système et à l'application avant de présenter les défis quant à assurer la qualité de service dans les réseaux mobiles. Nous poursuivons par l'analyse et la comparaison des architectures sans-fil existantes comme les réseaux complètement centralisés basés sur une infrastructure dédiée et les réseaux ad hoc, plus adaptés à l'internet des objets mobiles et à l'internet des véhicules. Après, nous proposons un moyen pour l'amélioration du routage dans les réseaux mobiles en considérant plusieurs types de métriques à la fois, incluant la vitesse de déplacement des appareils. Une architecture originale est également proposée pour la mise en place de la solution. Ensuite, nous mettons en oeuvre une méthode d'apprentissage artificiel pour classifier le type de mobilité des appareils en vue d'améliorer la sélection de noeuds relais mobiles dans l'internet des véhicules. Finalement, nous proposons un algorithme d'apprentissage artificiel pour prédire la position future des appareils mobiles afin de trouver la meilleure position pour placer une drone relais à l'aide d'un système de score qui tient compte des positions présentes et futures des appareils.

Des simulations sont effectuées pour valider les contributions et les résultats démontrent que les solutions proposées augmentent les performances au niveau réseau (taux de délivrance des paquets, délai bout-en-bout, ...) et au niveau système (consommation énergétique, taux d'occupation des files d'attentes, ...). Par conséquent, la fiabilité au niveau applicatif s'en voit améliorée, augmentant la qualité de service et la qualité d'expérience du point de vue utilisateur.

Mots clefs: *Internet des objets mobiles, protocole de routage, mobilité, Internet des véhicules, sélection de relai, apprentissage artificiel, fonction multi-objectifs*

Abstract

The amount of data generated in the Internet of Things (IoT) is increasing at a rapid pace. From the massive deployment of sensors in smart cities to the growing number of smartphone users, the number of connected devices is greater than ever. The advent of autonomous vehicles and emerging technologies such as digital twin cities impose strong Quality of Service (QoS) requirements. Indeed, on the one hand autonomous vehicles and, to a broader extent, mobile users and devices cause the network topology to constantly change through time. On the other hand, navigating in real-time through a city with augmented reality or through its digital twin in virtual reality require high throughput, low delay and reduced power consumption. A failing equipment, a sudden increase in network traffic caused by a social event or a natural disaster damaging the infrastructure are situations that are complicated to deal with.

The objective of this thesis is to work on global and flexible approaches to improve and ensure QoS in very dynamic infrastructure-less networks by considering several different aspects related to network, system and application. From a theoretical point of view, this is achieved by solving a multi-constraint optimization problem where all aspects are considered at once. The algorithm is distributed due to the lack of infrastructure and the need to answer local needs. A Machine Learning (ML) assisted decision-making process is implemented to select the best element in a set of possible candidates.

We first establish the context of wireless communications in the IoT and discuss the different types of networks relevant to this work. We then explain in detail the concept of network-, system- and application-oriented metrics before presenting the challenges of ensuring QoS in mobile networks. We continue by analyzing and comparing the existing wireless architectures such as fully centralized networks with a dedicated infrastructure and distributed ad hoc networks more adapted to the Internet of Mobile Things (IoMT) and the Internet of Vehicles (IoV). Next, we propose a scheme to improve routing in mobile networks through the consideration of several types of metric at once including the velocity of devices. A novel architecture is proposed as well to deploy the solution. Then, we provide a method based on a ML algorithm to classify the mobility-type of devices to improve the selection of mobile relay nodes in the IoV. Finally, we propose a ML-based algorithm to predict the future position of devices to find the best location to place a relay drone using a score system taking into account current and future positions of devices.

Simulations are run to validate the contributions and results show the proposed solutions increase performance at the network level (packet delivery ratio, end-to-end delay, ...) and system level (energy consumption, buffer occupancy, ...). This in turn yields better reliability at the application level, increasing QoS and Quality of Experience (QoE) from the user's point of view.

Keywords: *IoMT, routing protocol, mobility aware, IoV, relay selection, machine learning, multi-objective function*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Résumé français

Le premier réseau 2G a vu le jour en Finlande au début des années 1990. Depuis, les réseaux ont beaucoup évolué jusqu'au présent jour où plusieurs technologies sans-fil sont disponibles. Il en existe pour les communications en longue portée (comme LoRa, SigFox ou la 3/4/5G) ou en portée plus courte (comme le WiFi ou le Bluetooth). Au niveau réseau et transport, les protocoles les plus fréquents sont le IPv4/IPv6 (couche réseau) et le UDP ou TCP pour la couche transport. Le déploiement de la 5G est en court depuis quelques mois à l'heure de l'écriture de ce manuscrit et la recherche sur la 6G et la 7G est en déjà lancée.

Le nombre d'objets connectés (capteurs, véhicules connectés, service de vélo ou trottinettes en libre-service, *smartphones*, etc.) croît de façon exponentielle depuis plusieurs années et le trafic réseau généré par ces appareils est de plus en plus important. Pour assurer la qualité de service, les contraintes sont également de plus en plus strictes (délais toujours plus réduits ou débits plus élevés, par exemple). La mobilité des ces appareils pose un problème dans la mesure où elle cause une modification continue de la topologie ce qui, par conséquent, rend cette dernière instable.

Afin d'assurer une bonne qualité de service, plusieurs métriques peuvent être examinées et utilisées. Nous avons les métriques orientées réseau, système ou application. Les métriques réseau concernent les liens entre deux noeuds, tel le débit ou la qualité de la transmission. Les métriques orientées système sont spécifiques à l'appareil considéré (niveau de batterie restant, puissance de calcul, taux d'occupation des files d'attente, etc.). Finalement, les métriques orientées application portent sur le besoin actuel de l'appareil. Par exemple, un utilisateur exécutant une application de type VoIP sur un *smartphone* n'a pas le même besoin qu'un capteur envoyant une température par heure.

Les limitations de l'état de l'art comme la rigidité de l'infrastructure physique du réseau, la difficulté de s'adapter à un environnement dynamique ou encore la problématique d'anticiper les déplacements des appareils sont les éléments principaux qui motivent cette thèse. L'objectif de celle-ci est de développer une solution plus globale au problème du routage (c'est-à-dire trouver le meilleur chemin pour l'acheminement des paquets réseau) dans les réseaux sans-fil mobiles en tenant de la mobilité des appareils, de leur densité, du type de donnée, de leurs besoins et du type de ces appareils.

Nous proposons plusieurs solutions pour répondre à cette problématique : la création d'une fonction objectif qui tient compte de plusieurs contraintes (métriques orientées réseau et système, la vitesse des noeuds et une infrastructure de type SDN; chapitre 3), la classification du type de mobilité des utilisateurs pour assurer la continuité de service (en faisant appel à un algorithme d'apprentissage artificiel et à un mécanisme de sélection de relais; chapitre 4) et, finalement, la prédiction de la position future des noeuds (par apprentissage artificiel et à l'aide d'un système de points (score) permettant de définir des zones dites "critiques"; chapitre 5).

1.1 Context

1.1.1 General context

In the early 80s in Western Europe the telecommunications market was fragmented [6]. In 1982, the *Groupe Spécial Mobile* (GSM) was formed to study a new solution for a standardized digital network for telecommunications across Western Europe. The first 2G network to be deployed was the GSM in Finland, 1991 [6] [7]. This cellular standard was developed with flexibility in mind. It had to be able to work on handheld devices and, apart from other features that were to be implemented, the standard was required to eventually support voice encryption and new services different from speech. The end of the 90s was marked by the first specification of the 3G standard by 3rd Generation Partnership Project (3GPP)[6], the popularization of WiFi (IEEE 802.11b) [8] and the beginning of Bluetooth [9]. During the 2000s, new protocols were designed to offer more possibilities and respond to the new needs of users. Some examples are protocols based on IEEE 802.15.4 (for low data rate networks) [10] like the Routing Protocol for Low-power and lossy networks (RPL) [11] and ZigBee [12] or longer range protocols such as LoRa and SigFox (for low power and long range operating in unlicensed spectrum) [13] [14].

Today's computers (desktop and laptop computers) connect to Internet using mostly the IEEE 802.3 (Ethernet) [15] and IEEE 802.11 (WiFi) [16] standards for the lowest layers of a typical TCP/IP architecture (physical and link). For the upper layers (network and transport), the Internet Protocol (IP) version 4 [17] and 6 [18] (IPv4/IPv6) and the Transmission Control Protocol (TCP) [19] or User Datagram Protocol (UDP) [20] are used. On the other hand, mobile devices such as smartphones and on-board computers in motorized vehicles typically use cellular networks (3/4/5G) for the lower layers to connect to Internet. Because the range of WiFi is usually no more than about 100-150 meters and because the range of cellular networks can go up to a few kilometers, the latter is favored when mobility is present. With 5G networks already being deployed as of 2020, research on 6G has begun [21]. It is expected to offer throughput between several hundreds of Gbps to 1 Tbps and ultra low latency (at the microsecond scale) [22] [23]. This would unlock applications such as real-time Virtual Reality (VR) to travel through the digital twin of a city.

1.1.2 The advent of "Things"

Recent years have been marked by a shift from the use of computers to smaller devices like smartphones or tablets. Indeed, improvements in processing power, memory/storage capacity, battery life and the lowering of production cost of wireless devices, coupled with the advent of 4G cellular networks [24] [25] have contributed to the massive deployment and increase in their number [26]. This caused a major

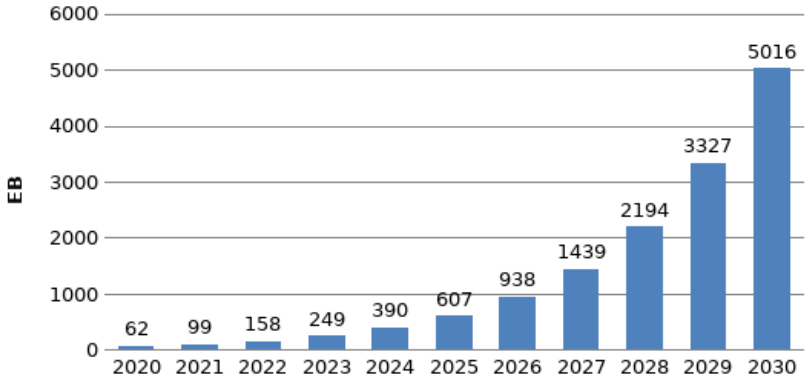


Figure 1.1: Estimation of mobile traffic worldwide by ITU [2].

expansion of wireless networks [27] [28] [29]. A report from the International Telecommunication Union (ITU) [2] predicts the global mobile traffic will increase by a factor of about 80 from 2020 to 2030 (see fig. 1.1).

Sensors are also a major part of wireless networks. The range of usage is vast and sensors are deployed in many places such as in industrial complexes [30], farming areas and cities, to name a few.

In industrial areas it is important to monitor metrics like temperature, presence in the air of specific gases or pressure inside tanks for safety reasons. For instance, if a fire breaks out it is crucial to detect it as soon as possible. Sensors can also be used to monitor physical or chemical processes in the industry. A precise temperature and pressure might be required for a chemical reaction to occur. Connecting sensors and actuators wirelessly has been possible since the third industrial revolution, that is, the inclusion of computer science in industry (or *digitalization*) [31]. In 2020, we are in the midst of Industry 4.0 which refers to the extensive use of IoT to connect the manufacturing process to Internet [32]. Industry 5.0 is expected to be the cooperation between humans and robots to increase safety and productivity [33]. It comes at the cost of a higher complexity (more sensors and more data exchange) and, to ensure safety, very low delay and high reliability are required.

In farming, the monitoring of weather (quantity of rainfall, amount of wind or duration of sun lighting) can help farmers in the growing of crops by selecting which ones to place at specific locations. Measuring rainfall can help in reducing water consumption by deciding which plants to water and on the amount of water to be used. Sensors can be placed on cattle to monitor their health [34].

In smart cities [35] sensors are used for many purposes such as monitoring the quality of air or detecting traffic jams, damage to streets/sidewalks or full trash bins. For example, knowing the quality of air can help the city in deciding when to discourage the usage of cars to help reduce air pollution and improve the health of citizens. The detection of traffic jams can trigger a message to be sent to the on-board computer of concerned cars so the path to destination can be modified accordingly. Detecting full trash bins or damage to infrastructure and structures allows to optimize the use of human resources by sending a crew only where there is a need. Such information can be used in real-time when rendering the digital twin of a city to ease the monitoring of the real one or to facilitate the management of abnormal situations [36].

In those examples, sensors are usually static. However, it is possible to place them on a mobile object like a car. Other examples of mobility include connected vehicles and smartphone users who also travel at different velocities.

1.1.3 About mobility in the IoT

Motorized vehicles are becoming more and more connected. Global Navigation Satellite System (GNSS) is used to locate a vehicle and Internet connectivity allows to download up-to-date maps to guide the driver to the destination. Internet can be used for entertainment when passengers of a car, bus or train watch a multimedia stream while traveling. Connected vehicles also unlock possibilities like locating cars in real-time for traffic management [37] without the need of sensors by roads, automatic accident reporting [38] [39] and autonomous vehicles to name a few.

Connected vehicles, smartphones, sensors and other such devices are considered as connected mobile things and can connect together to form a network (see next subsection): they form a subclass of wireless networks where devices are in motion. This adds a layer of complexity in finding the best path to send data. Contrary to static wireless networks, the mobile case implies a more unstable topology. As devices travel from one location to the next, they might find themselves out of reach of a Road Side Unit (RSU) and move in range of other devices and/or RSUs. The topology is thus constantly evolving and the quality of links varies even faster.

The changing in the quality of links is due to path loss when moving out of range of another device or antenna, shadowing caused by new obstacles blocking waves (e.g. entering a tunnel, turning on a street corner behind a building, walking behind a bus and so on) and by multipath fading, where users and other people or vehicles only need to move slightly for the impact to be significant. Hence, the varying quality of links is caused by changes in the environment of users.

Also, many of these devices are on battery and the lifetime of a network is as long as the lifetime of the first device to fail. This makes the consideration of energy saving an important aspect in such networks to ensure their stability. Indeed, if devices consume too much energy, a low-battery device might unexpectedly disconnect from others and break the connectivity of the network.

This dynamic topology also implies that the needs of users are not necessarily location-related. In the case of static networks it is possible to make some assumptions on the usage of certain devices or certain locations: a supercomputer has a low probability of generating multimedia or Voice over IP (VoIP) traffic. Delay is not a primary concern if we assume it runs simulations locally, but a large bandwidth to download results is very important. Laptops connected to a WiFi Access Point (AP) in a library are probably used to make queries on Internet where delay is important but throughput is less critical. A home device - desktop, laptop or ARM-based device (tablet, smartphone, Raspberry PI) - has a higher probability of generating all kinds of traffic.

It is different in mobile networks as all users and devices can potentially have different needs and move in any direction. These needs must be met in order to ensure Quality of Service (QoS) and Quality of Experience (QoE). In practice, one must guarantee that certain metrics (system and network) have a minimum or maximum value for QoS to be satisfactory. For instance in VoIP applications,

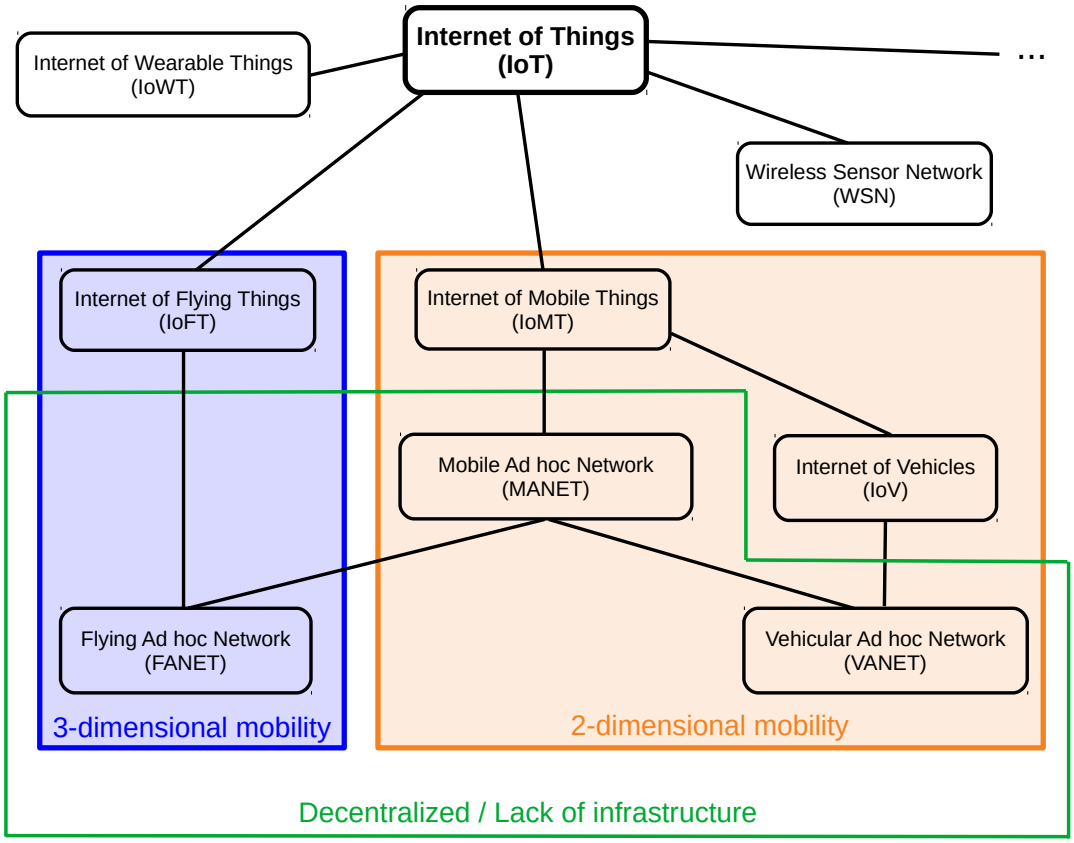


Figure 1.2: Selected types of networks related to the IoT.

delay must be no more than 50 ms. When watching a live video, throughput must be high enough to download the frames at least as fast as they are rendered on the screen but a higher throughput is not necessary.

Data transportation in mobile networks is thus a complex problem to solve and research is still very active in this topic. In this thesis, we focus on connected mobile things. However, before discussing the challenges and the focus of this work, we begin by defining the different network types related to the IoT.

1.1.4 Classification of network types

In this subsection, we shall define different types of networks. The aim is not to be exhaustive on all types of wireless networks but rather to show what is of interest for this work. Figure 1.2 illustrates the different types of network that are discussed in this subsection and how we classify them.

There are several definitions of the Internet of Things (IoT) [40] [41] [42]. We shall define the IoT as a network composed of any equipment capable of generating or processing some data and that is able to connect wirelessly to other such equipment to form a network. An equipment can thus be a simple static sensor that measures and sends the ambient temperature once an hour or it can be a smartphone where the user is talking *via* a VoIP application while browsing a website.

As depicted on figure 1.2, the Internet of Wearable Things (IoWT), the Internet of Flying Things (IoFT), the Internet of Mobile Things (IoMT) and Wireless Sensor

Network (WSN) are all considered as a subset of the IoT. Then, Mobile Ad hoc Network (MANET), The Internet of Vehicles (IoV), Vehicular Ad hoc Network (VANET) and Flying Ad hoc Network (FANET) are subsets of IoMT.

The IoWT regroups things worn by people. Examples include smart watches, body sensors or smart clothing [43]. They are useful in monitoring the body for sportsman/sportswoman or to detect health issues in people suffering from a condition. It is worth noting that the sensors can be connected to a device the user is carrying (such as as smartphone) or to an Access Point (AP) or gateway. In the former, sensors have a null relative velocity and in the latter the mobility has to be taken into account (sensors can end up being out of range of the AP).

The IoFT refers to things that are flying or hovering [44]. They can be a fixed-wing drone flying from one location to another one while being connected on some network or a static object (drone with propellers or a balloon hovering at a specific location). Here, we must make the distinction between *fly* and *hover*: flying implies movement, hovering does not. Indeed, a hovering balloon must first reach the intended location by flying, but the connection to a network (or acting as an AP) happens after the balloon has stopped.

A WSN is a subset of IoT (see fig. 1.2) composed of sensors that lack the computational power of more complex devices like smartphones or laptop computers. Those devices are usually in great number with limited power and it can be hard to recharge their battery (sensors in great number, placed inside a wall or using a non-rechargeable battery): power efficiency is very important. WSN is not a subset of IoMT because sensors can be static or mobile though they are usually static [34] [45].

The IoMT is a subset of the IoT where nodes have a non-null velocity [46]. On figure 1.2, IoMT is not a super-set of IoWT, IoFT or WSN because there is not necessarily motion involved in these. More specifically, in the IoMT we consider that at any moment a device can start moving though an AP can be either static or in motion.

The IoV comprises devices onboard motorized vehicles (cars, trucks, buses, etc.) driving on a system of roads [47]. It is possible to include bicycles and scooters as well because they can be present on roads and circulate at similar velocities in urban environments. IoV may depend on a dedicated infrastructure using for instance RSUs or may be without infrastructure (VANET).

MANET includes VANET and FANET (see below). Ad hoc networks are characterized by a lack of infrastructure [48]. Nodes connect to neighbors which creates a topology allowing them to communicate amongst themselves [49].

VANET is a subset of IoV but also of MANET [49]. VANET and MANET are both ad hoc networks which are by definition networks without a dedicated infrastructure. The difference lies in the fact VANET concerns vehicles, excluding for instance pedestrians walking with smartphones. The use of drones to only assist in establishing the topology [50] or improving performance is considered as VANET (that is, drones do not generate data).

FANET is a subset of MANET and IoFT. The difference between IoFT and FANET is that a network of flying things (IoFT) is not necessarily ad hoc. FANET is

Network type	Mobility	Predictability
IoT	any	any
IoWT	any	any
IoFT	any	any
IoMT	low/high	any
WSN	none/low	simple
IoV	average/high	average
VANET	average/high	average
FANET	high	complex
MANET	low/high	average/complex

Table 1.1: A mobility-oriented taxonomy of the different networks.

a relatively new concept [51] [52] where devices are flying drones/Unmanned Aerial Vehicles (UAVs). Note that as depicted on figure 1.2, FANETs are not a subset of VANETs. The problematic is different as in a VANET devices are constrained to a 2-dimensional (2D) environment, they evolve on roads and they are blocked by obstacles such as buildings, terrain and vegetation whereas no such restrictions are present in a FANET.

Table 1.1 depicts the classification of the different network types (as seen on fig. 1.2) according to mobility and predictability. Mobility refers to the devices' velocity and predictability concerns how complex it is to predict their trajectory. Given IoT, IoWT, IoFT are broad categories they cover the whole spectrum from no to high mobility, and from simple to complex predictability. IoMT is similar though it is assumed some mobility at least is present.

Sensors part of a WSN are usually placed at specific locations to monitor the environment [45] or they can be placed on cattle, for instance, to monitor their health [34]. In the latter, the velocity is low and the area where they move is limited. Hence, predicting the trajectory of devices in a WSN is simple. Note that sensors on-board cars are connected using wired connectivity and it is the car that will aggregate data emitted by them and will eventually communicate with other cars. Sensors on vehicles are thus considered as part of VANET.

In the IoV and VANETs, users can move at an average speed if we consider an urban setting including motorized vehicles, bicycles and scooters. They drive on roads which restrains the space of possible trajectories [53]. On the other hand, in a big city there can be many intersections so the predictability is of average complexity for the urban case. We must also consider the case of countryside roads and highways with low road traffic and fewer soft-mobility users such as bicycles and scooters. Here, the velocity of users is higher compared to the urban setting but intersections are less common, so we consider the predictability to be of average complexity.

In a FANET, UAVs can potentially move very fast (in the case of fixed-wing drones) and they can increase and lower their altitude. The adding of the Z-axis and the lack of obstacles (at least when the drone flies high enough to avoid buildings and vegetation) makes it possible for a UAV to move along a complex 3-dimensional trajectory that is hard to predict.

Given MANET includes FANET and VANET, the possible velocity of users

ranges from low to high. In a MANET, users can also be pedestrians walking through buildings, forests, unmarked paths and so on which increases the complexity of prediction.

Now that we have classified the different types of network, we will discuss about metrics to see how they affect routing once devices on a network start to communicate between one another.

1.1.5 Metric and performance index

Devices required to join a network can choose to which neighbor(s) or AP they will connect to through the use of metrics. In this manuscript, the word "metric" refers to measurements of instantaneous or averaged (using a sliding window mechanism, for instance) values relevant in deciding which neighbor or AP is the best. These values can be network-oriented, system-oriented or application-oriented:

1) Network-oriented metrics

Network-oriented metrics are deduced from wireless transmissions. They are related to the link-state between nodes. Sending and receiving packets allow to estimate delay, jitter, Expected Transmission Count (ETX) [54], hop count, throughput, connectivity, etc. Environmental factors like the position of devices (high density causing interference or inside a building causing strong shadowing) or their mobility (high fading) affect those metrics.

2) System-oriented metrics

These metrics only depend on the device that is considered (they are related to the state of the node). They are not affected by external factors and they are not mutable. They include Buffer Occupancy (BO), residual energy, available memory, processing power, duty-cycle of the wireless antenna and so on.

3) Application-oriented metrics

Application-oriented metrics depend on the type of software that is running or the need of the device or user. So, to be clear, "application" does not necessarily means "software". A person using a VoIP requires a low delay. A small sensor inside a wall needs to save as much energy as possible. These metrics can be constant or change through time, like the sensor inside the wall minimizing energy consumption (constant) or a smartphone user switching from VoIP to downloading to watching a multimedia stream (varying).

It is worth noting the nature of the environment in which devices evolve affects all the above metrics. If the device has a high velocity (its relative speed compared to the environment is high), important fading can cause many transmissions to fail. This implies increased ETX, lower efficient throughput and higher delays. A higher ETX means more transmissions to send the same data, thus more energy usage.

The term "Performance Index" (PI) refers to values measured throughout an experiment (simulation) and used to realize the evaluation of performance. The end-to-end delay, average throughput usage, total energy consumption, Packet Delivery Ratio (PDR) are some examples. They usually represent the averaged value from

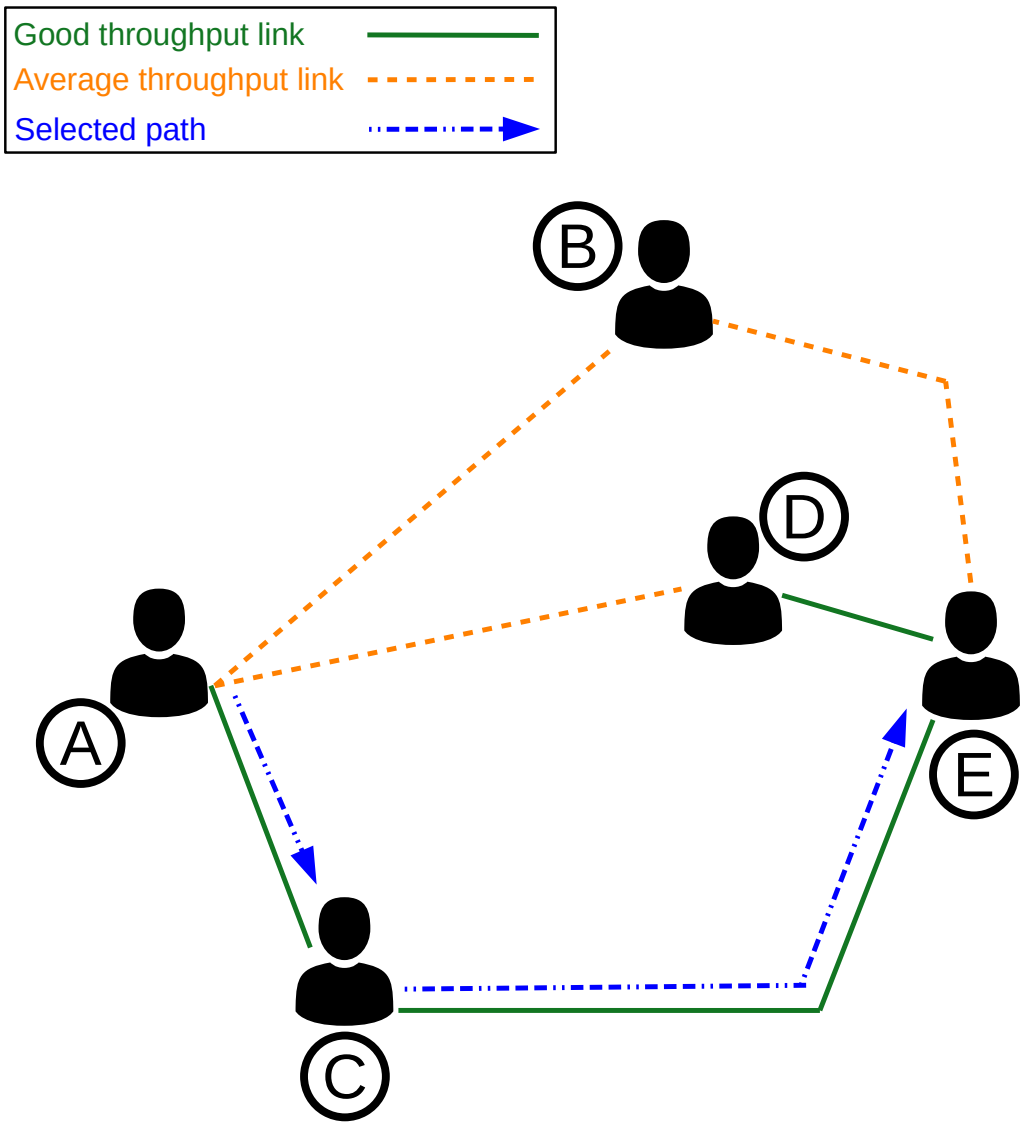


Figure 1.3: Using only network-oriented metrics, the chosen path is the one with the highest throughput.

all nodes on the whole simulation. They are shown on graphs in the results sections of the contribution chapters (chap. 3, 4 and 5).

To avoid confusion, values measured to determine the best path will be referred as "metrics" and simulation results will be referred as "Performance Indices" (or "PIs").

Figures 1.3, 1.4 and 1.5 show a simple example topology where user A wants to connect to user E for a VoIP conversation. User A is directly connected to users B, D and C who are directly connected to E. There are thus three possible paths for A to send his data. On figure 1.3, A computes the best path using only network-oriented metrics. On figure 1.4, the best path is found using network-oriented metrics and system-oriented metrics. In the last case, depicted on figure 1.5, A finds the best path with network-, system- and application-oriented metrics.

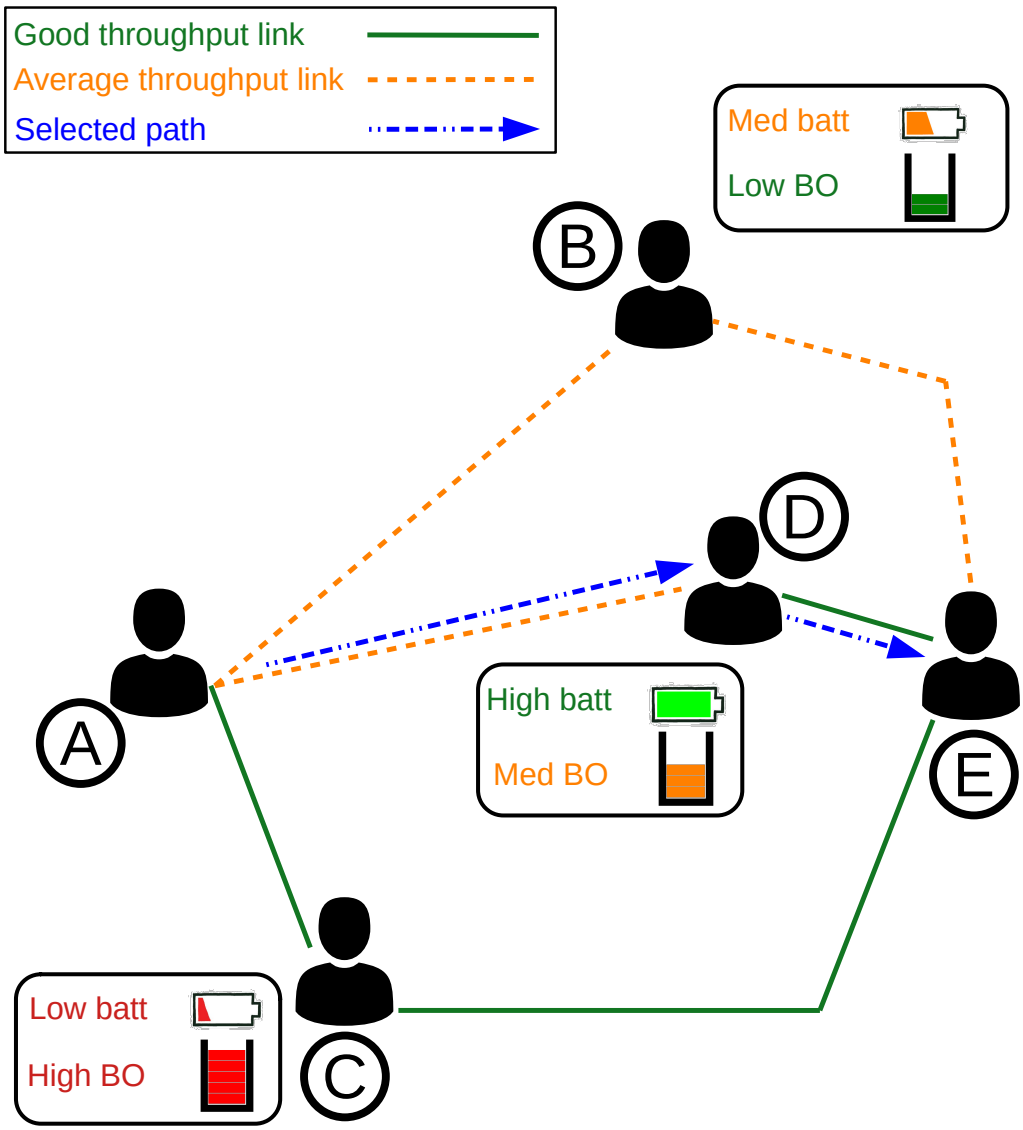


Figure 1.4: Taking into account system metrics as well refines the choice of path. "BO" stands for Buffer Occupancy.

Figure 1.3 illustrates the case where the best path is computed by only considering network-oriented information. On the three possible paths, the one with the highest throughput is $A - C - E$ (bottom of fig. 1.3). The middle path, $A - D - E$ has a low throughput link for the first hop (from A to D) then a high throughput link (from D to E). The top path ($A-B-E$) has the lowest throughput. In this case, A will choose the bottom path (depicted by the blue arrows on fig. 1.3) because it is considered best with the information at hand.

On figure 1.4, the configuration of links and their throughput values are the same compared to figure 1.3. The difference is that on figure 1.4 system-oriented metrics are available as well as network-oriented ones. We see that, although user C is on the highest throughput path, she suffers from heavy congestion and the battery of her device is low. User D has an average battery level and low Buffer Occupancy (BO) and user B has a high battery and average BO, so they both have one "good" and one "average" system metric value. However, given link $D - E$ is

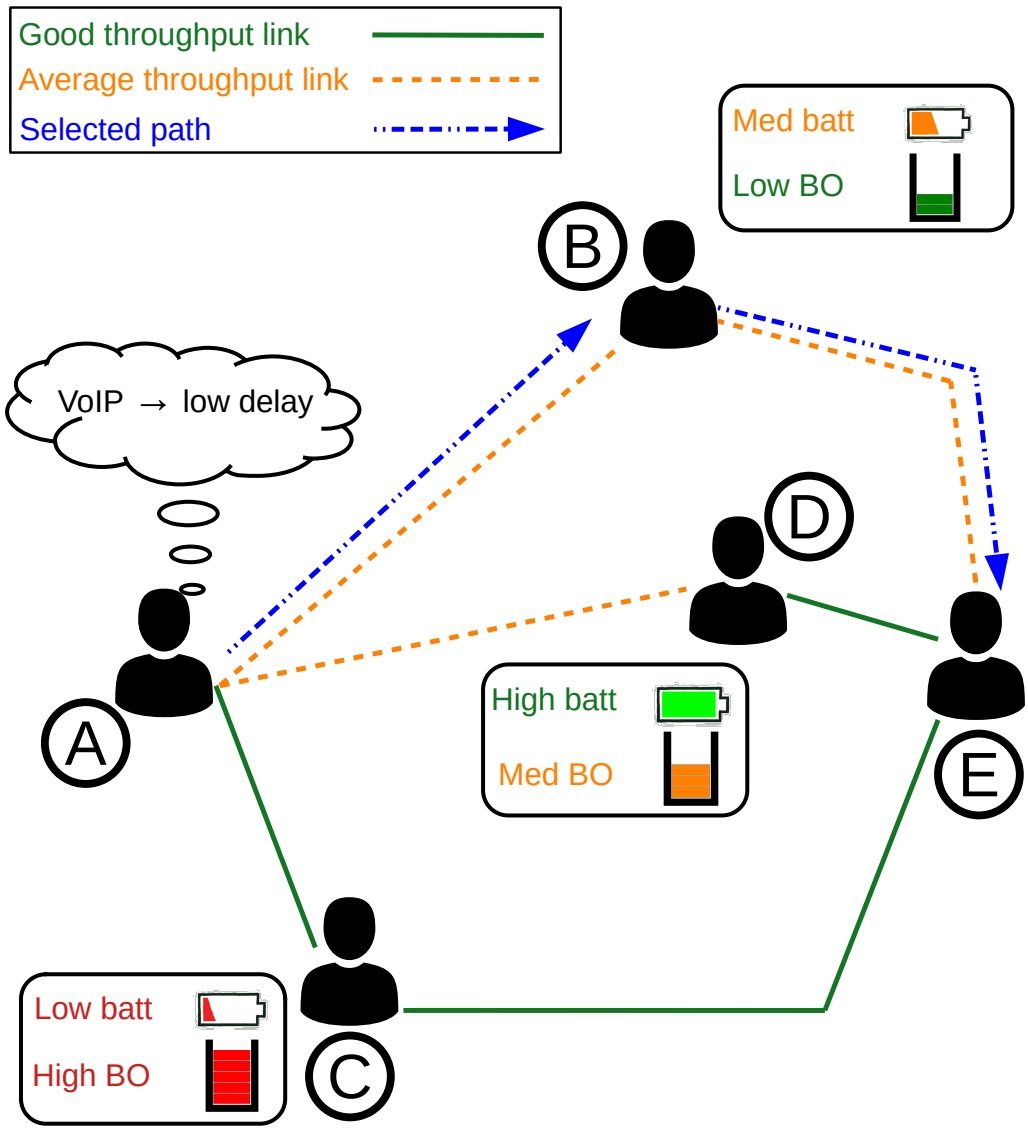


Figure 1.5: Considering network-, system- and application-oriented metrics optimize the chosen path. "BO" stands for Buffer Occupancy.

better than link $B - E$, user A will decide to establish the path $A - D - E$ for the VoIP communication.

Finally, in the case depicted on figure 1.5 user A computes the best path given the three types of metrics: network, system and application. The reasoning for network and system metrics is the same as discussed on figures 1.3 and 1.4. However now, user A also considers his need: VoIP. In order to ensure good QoS, VoIP requires low delay and jitter and only a minimal guaranteed throughput is enough to send voice data. Given this information, the path $A - B - E$ is considered the best because the "average throughput" available on it is enough to accommodate the data and the low BO of user B is interesting as delays are expected to be lower.

Of course, the example shown on figures 1.3, 1.4 and 1.5 is rather simple but it is enough to see that taking into account several aspects - not only network information - can help in taking a better decision for the choice of a path. Mobility can also be

considered as a metric because it creates instability on the topology. See chapter 3 for more details on the subject.

1.2 Limitations and challenges

A major challenge in wireless networking is its dynamic nature. The variations of link quality due to a changing environment, the mobility of users constantly reshaping the topology and the heterogeneous devices continuously connecting and disconnecting from the network are parameters that are hard to control while QoS depends upon them.

The following points depict the shortcomings of a static infrastructure and are highlighted on figure 1.6:

1. The service area is limited. If a user moves out of range of the antenna or if environmental conditions are such that the range of the antenna is lower than usual, a user that is too far away will have no connectivity;

2. The capacity of the antenna is limited. If a sudden spike in network traffic happens and local access points were not designed to absorb such a high amount of packets, the network might crash;

3. Load balancing is thus not optimal. It is possible that at some point in time many users will be located to a very small area. The local access points will be overloaded and antennae somewhat farther away might be only lightly loaded;

4. Resilience is not optimal. If an AP fails, the QoS might drop significantly. No new antenna can be deployed in real-time to compensate for the lost one;

5. Emerging technologies and future needs are not accounted for. For instance, augmented reality and autonomous vehicles have very high network requirements [55]. Current solutions are not adapted to ensure performance and deliver acceptable QoS for such novel applications [56] [57] [58].

Furthermore, catastrophes such as floods, a massive blaze, earthquakes or tornadoes can cause heavy damage to infrastructure. QoS will suffer significantly in such a degraded context. Social events like sport competitions, music festivals or demonstrations are temporary but they still cause an overloading of the network as many users are located in a relatively small area. Catastrophes and social events are two examples of exceptional situations that are challenging to deal with.

Such factors - whether common (like the motion of users) or exceptional (like a flood) - are not controllable by network operators. It is thus very hard to anticipate how the topology will be organized in the future, what needs will become and which devices will be connected. Exceptional events can exacerbate these issues by overloading the network or by causing damage or destruction on the infrastructure.

A more flexible approach to deal with the limitations of a static, dedicated infrastructure and the very dynamic nature of wireless networks is necessary.

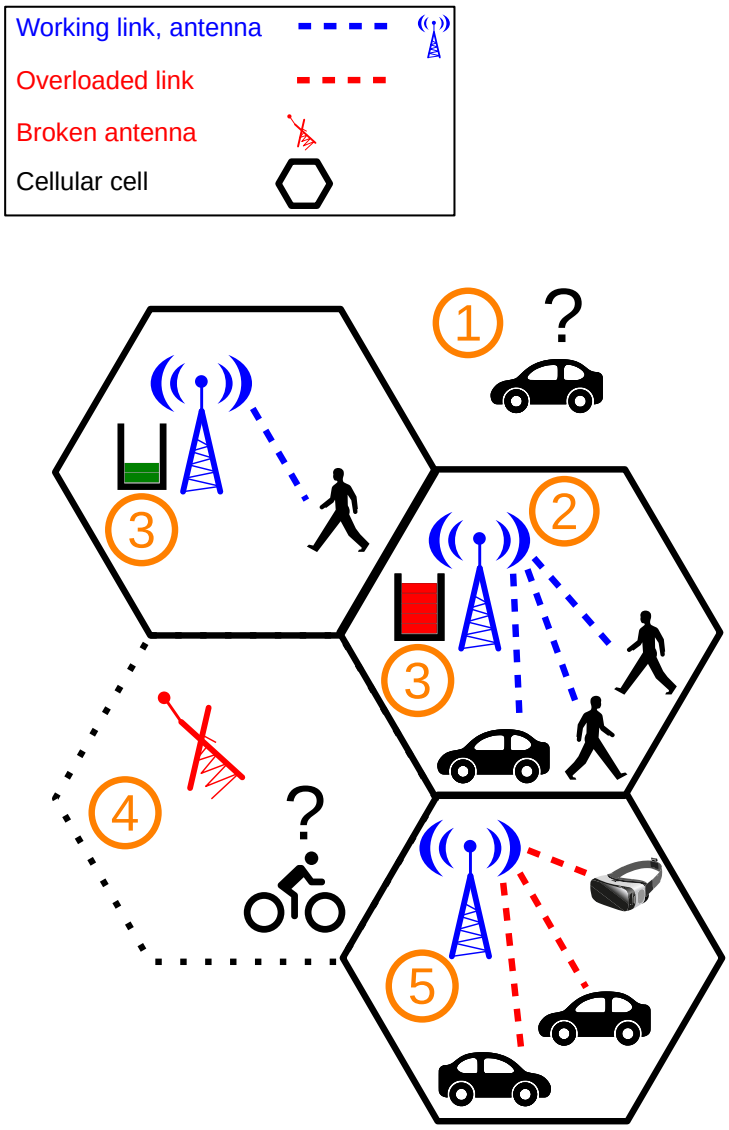


Figure 1.6: Limitations inherent to a static infrastructure.

1.3 Thesis motivation

The subject of this thesis arises from the need to find a more general approach in improving and guaranteeing QoS in dynamic wireless networks compared to what has been achieved in the state of the art so far (see chap. 2).

This work proposes to overcome the aforementioned limitations by better accounting for the different needs of users and devices (application level), the environment in which they evolve (changing network conditions, mobility) and the characteristics of the connected devices (residual energy, processing power).

To do so, let us detail three methods to address the issue:

- Adapt dynamically (proactive) to changes at the network, system and application levels;
- Bring the service closer to the end user to improve QoS and QoE;

- Implement predictive mechanisms to deduce information on users such as mobility profile and trajectory.

An adaptive solution is one that reacts to events happening on the network or at the application level. Allocating more resources to one specific location because of a spike in the amount of traffic is one such example. If the adaptive solution reacts too slowly it will not be efficient as the problem will not be managed in time to continue guaranteeing QoS. On the other hand, reacting too quickly could cause problems as well because a sudden increase in traffic might be the consequence of a temporary problem (loss of connectivity of a node reconnecting right away) that need no actions. If the user is starting a new application that have different requirements from the previous one, it might be needed to connect to another neighbor offering better throughput or lower delay, for instance.

Decentralization is a mean of tackling with the problem of distant transmissions needed when a device is connected to a server located somewhere in Internet. Distant communications imply a higher probability of packet loss, higher energy consumption, more hops to reach the destination and longer delays. On the other hand, when the service is brought close to users, delays are expected to be shorter and less packets are lost, which increase QoS and QoE. Fog computing is an example of decentralization [3] (see chap. 2).

Prediction can be achieved through several means (this will be covered later in this manuscript). It is useful in deducing information about devices such as their mobility profile and trajectory. Users traveling far and fast will be more concerned by handover compared to slower users, devices heading towards a crowded area (high interference) or inside a tunnel (high shadowing) might require a relay node to forward their data back and forth. Predicting the trajectory of devices to anticipate their future position can help in placing relays at key locations. Compared to the adaptive aspect discussed previously, prediction mechanisms do not necessarily concern network, system or application *per se*.

1.4 Problem statement

In the last two subsections (1.2 and 1.3), we saw the limitations of the state of the art and a brief overview on how we will tackle these limitations. Before going into more details about the solutions, we shall state the problem.

In this thesis, we study and implement algorithms and mechanisms to improve the routing aspect of the network (i.e. finding the best path to send data packets from one source to one or several destination(s)) in mobile wireless networks in order to guarantee QoS. Though many issues have been addressed in the state of the art such as taking into account mobility or different needs of users and devices, those issues are dealt with separately. In other words, there is only solution for each case.

The problem is thus to develop algorithms and mechanisms to route packets and that will ensure QoS in all studied situations by taking into account all of the following at the same time :

- The mobility of devices (no mobility, low velocity and high velocity). Note that all devices can have a different mobility at one given time, and each device

can potentially change its mobility pattern (speed, trajectory);

- The density of devices. It can be low (countryside) or high (city, special events). The density can also evolve through time;
- The type of data. The routing algorithms must be able to manage a one packet communication (such as sending one temperature measurement every hour) and a more demanding connection like a multimedia stream. It is possible the type of data flow changes on any device, such as a smartphone switching from a VoIP to a browsing application;
- The needs of the devices. A sensor that needs to maximize battery life over anything else does not have the same needs/requirements compared to a connected car that must communicate with a low delay and
- The type of the device. A sensor, a smartphone, a laptop computer or a connected car are examples of devices with a different architecture (processing power and capabilities), wireless communication technology, battery capacity, etc. The routing solution must allow those different devices to communicate together in order to send, forward and receive data.

The following subsection presents the proposed solutions and how they answer to the problem of routing in mobile wireless networks stated above.

1.5 Proposed solutions

The following propositions aim at fulfilling the methods depicted above.

1.5.1 A distributed objective function to consider different constraints

The first proposition (chap. 3) focuses on:

- Improving QoS using network- and system-oriented data available from users;
- Strengthening the model by considering velocity as a metric to add mobility support and
- Designing a novel infrastructure inspired from Software Defined Networking (SDN).

In an ad hoc network, wireless links between devices are selected based on measurements like Received Signal Strength (RSS) to deduce throughput or the time it takes to receive a response after sending a packet (used to deduce delay). By probing neighbors, it is possible to select the best one according to a global or local policy. If the policy is to prioritize low delay, any node will connect to the neighbor providing the shortest delay. Note that this neighbor might not be the nearest or the device with the highest performance, as other factors such as congestion, high fading or mobility could impact radio transmissions, increasing delay. By considering many metrics, the adverse effects impacting the performance of the network can be compensated to either enhance QoS or, in the worst case, limit the degradation of the service.

The solution then takes this topology building to the next level on two aspects:

The first aspect is the consideration of several metrics into an Objective Function (OF) to achieve the improvement of QoS. Indeed, given most metrics are related and the needs of users and devices are many, it is not optimal to focus on one aspect of the network. For instance, to minimize delay within the network, it is relevant to consider throughput, energy consumption, congestion and Expected Transmission Count (ETX) [54], but other metrics could be useful as well. A high throughput allows to send and receive packets faster. A low energy consumption increases the lifetime of devices on the network and prevents battery saving features from hampering network performance. A low congestion (by smart load balancing, for instance) also helps in reducing delay by minimizing the waiting time of packets in buffers.

The second aspect deals with the update of links. As users travel and change the application they are currently using, the network evolves. Some user can start a download and becomes a poor candidate to forward traffic to others. This is to be detected as soon as possible by neighbors to find another neighbor to connect to before increasing congestion impedes QoS. The computation of new links can also be used to improve a fairly stable topology through time. As new nodes are added (new devices are connecting to the network), more choices become available to establish paths from a source to a destination and it can be worth it to connect to a new neighbor to further improve QoS. Also, whenever a device travels out of range of a neighbor to which a link is established, handover is needed to avoid a disruption of connectivity.

Finally, a SDN-like architecture is used to increase the flexibility of the solution. This makes it possible to adapt policies locally instead of having a global one that does not fit all situations at once. The control plane of the architecture is used to dynamically program the policy for each concerned (sub-)network.

However, shortcomings are present such as the lack of anticipation due to the mobility of users.

1.5.2 Classifying the mobility type of users to ensure service continuity

Though the consideration of velocity as explained in the previous subsection (subsec. 1.5.1) improves performance, it is possible to further improve the model. Indeed, the previous proposition only takes into consideration the raw velocity of users. It is possible to go beyond using Machine Learning (ML) and this is the essence of the second proposition (chap. 4) in which:

- The mobility type of users is deduced using a classification-based ML algorithm and
- The result of the classification is used to elect mobile relays in a flexible infrastructure-less topology.

With the help of ML, the second installment of this thesis focuses on determining the type of mobility of users. As they progress on the studied area, data such as velocity, acceleration and position can be used to infer certain facts. For instance,

a user located on a trail inside a wooden area is very unlikely to be a car. In the same way, a user moving at about 100 km/h on rail tracks is very unlikely to be a pedestrian.

Using such information allows to help deciding in the choice of relays that will be used to serve as access points to other nodes. Such an architecture is composed almost entirely of standard nodes, including relays. Nodes with a profile considered as optimal are selected to act as relays. The amount of relays is not fixed so the topology can adapt to any situation. This is also interesting because dedicated hardware such as RSUs are not required, except for one cellular antenna.

Differentiating the different types of mobility is very helpful but it can be improved further by predicting the future position of nodes.

1.5.3 Predicting the future position of devices to optimize the placement of relay drones

The third and final installment (chap. 5) of this thesis is to move beyond the inferring of the second one by adding position prediction. Thanks to ML, it is possible to anticipate where devices will be located to place relays in advance. This solution is based on two aspects:

- The future position of devices is predicted using a ML algorithm and
- A scoring system is proposed based on current and future positions of devices to find critical locations for the placement of relay drones.

By gathering the location-related data of devices on a specific area, it is possible to recognize their traveling patterns. This is turn can be used to predict the future position of devices following similar trajectories on the same area. Taking into account these future positions allows to position relay drones somewhat between where devices are and where they will be.

To do so, we design a scoring system where the studied area is divided into tiles, represented as a matrix. Each tile has a score depending on the number of devices present. Then, the ML algorithm is run to predict the future positions of devices, generating a matrix of future positions. Merging both allows to find the best location to place a relay drone. This not only permits to ensure a continuous service to devices, but also to ensure continuous satisfying QoS from their point of view.

1.6 Outline

The rest of this manuscript is organized around five chapters. Chapter 2 presents the state of the art and brings to light the major concepts around which the contributions are centered. Chapter 3 elaborates on the first contribution of this thesis which concerns the novel Programmable Objective Function (POF). In chapter 4, the second contribution is explained and is about the classification of the mobility type of users in the IoV. Chapter 5 is the last contribution and covers the prediction part of ML to anticipate the future position of users, allowing a

better positioning of relays. Finally, chapter 6 concludes this work before discussing perspectives.

Chapter 2

State of the art

Résumé français

Les technologies sans-fil peuvent être séparées en deux catégories : d'une part, nous avons celles qui opèrent dans un spectre soumis à autorisation (*licensed spectrum*) et, d'autre part, il y a celles qui utilisent les longueurs d'onde libres (*unlicensed spectrum*). Dans le spectre soumis à autorisation se trouvent certaines implémentations des technologies cellulaires (1/2/3/4/5G+) comme la LTE. Pour ce qui est du spectre non-soumis à autorisation, nous avons la WIMAX (autre implémentation de 4G), LoRa/LoRaWAN (pour l'Internet des objets), la WiFi ou encore des technologies comme le ZigBee ou Bluetooth (basés sur le IEEE 802.15) pour les transmissions en courte portée.

Nous allons discuter de deux aspects différents concernant l'architecture réseau : les aspects de virtualisation et de centralisation/décentralisation. La virtualisation permet d'enlever les barrières liées au matériel dédié. En effet, un serveur spécialisé pour héberger un service précis manque de flexibilité et certaines situations comme une panne d'équipement ou une surcharge ponctuelle du serveur peuvent mettre en péril la qualité de service. Les machines virtuelles permettent d'exécuter plusieurs systèmes d'exploitation simultanément sur une même machine. Cette virtualisation est transparente pour une application executée dans un tel environnement. Cela permet d'adapter le nombre d'instances selon le besoin. Pour aller plus loin, il est possible d'utiliser des conteneurs (*containers* en anglais). Ceux-ci sont plus légers qu'une machine virtuelle et peuvent de ce fait être déployés proche des utilisateurs dans le *edge*.

Déplacer des services dans le *edge* est une forme de décentralisation. Cela permet de diminuer le délai réseau, d'éviter la surcharge de liens vers Internet, d'économiser de l'énergie et augmente la résilience du réseau par rapport à un modèle centralisé. Dans ce dernier, un centre de données est relié à Internet et les utilisateurs s'y connectent pour accéder au(x) service(s) qu'il fourni. Il est possible de rapprocher ce service en mettant en place un micro-serveur qui peut par exemple être connecté à un réseau local. Les utilisateurs communiquent avec le micro-serveur local qui communique à son tour avec le centre de données, si besoin. Pour aller plus loin, il est possible de délocaliser le service directement sur les appareils faisant partie du *fog*. Cela permet d'avoir un réseau sans infrastructure où tout appareil peut contribuer à l'effort collaboratif.

Plusieurs autres technologies permettent d'améliorer les performances dans le *edge*, comme la virtualisation des fonctions réseaux (*NFV*), la séparation des plans

données et contrôle au sein du réseau (*SDN*), la mise en mémoire de données susceptibles d'être requises par d'autres appareils (*caching*), le partage de ressources non-sollicitées ou encore la séparation en couches du réseau (*slicing*).

Plusieurs algorithmes de routage ont été développés pour l'acheminement des données dans les réseaux sans-fil. RPL est un protocole très connu et a beaucoup été étudié. Il fonctionne en construisant une topologie en forme d'arbre. C'est la racine qui en initie la construction et cela permet un routage plus simple des données. RPL est par définition très flexible et il est utilisé pour la base des travaux de cette thèse.

L'apprentissage artificiel est une sous-catégorie de l'intelligence artificielle et permet de classifier des données ou d'effectuer des prédictions sur certaines données. Pour ce faire, il faut d'abord entraîner l'algorithme (on parle alors d'apprentissage supervisé) sur un jeu de données connues avant de l'utiliser.

2.1 In a nutshell

In this chapter, we cover aspects of wireless networks and tools to enhance performance in the network. We first discuss selected wireless technologies. Next, we explain the differences between a centralized, infrastructure network and a decentralized, infrastructure-less network better adapted to our problem (as discussed in section 1.3). After the part concerning infrastructure/architecture in wireless networks, we present routing protocols in the IoT. Finally, we cover Machine Learning (ML) by explaining how it is used to improve efficiency in our case.

2.2 An overview of wireless technologies

Wireless networks based on an infrastructure need dedicated access points to connect users and devices to Internet. This is depicted on figure 2.1 where different users with different needs connect to some service (represented as the datacenter on fig. 2.1) through Internet. As explained in section 1.2, such a dedicated infrastructure has limitations.

2.2.1 Licensed spectrum

3GPP [59] is the group of organizations responsible for writing the 3/4/5G+ network specifications. The first versions 1/2G (before the entity was named 3GPP - see section 1.1.1) was used for telephoning services and the infrastructure was based on circuit switching [60]. 3G uses both circuit and packet switching and more recent versions (4/5G) are based on pure packet switching. Internet services like multimedia or file sharing are possible with the last versions because a high throughput is required. An example implementation is the Long Term Evolution (LTE)[1].

The rapid increase of users and connected things have motivated the development of 5G. It allows to connect a massive amount of objects to the network, to support emerging services like augmented reality and services requiring high throughput like Ultra-High Definition (UHD) videos [60]. Such performance is possible mainly

[1] LTE does not meet the requirements for 4G (it is classified as 3.75G in [60]), but the improved version LTE-Advanced (LTE-A) is a 4G.

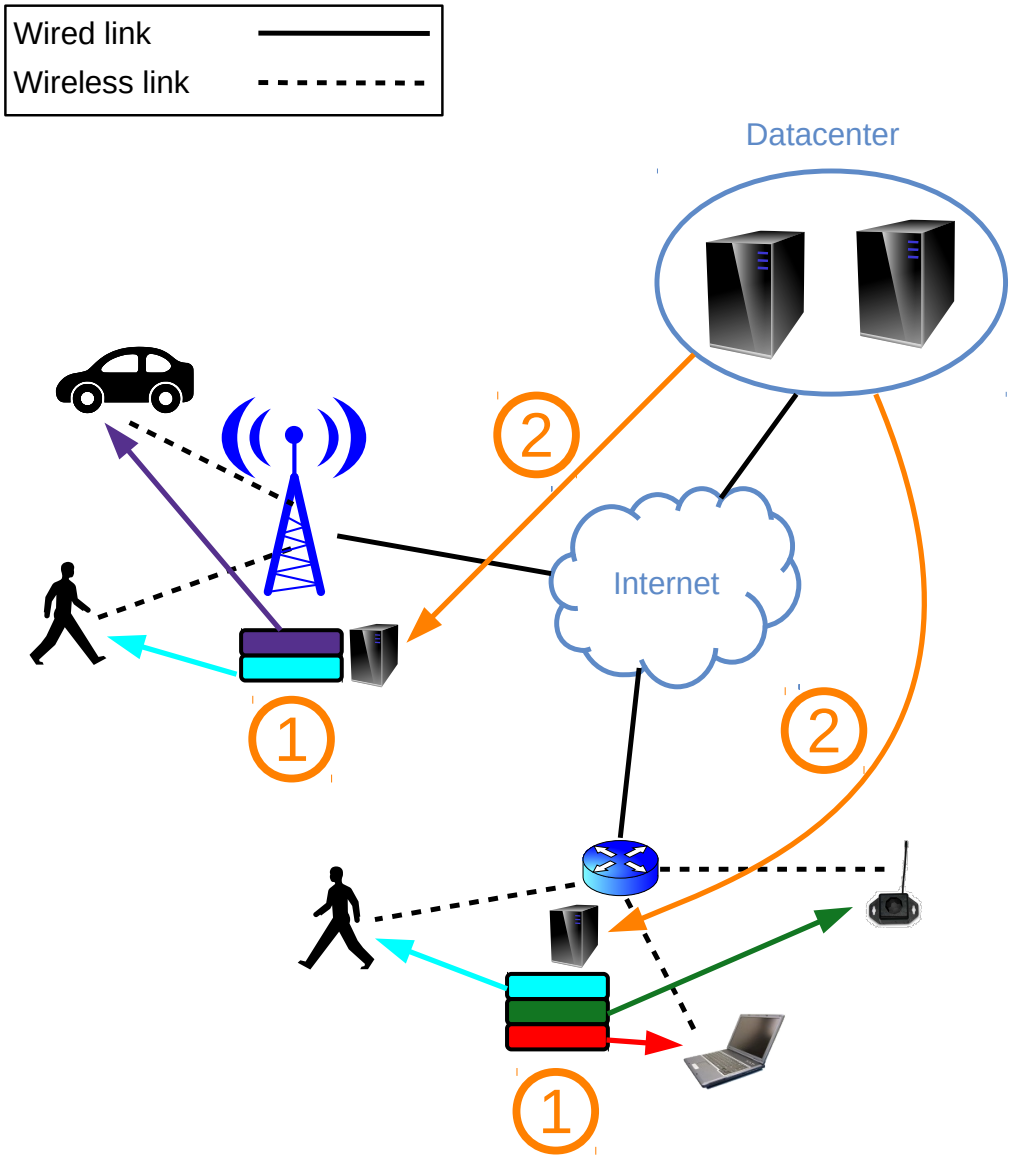


Figure 2.1: A representation of a wireless network infrastructure. Virtualization (1) and decentralization (2) are more adapted to the IoMT.

through the use of massive Multiple-Input Multiple-Output (MIMO) and several "layers" of cells [61]. MIMO is the use of several antennae elements (antenna array) to transmit/receive data, increasing capacity. In Massive MIMO, the principle is extended by adding even more elements that can be spread around in a cell, improving spectral and power efficiency. Instead of relying on one 5G macrocell to cover an entire area, several smaller cells can be used inside a macrocell. This scheme reduces interference and can better account for users' heterogeneous needs.

2.2.2 Unlicensed spectrum

Worldwide Interoperability for Microwave Access (WIMAX, IEEE 802.16) is an unlicensed 3.75/4G[2] technology and is thus similar to LTE/LTE-A.

[2]Like LTE, fixed-WIMAX is not a 4G [60] but its improved version Mobile-WIMAX is.

LoRa/LoRaWAN is a wireless technology designed for the IoT [62] [13]. LoRa refers to the physical layer whereas LoRaWAN designate the protocol stack on top of LoRa. It consumes low power, offers low throughput and is able to communicate up to a range of several kilometers. A LoRa network is composed of end devices (usually sensors/actuators) connected to gateways. Several end devices can be connected to a single gateway. Gateways are connected to servers which receive and process data from end devices. The link between an end device and the gateway uses the LoRa/LoRaWAN stack whereas the link between gateways and servers operates on a faster technology like 4/5G or Ethernet. LoRa is adapted to small transmissions like the upload of measurements from sensors to servers or the sending of commands to actuators from servers, but its low throughput makes it unusable for more intensive data transfers. SigFox is a technology [14] comparable to LoRa and operates similarly.

IEEE 802.11 (WiFi) is a physical/link layer protocol designed to be used for Wireless Local Area Networks (WLAN) [8]. A WiFi device connects to a router which is connected to a wider network like Internet. Devices can also directly connect to one another, removing the need of an infrastructure. Several standards exist for WiFi to address different needs like 802.11a supporting Orthogonal Frequency Division Multiplexing (OFDM) in the 5 GHz range, 802.11p for vehicular environment and 802.11s for mesh networking to name a few [8]. Throughput can achieve values up to several hundreds of Mbit/s and the maximum range is about 100-150 m. In chapters 3, 4 and 5, we use WiFi to connect devices together on a topology.

ZigBee [12] is a short range technology based upon the IEEE 802.15.4 standard. It is intended to connect low-powered devices together such as sensors in a tree-like fashion. The topology consists of one coordinator (the root of the tree), routers (internal nodes) and the end devices (leaf nodes). In ZigBee, there are two kinds of devices: Full Function Device (FFD) that can act as coordinator or router and Reduced Function Device (RFD) which is the end device (sensor, actuator). ZigBee is somewhat similar to LoRa/SigFox, but for short-range communications.

Bluetooth [63] is based on IEEE 802.15.1 and operates on a shorter range, with higher throughput and consumes more energy compared to ZigBee [64]. In Bluetooth, two devices are directly connected to one another and its purpose is to remove the necessity of cables when connecting devices such as keyboards, mice, audio headsets, etc.

2.3 From infrastructure to ad hoc networks

In this section, we discuss different types of architectures that exist to connect users. Architecture is somewhat vague and can refer to different aspects of how the network is built. We first discuss the perspective of a virtualized/non-virtualized server or datacenter (see the examples shown on figs. 2.2, 2.3 and 2.4). Then, we explain the architecture from a centralized/distributed point of view (2.5, 2.6 and 2.7). It is possible to combine both aspects together to a different degree.

Virtualization, shown as (1) on figure 2.1, is a mechanism of migrating hardware functions to software [55]. This allows more flexibility in terms of providing a service adapted to local users. This is discussed in section 2.3.1. Decentralization is the distribution of an architecture from a central location managing all services to local

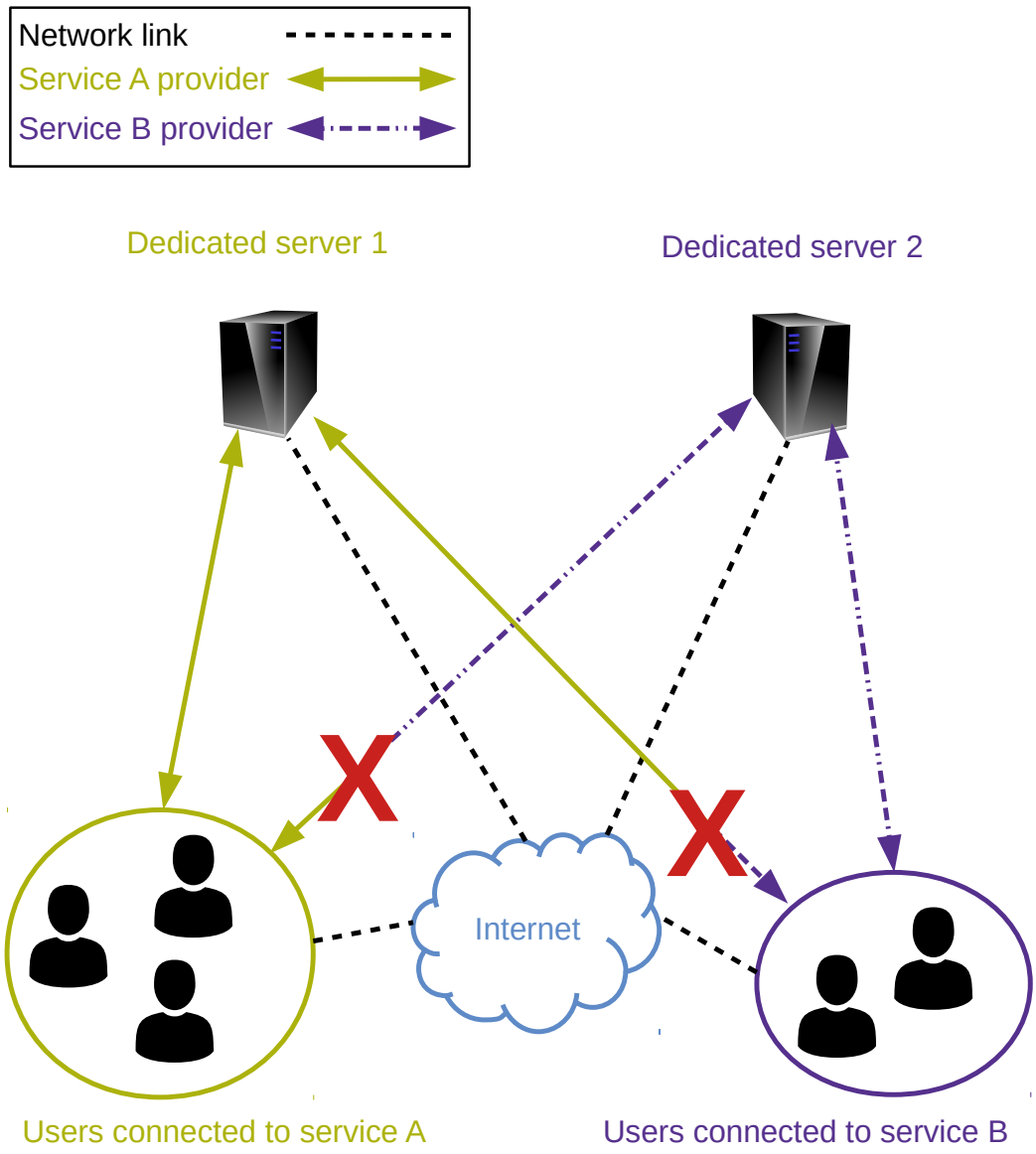


Figure 2.2: An example architecture with two servers, each providing one different service.

geographic areas. This is illustrated as (2) on figure 2.1 and aims at bringing the service closer to the end user. This is covered in section 2.3.2.

2.3.1 Removing hardware barriers through virtualization

The three figures 2.2, 2.3 and 2.4 shows the same context where two distinct groups of users connect to some service (services A and B) through Internet. Example services are streaming, VoIP, website browsing, data downloading and so on.

On figure 2.2 we see a "classic" architecture with no virtualization. There are servers 1 and 2 and they are configured to provide services A and B respectively. Server 1 cannot provide service B and server 2 cannot provide service A. In order to change the services on a server, additional installation is required. In the case where an application only runs on a specific Operating System (OS), it can be time-consuming to modify the server to provide another service. This architecture is not

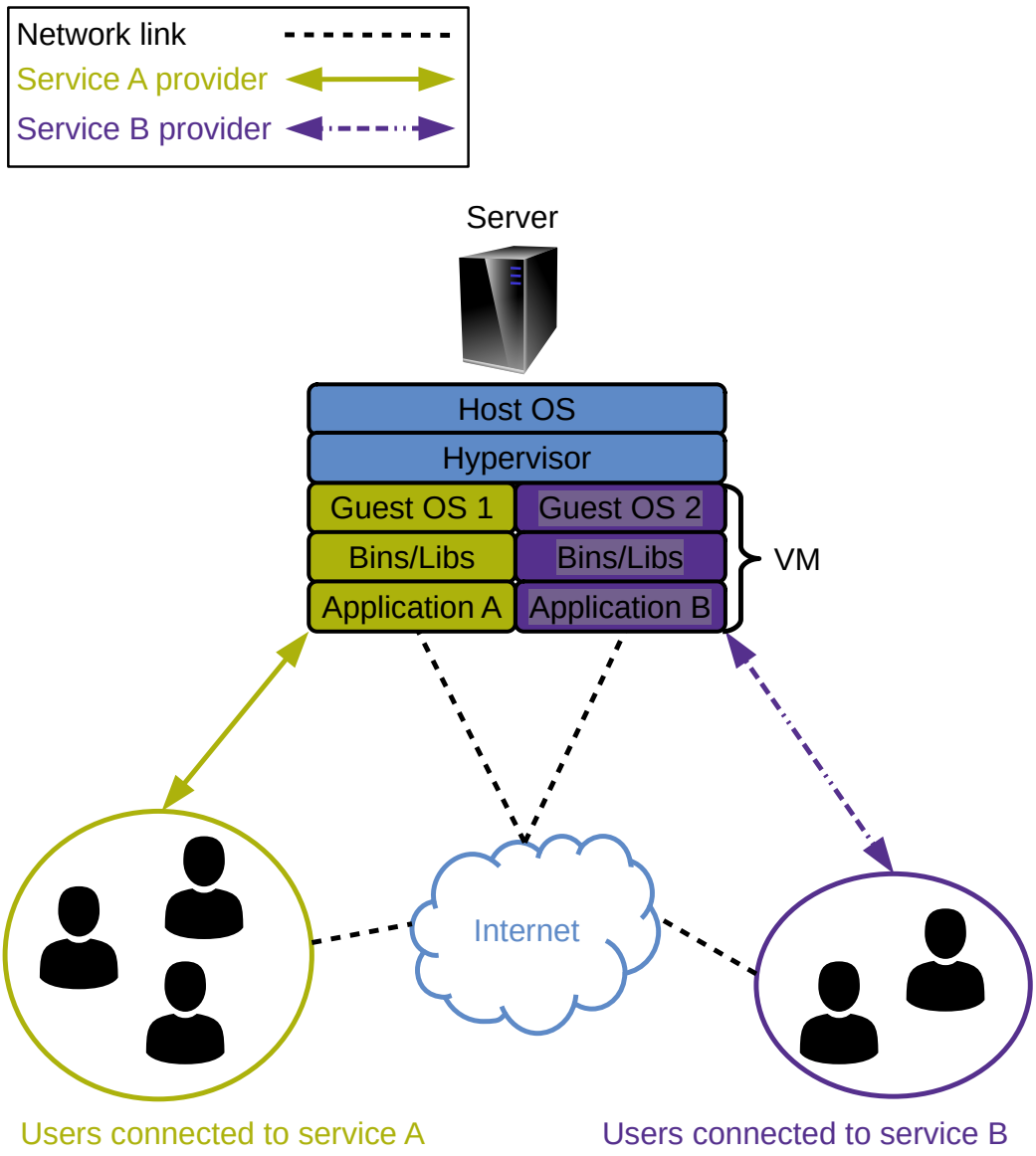


Figure 2.3: The modified architecture with one server providing two different services [3].

very flexible nor reliable. Indeed, if one server fails the service will not be available any longer. Furthermore, if very few or no users are connected to one service and many are connected to the other, one server will be overloaded and the other will be idle most of the time: load-balancing is uneven.

Figure 2.3 represents a modified version of the "classic" architecture from figure 2.2 where there is one server running Virtual Machines (VMs). A VM is a flexible option that allows to run several instances of guest OSs at the same time on one physical machine (server) or several (datacenter) [65] [66]. It is possible to save the state of a VM and continue running it later or on another physical machine. Applications running on the guest OS are not aware of the host OS. This gives VMs a greater deal of flexibility compared to a "classical" architecture as depicted on figure 2.2.

On figure 2.3, we can see that users requesting services A and B can be connected

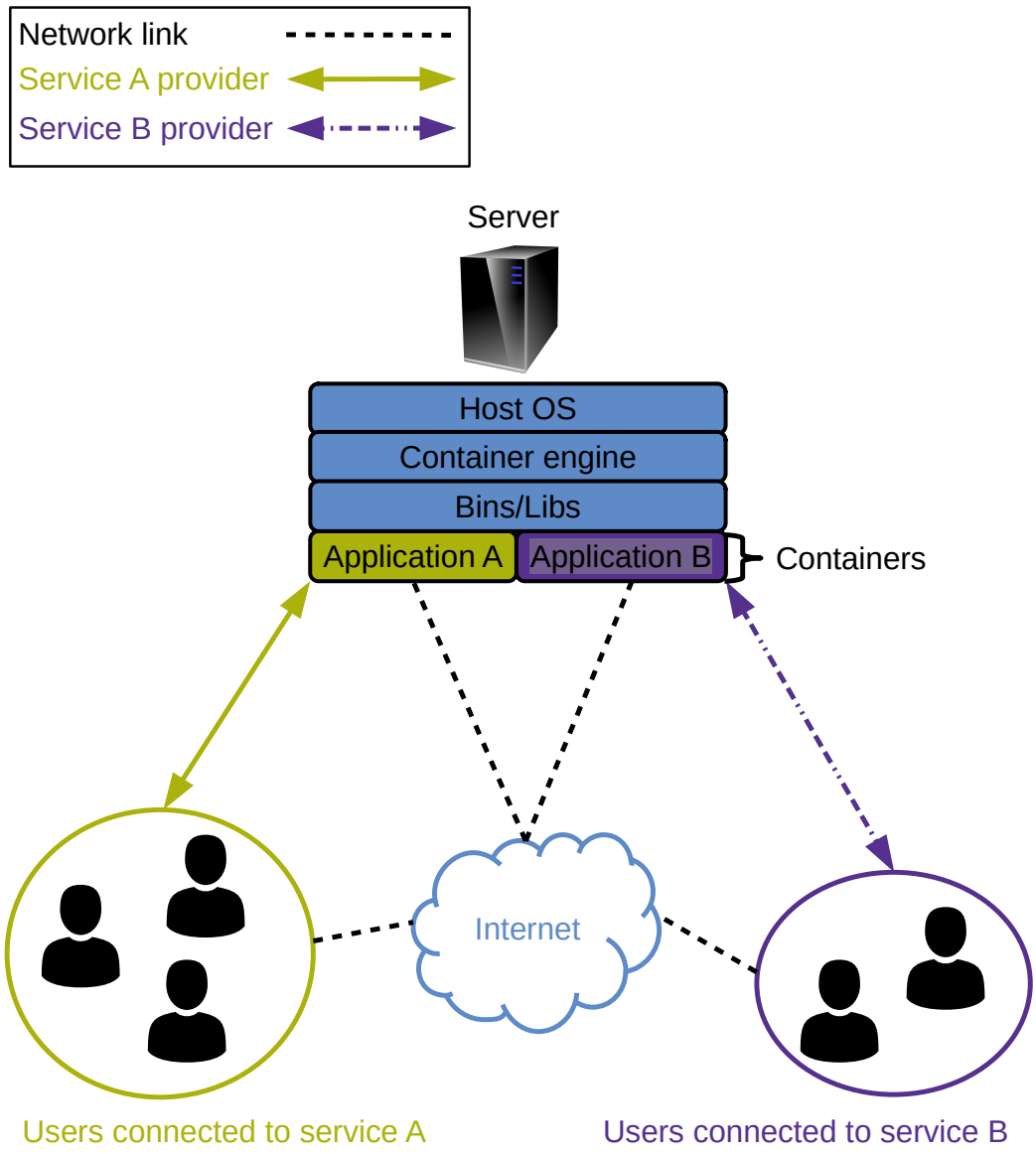


Figure 2.4: The architecture using containers is lighter and more flexible compared to VMs [3].

to the same server. If a new service is needed it is possible to add a new VM to run it. If very few users are connected to one service, resources will naturally be used by other (more solicited) services because they all run on the same physical computer.

On figure 2.4, the same architecture as in figures 2.2 and 2.3 is adapted by using containers. Containers are lighter than VMs and instead of running a full guest OS, only libraries required for the applications are run [65] [66]. Given containers are light, they can run on the edge to bring services closer to end users.

2.3.2 Bringing the service closer to end users with decentralization

In a fully centralized architecture there is one server providing one or several services. This is not very flexible and if the server fails the service is interrupted. To remedy this, it is possible to connect several servers together to form a datacenter. The

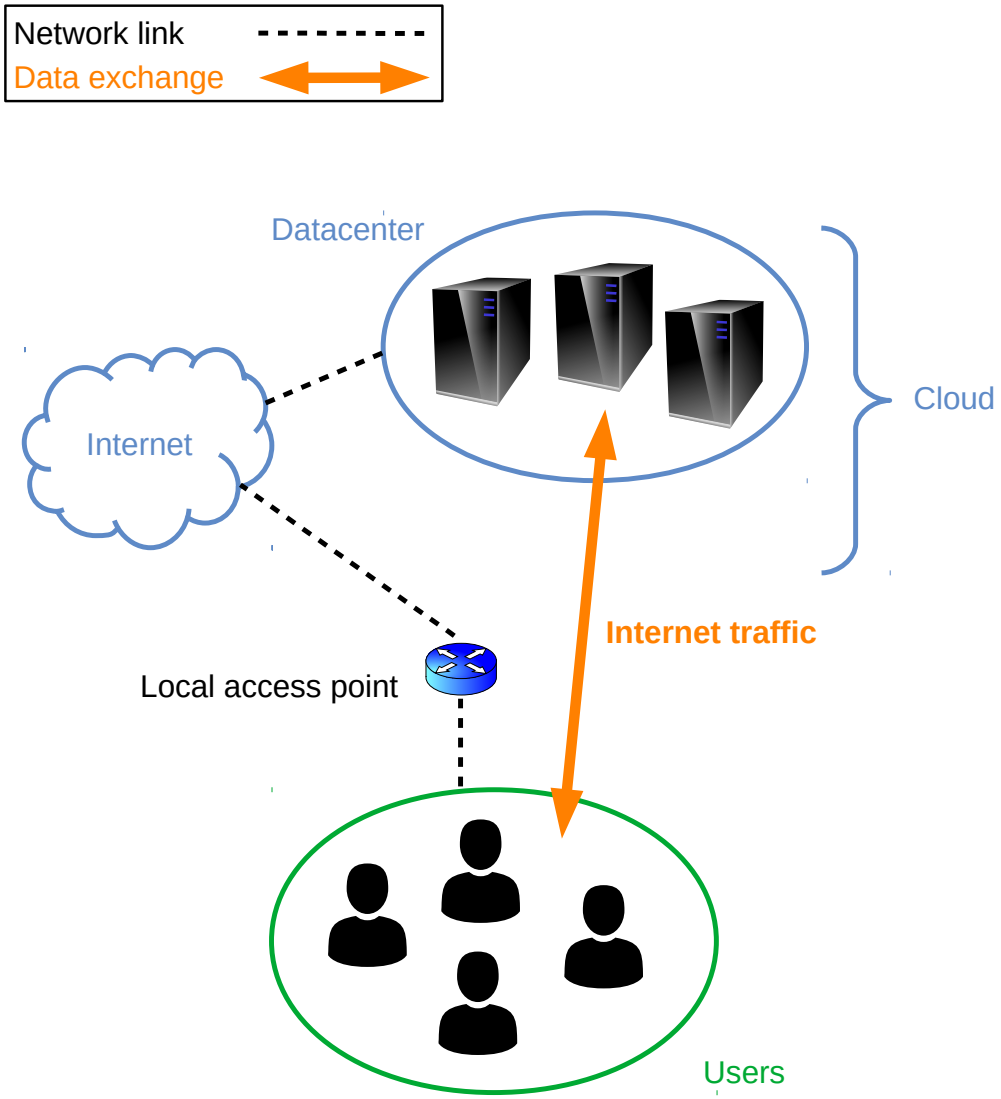


Figure 2.5: A datacenter in the cloud provides some service to users.

load is balanced on several servers, increasing the pool of available resources and reliability.

Cloud computing [3] is the deployment of technologies in the cloud to remove the constraint of having to buy and maintain potentially expensive hardware. Cloud computing can be Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) or Software-as-a-Service (SaaS) [67] [68]. In IaaS, hardware (CPU power, storage, network) is rented. PaaS is the rent of a platform to develop and run software and SaaS is the rent of a software running in the cloud.

Figure 2.5 shows an example of a topology where several users that are geographically close must each connect to a datacenter to access some service. The datacenter and Internet are in the "cloud" part of the network. A local access point (that can be a WiFi router in a shopping mall or a cellular antenna) gives users the Internet connection they need to access the datacenter. This topology generates a significant amount of traffic pictured as an orange arrow labeled "Internet traffic"

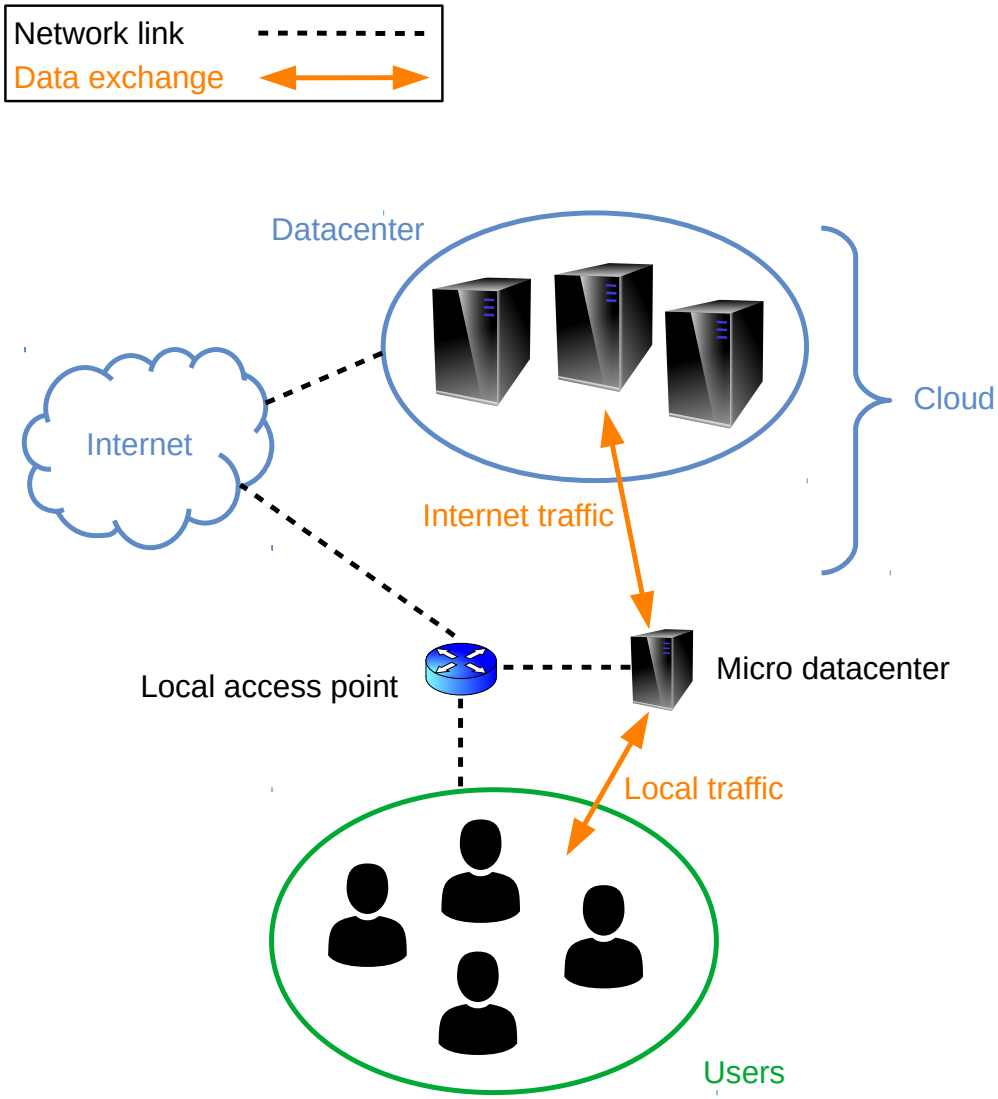


Figure 2.6: Cloudlets enable services to be located closer to users through the use of a mini datacenter.

on figure 2.5[3] because each user maintains a connection to the datacenter through Internet. Another drawback is delay due to the distance between users and servers.

Cloudlets are used to bring the service closer to users [69] [70] [71]. The idea is to deploy servers at the edge to lower the delay to access the service and reduce throughput consumption as less transmissions to the central server located in the cloud are needed.

The topology shown on figure 2.6 is similar to the one depicted on figure 2.5 with one difference: a mini datacenter is located at the edge, near users. Instead of having all of them connect to Internet to get some service, they can connect to the mini datacenter. This can be a simple server connected to the local network and running containers specific to local needs [3]. It is connected to the datacenter

[3]The arrow is thicker compared to the equivalent arrows of figures 2.6 and 2.7, see below.

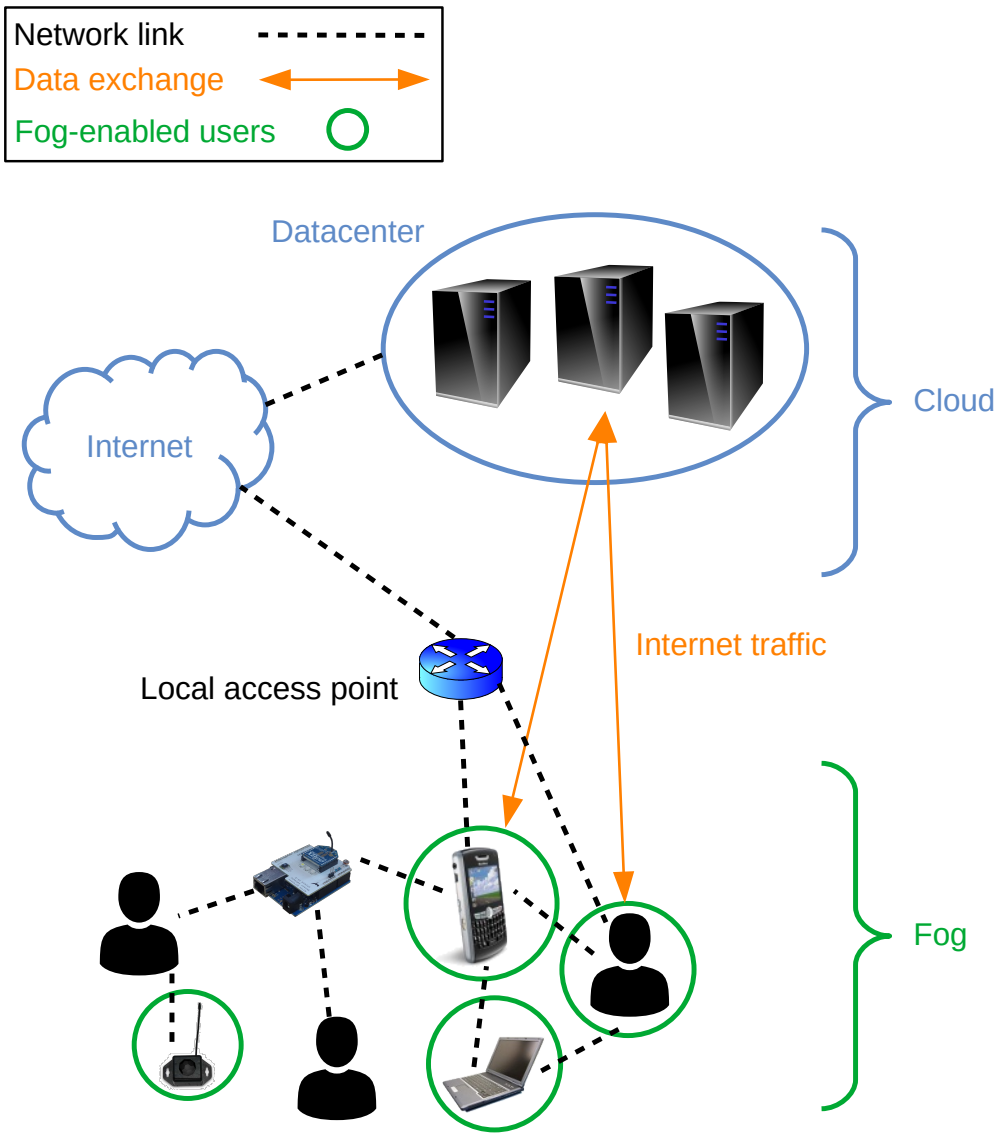


Figure 2.7: Fog architecture is even more decentralized compared to cloud and cloudlets.

from which new container instances can be downloaded if need be. Cloudlets allow to lower the amount of Internet traffic and delay because of the proximity between the mini datacenter and users.

Fog computing goes one step further compared to cloudlets. It is the deployment of services at the edge [72] [73] [74]. Fog computing brings what a cloud offers closer to users but in an even more distributed way compared to cloudlets. A fog-enabled device acts as a tiny datacenter and such a device can be anything from sensors to computers. This is interesting because it can be deployed in ad hoc networks so users can directly provide service to one another. Internet traffic is even more reduced compared to a cloud topology and cloudlets (see figs. 2.5 and 2.6, respectively).

In the fog topology depicted on figure 2.7, any user can potentially be part of the collaborative effort to create the distributed fog. In this example, four devices are acting as tiny datacenters (they are the green circled users on fig. 2.7).

2.3.3 Unlocking edge computing

Virtualization and decentralization combined together can greatly enhance QoS in the IoMT [3]. Distributing services and resources geographically in close proximity to users is referred to as "edge computing" [75] [76]. Edge computing has been gaining interest since the past few years. It enables the deployment of large scale services without the need of a data center or other expensive architecture. Given smaller devices like smartphones are very much present nowadays and their cost is decreasing each year, edge computing is a viable solution to improve performance in networks, including the IoV.

Network Function Virtualization (NFV) [77] [78] is the decoupling of hardware functions from network equipment to software. Removing the proprietary constraint and having a virtual architecture is significantly more flexible for operators so they can adapt the network infrastructure more easily with less costs. This is different from virtualization such as VMs and containers in that it concerns the managing of the network.

Software Defined Networking (SDN) [79] [80] [5] [81] is an architecture in which the control and data planes are separated. This increases flexibility because both planes can be managed without affecting one another. This allows the deployment of programmable networks (dynamic modifications of the control plane) without modifying the data part. The network can be reconfigured as needed to provide new services or increase performance to specific parts of the network with minimal costs as no physical modifications to the network are required. This paradigm is interesting because it makes it possible to modify the network in real-time (concept of programmable network). This is adapted to very dynamic topologies with heterogeneous traffic like the IoV.

Other concepts can be implemented such as caching or sharing of resources from devices [82]. In caching, when a user accesses some data from a remote server or other user, the data is kept in memory in the case nearby users are interested in receiving this data. Users near the same location have a higher probability of using the same data (such as downloading the same part of map from OpenStreetMap). (Re)sharing the data locally lowers the consumption of network links to Internet.

Sharing of unused resources can optimize network performance through better load balancing. In [83], for instance, authors propose the use of park cars equipped with wireless connectivity to help users connect to Internet.

Network slicing [84] [85] refers to the concept of separating network resources into several instances. An instance can be defined as a frequency or time range. It is possible to optimize each instance to fulfill a given function (e.g. minimize delay). Instances can be allocated in real-time if, for example, another instance is overloaded with many users (allocating more communication time or a larger bandwidth).

2.4 Data transportation in fog computing

Apart from the architecture, the choice of routing protocols is very important. Many routing protocols have been defined and are used in wireless networks [86] [87]. Due to less stability and performance in wireless networks compared to wired networks, routing protocols need to be lighter in terms of overhead and must react

faster to changes. Wireless equipment are often on battery power and energy consumption is a major concern as well. Users are potentially mobile especially with the popularization of devices like smartphones and the increasing usage of sensors and actuators.

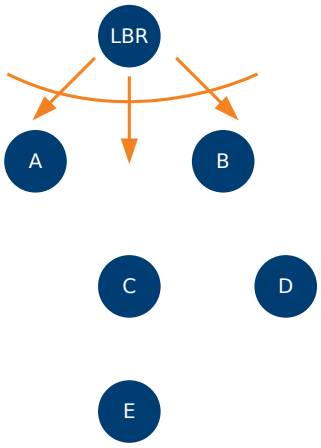
The Ad hoc On-Demand Vector (AODV) [88], is a routing protocol where a route is established whenever data needs to be sent. The source node broadcasts a Route Request (RREQ) packet to all neighbors. A node receiving the RREQ for the first time will forward it on all links (except the one where it came from) if it is not the destination of the RREQ. If other similar RREQs are received they will be ignored. The first RREQ to reach the destination is considered to have traveled on the best path. A Route Reply (RREP) is then sent back on the same path to establish the route between the source and destination. Drawbacks of AODV include the delay from computing the route to the sending of data, the fact the established path is not necessarily the most optimal and the lack of path adaptation if conditions change.

The ADaptive Access Parameters Tuning (ADAPT) algorithm [89] uses several metrics to focus on one aspect of performance. It is designed for WSNs to improve reliability by measuring the delivery ratio of packets and by modifying MAC parameters in order to reach the desired PDR. The ratio is smoothed over time to avoid sudden changes of having too much impact. There are two thresholds in this algorithm: when the lower one is reached, it means the delivery ratio is not good anymore and the algorithm is run to correct the problem. The upper bound is used to avoid unnecessary runs of the algorithm which would consume too much energy. These two thresholds are set to specific values to achieve the best performance.

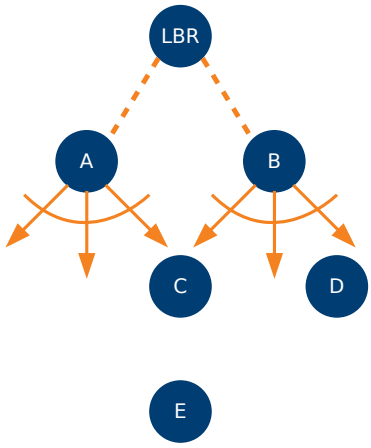
RPL is a protocol particularly well-suited for unstable wireless networks. It was first drafted back in 2009 and it became RFC 6550 in 2012 [11]. It's purpose is to connect devices within a Low-Power and Lossy Network (LLN). A LLN is usually a wireless network of sensors. Error rate can be high and energy consumption is a priority. A high throughput is not required. However, RPL is designed in such a way as to allow developers a great deal of flexibility.

RPL uses a tree-like topology called Directed Acyclic Graph (DAG) rooted at one or several sinks. For each root (or sink), at least one Destination Oriented Directed Acyclic Graph (DODAG) is needed. A sink might also act as a border router, connecting the topology to some wider network like Internet. It is possible to define several DODAGs (called instances) according to different or specific needs. For example, one instance could focus on energy-saving and another one could favor throughput, allowing a joining node to select the best instance according to its needs.

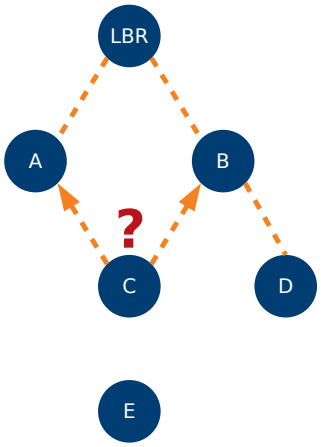
An instance is defined through an associated Objective Function [90] [91]. The OF takes metrics [92] as input to compute the rank. Metrics can be network-related (delay, throughput, Expected Transmission Count [54] (ETX), etc.), system-related (residual energy, buffer occupancy, available CPU cycles or RAM, etc.) or even related to the environment (such as interference level). The rank is used to score the different nodes that are part of the DODAG. A low rank is better than a high one. When a node wants to join the DODAG, it requests the rank of nodes in range and connect to the one with the lowest rank. One property of the rank is that it can only increase to prevent the formation of loops. It can be based on additive or non-additive metrics. When additive, the value of the metric is the total value from



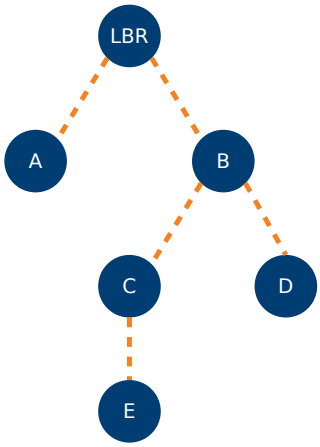
(a) Initial broadcast from the sink node.



(b) Nodes A and B connect to the LBR. Wireless links are depicted as dashed lines.



(c) C may choose between A or B.



(d) The completed topology. Note that C has selected B as parent.

Figure 2.8: An example of the building of a topology in the RPL protocol.

current node to the root (the whole path) whereas the non-additive case refers to the value of the current link only.

An OF can also combine any metrics to any extent (by using a factor to modify the value of the metric or a predefined threshold, perhaps). This allows more complex needs to be taken into account. For instance, one might consider delay but also take into account residual energy in an exponential manner to prevent nodes from running out of energy. This would make a node with a low battery have a very high rank and it will greatly lower the probability of any other node wanting to connect to the low-battery node. With less traffic to forward, the node will be able to reduce energy consumption and network lifetime will increase.

There are two main advantages to RPL. The first is that the resulting tree-like topology allows for simple data forwarding: nodes sending data upwards only need to send it to their parent (no routing required). The second advantage is the fact that

a node needs only to probe its neighbors for their rank, so almost no computations and only a small amount of memory are needed on all nodes but the root. RPL is thus a fine choice for a many-to-one configuration. It is worth mentioning that it is possible to send data downwards but routing tables are needed (although only a reduced set containing the sub-tree rooted at the current node). The challenge is the design of OFs to obtain the best performance or to reach the desired goal.

Performance evaluation of RPL was achieved early on [93] [94] [95] and several enhancements were promptly proposed as well [96] [97] [98].

Figure 2.8 shows an example of the building of a DODAG. The LLN Border Router (LBR) is the sink (or root) of the topology and initiates the building process. The sink starts by broadcasting a DODAG Information Object (DIO) [11] (fig. 2.8a). The DIO is a signaling packet that contains information relevant for a node to discover an instance. Upon receiving a DIO, a node that wishes to join the instance sends back a Destination Advertisement Object (DAO) to the new parent and broadcasts a DIO (fig. 2.8b). The purpose of rank is illustrated on figure 2.8c: node C receives a DIO from nodes A and B. It requests the rank of both and decides to join the DODAG with node B as parent (as can be seen on fig. 2.8d). The process continues until all nodes that want to join do so. It is also possible for a node to request a DIO by broadcasting a DODAG Information Solicitation (DIS) packet.

Many work has been achieved to extend the functionalities of routing protocols and improve their performance in different use cases. Adding support for mobility is very important for the IoMT.

Authors in [99] propose an enhancement to AODV [88] by taking into account the mobility of users. In [100], authors propose to extend RPL through the use of an extended Kalman Filter to better account for mobility. LoSeR0 [101] exploits the mobility pattern of users to decide whom should receive forwarded packets.

Subsections 3.3, 4.3 and 5.3 in chapters 3, 4 and 5 respectively will cover more in details the aspect of mobility specifically for the contributions detailed in these chapters.

2.5 ML for efficient data transportation in mobile networks

ML is a subset of Artificial Intelligence (AI) [4]. A ML algorithm uses one or several characteristics called *features* to either classify or make prediction on data. The goal of classification is to deduce the type or label of data given its features. For instance, to differentiate a car from a pedestrian, one could use features such as velocity and the position of the user. If the velocity is high (e.g. 100 km/h), it is unlikely the user is a pedestrian. On the other hand, if the user is located in the middle of a forest far from any roads, it is likely to be a pedestrian.

Prediction is useful in anticipating the future state of a system or device. This unlocks new possibilities like deducing the number of devices in a location to deploy enough resources to service all of them. This is achievable, for instance, by predicting the future position of devices according to their last known positions. Prediction can also be used to anticipate when a piece of equipment will break to

schedule maintenance before it happens [102] [103]. Another example is to find security breaches on a computer system or network. First, users' behaviors that are considered as "normal" are recorded. It is then possible to deduce when a user is acting abnormally and a security breach is imminent [104] [105].

An algorithm can be trained using *supervised learning*: it is fed with data and the associated labels. The labels are the nature of the data (for classification) or the next expected value (for prediction). The algorithm can then establish the relationship between the values of the input data and the resulting label (desired output). Once trained, the algorithm can be tested by feeding it with data without the labels, then a comparison is made between the labels returned by the algorithm and the real labels of the data. The resulting percentage of correctness allows to assess how accurate the algorithm is.

It is worth noting that it can be tough to make some predictions, especially if the wrong input data is used (or if not enough data is considered). For instance, let us suppose we want to predict the future values of delay in a network. This could be useful to anticipate peaks of transmissions to minimize interference or maybe to wait some time before sending data that is not time-critical. However, using only delay values through time to make a prediction might be irrelevant and yield very poor results. This is because the value of delay is impacted by other factors. If for instance the delay has been increasing in the last few seconds, there is no guarantee it will not suddenly drop (or increase even more, for that matter) because the increasing could be the result of simultaneous forwarding of data from sensors sending measurements at the very same time. In that case, the peak will last for a short time and things should return to normal promptly. On the other hand, it is possible the increasing in delay is due to congestion because more data is generated on the topology compared to what the network can forward/transmit. We can see in this case that predicting a metric such as delay can be complicated and might require the consideration of other metrics, including ones that are not related to the network, like the time of the day (if sensors transmit one value each hour, for instance).

ML is thus very interesting for dynamic networks and has already been investigated by several researchers to use in wireless networks [106]. Classification and prediction can be useful in improving the deployment of relays by knowing which users are the most stable and where users are heading to anticipate problems related to congestion and interference (deploying more relays or allowing more channels). The contribution depicted in chapter 4 uses classification and the one detailed in chapter 5 uses prediction.

2.6 Conclusion

In this chapter, we covered different kinds of wireless technologies and architectures. We saw that a dedicated static infrastructure is not well adapted to mobile networks such as the IoMT and ad hoc networks. Then, we discussed different methods used to bring better service to end users such as virtualization and decentralization that unlock the concept of edge computing. After, we argued why RPL is a powerful and flexible routing protocol adapted to mobile networks. State of the art solutions to account for mobility lack a global approach that also considers the needs of users and the nature of the device. We finally introduced the concept of ML and discussed

how it can be helpful in deducing and anticipating the needs of the topology. The next three chapters will cover the contributions of this thesis.

Chapter 3

Programmable OF for improving QoS in mobile networks

Résumé français

Ce chapitre se concentre sur la problématique d'assurer la qualité de service dans un contexte hétérogène du point de vue des besoins des appareils et utilisateurs ainsi que de leur type de mobilité. Le protocole RPL peut, sur un même ensemble de noeuds physiques, créer plusieurs instances (virtuelles) simultanément. Ceci permet de répondre à divers besoins en fonction du contexte et des besoins des appareils connectés. Cependant, si trop peu d'appareils ont un besoin spécifique il est possible que certains se voient exclus d'une instance car trop peu contribuent à cette instance. Les solutions de l'état de l'art ont souvent des limitations comme imposer une vitesse nulle ou faible à l'ensemble des noeuds ou elles considèrent que les conditions radio sont d'une très bonne qualité.

Dans ce chapitre, nous proposons d'abord une architecture inspirée du paradigme SDN. Cette architecture permet de créer des instances réseaux virtuelles lors de la création ou modification d'un besoin. Ceci peut être réalisé en temps réel aux endroits où ce besoin apparaît, change ou disparaît. Par exemple, une route sur laquelle circule des véhicules connectés peut devenir encombrée durant l'heure de pointe. L'augmentation du nombre de véhicules connectés (les noeuds du réseaux) diminuera la qualité de service et ce cas nécessite de mettre un place une nouvelle fonction objectif ou un nouveau point d'accès afin de fragmenter la zone d'intérêt en zones plus petites.

Ensuite, nous proposons la fonction objectif programmable pour RPL. Cette fonction tient compte de plusieurs métriques à la fois qui sont reliées par une équation où chaque métrique (délai, débit, énergie résiduelle, vitesse, etc.) est multipliée à un poids. Le poids dépend du besoin : si l'économie d'énergie est importante, le poids sur l'énergie résiduelle sera élevé par rapport aux autres métriques. Tenir compte de plusieurs métriques en même temps permet une adaptation dynamique du réseau, puisque chaque changement (du délai, de l'énergie, ...) va affecter la qualité des chemins et les liens entre noeuds pourront être recalculés et changés dans le temps. Nous intégrons également la vitesse relative en tant que métrique. En effet, si deux noeuds sont à l'arrêt ou se déplacent suivant le même vecteur-vitesse, on peut espérer que la qualité du lien sera stable. Cependant, si deux noeuds voyagent à vitesse égale mais opposée, leur qualité de lien sera très variable et il peut être souhaitable qu'ils ne se connectent pas directement l'un à

l'autre. L'évaluation de performance à été réalisée avec le simulateur OMNeT++. La solution proposée permet surtout d'améliorer le taux de délivrance des paquets en présence de mobilité et de diminuer la consommation énergétique, augmentant ainsi la durée de vie du réseau.

3.1 In a nutshell

This chapter focuses on addressing QoS issues related to heterogeneous needs of users and devices and their mobility. As discussed in previous chapters, it is important to consider several aspects of devices in a mobile network. Computing the best path to route data through a mesh topology by only taking into account network-oriented metrics is not enough to guarantee good QoS. Indeed, the notion of best path is complex and it is crucial to properly define it according to the current context. In this chapter, we propose a solution that is not only based on network metrics like ETX or link delay but also on system-oriented metrics like residual energy of nodes and application-oriented metrics such as the requirement of a low end-to-end delay (as explained in subsection 1.1.5). The chapter is centered around the following items:

- Proposition of a new architecture inspired from a Software Defined Network (SDN) approach [3] [79] [80];
- Creation of a new Programmable Objective Function which uses fuzzy logic to take into account several metrics at the same time;
- Consideration of the mobility of users by introducing the concept of "obsolete parent".

3.2 Context

In order to ensure QoS in wireless networks, metrics such as delay, energy consumption or mobility must be respected. For instance, to avoid an accident on the road the delay must be very short with a very low jitter. In the case of IoV, it is possible to gather several different types of data from vehicles. Indeed, cars, trucks, buses and also bicycles and scooters (e.g. in a bike/scooter-sharing service) are equipped with sensors to determine position, speed, acceleration and so on. This data can be used to optimize the device's network performance by making decisions such as which neighbor to connect to. The data from many devices can be analyzed to understand their general behavior or the environment in which they evolve. For instance, rush hour can be determined to happen during a specific span of time in the day and the most jammed (in terms of traffic on the road but also the overloading of the network) sections of roads or areas can be found. Vehicles can then be redirected to alternate paths.

RPL [11], as explained in section 2.4, allows for simple routing in a many-to-one configuration, as nodes (internal nodes or leaves) only need to send packets upwards in the tree. The concept of rank simplifies the connection of new nodes on the topology. RPL can also send traffic downwards. To do so, one method is for any non-leaf node to store a list of all nodes that are its descendants. Upon reading a packet's destination, the current node searches its routing table (the list of nodes)

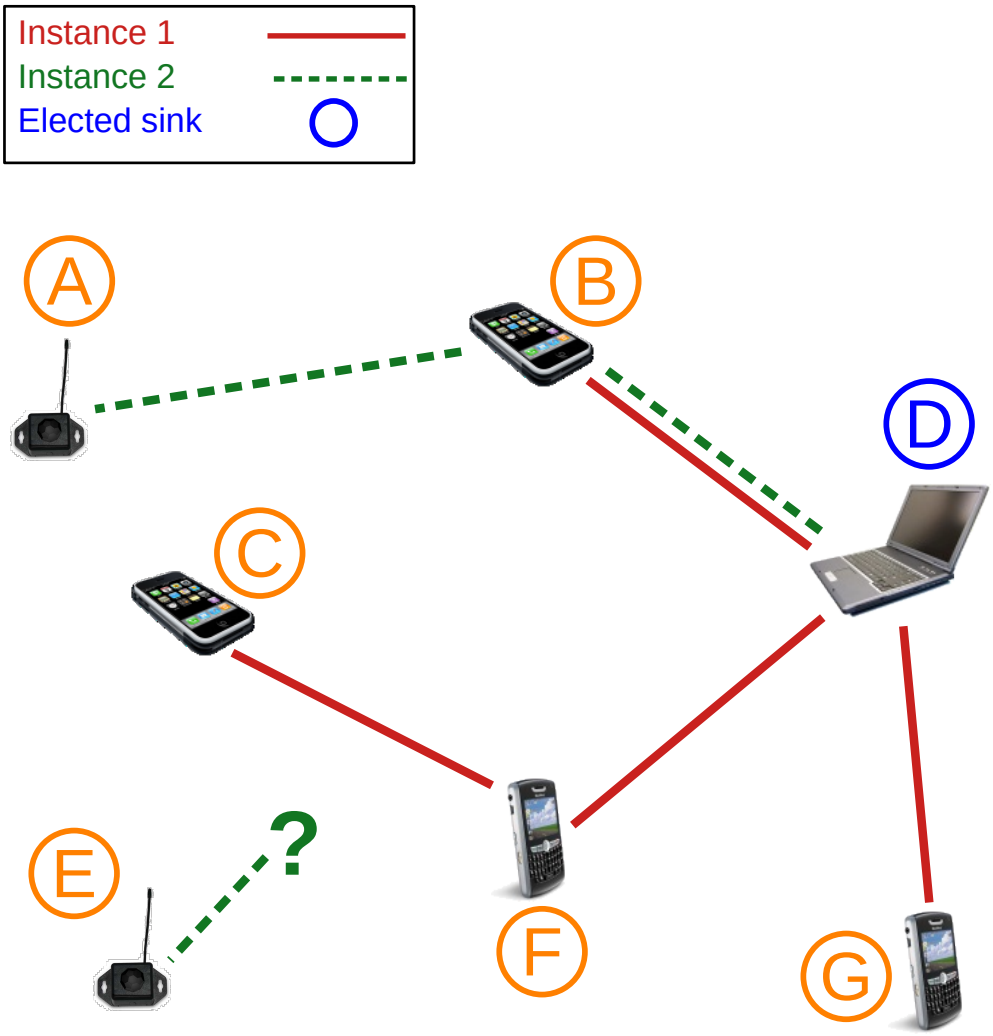


Figure 3.1: An example of a topology with one sink and two instances. Sensor node E is not in range of any device that joined green instance 2.

and if the destination is in it, the packet is sent downwards to the next hop. If not, the packet is sent upwards to the parent.

RPL can take into account several metrics [92] to answer multiple needs. Considering only one metric might result in ignoring other needs (e.g. if only delay is considered, energy consumption will be ignored). Of course, the concept of instance allows to have different OFs to meet different needs [90] [91]. However, if it is not possible to foresee the needs of devices and if there are many needs it can be very tricky and sub-optimal to define a lot of different OFs and generate as much different instances. Nodes are not requested to join any instance and if a specific need is shared by only few nodes, it is very likely that they will be too far apart to connect together to form a DODAG and they will not be satisfied. Figure 3.1 depicts such an example on a topology with two instances and several nodes such that one node (sensor E in the figure) does not have access to an instance that would satisfy its need.

3.3 Related work

Related work is presented and classified in the three following categories: general approaches non-specific to RPL (sec. 3.3.1), enhancements to improve routing and topology building in RPL (sec. 3.3.2) and RPL-specific modifications to support mobility (sec. 3.3.3).

3.3.1 General approaches

In [107], the authors propose a solution called Kalman Positioning-RPL (KP-RPL). Static nodes connect to one another using ETX and the link quality metric. Kalman filtering is used on each mobile node to estimate its own future positions. Each node then creates a blacklist of nodes that will not be in radio range so they do not attempt to connect to them. Although several nodes are mobile, it is assumed there are anchor nodes that are static. Furthermore, mobile nodes do not exceed a speed of about 2 m/s.

Bouaziz et al. [100] [108] use an extended Kalman filter as well but also consider non-linear trajectories (compared to [107]). The direction taken by users is considered to better predict their future position. However, the performance evaluation of these algorithms uses low mobility (about 2 m/s) which is not well-suited for the IoV. They consider several static nodes are available, which is not necessarily the case in a vehicular network.

Chang et al. [109] propose a solution based on ML and a genetic algorithm to improve several aspects of WSNs. The ML part of this solution uses K-means clustering to find the best cluster heads and prevents overloading them. A multi-objective optimization model is used to improve network lifetime, connectivity and reliability. A genetic algorithm is used to lower the complexity of the problem. However, the proposed scheme is evaluated using static sensors.

3.3.2 Routing and topology enhancements for RPL

BRPL [110] extends RPL by adding two main features called QuickTheta and QuickBeta. QuickTheta is used to load-balance traffic on the network by taking into account the queue length of nodes. QuickBeta is used to monitor the change of state (from online to offline and vice-versa) of nodes. A node is seen as mobile if it changes state often. The load balancing aspect is interesting and works well. However, the detection of mobility does not consider the movement of nodes explicitly. A node that wants to save as much energy as possible might be considered mobile even if it is not. Experiments were also realized indoor with low mobility (i.e. walking speed).

Khallef et al. [111] have introduced the Non-Linear Objective Function (NLOF). In NLOF, all available metrics are first normalized to obtain values on the same range. Then, they are compared and the worst one is selected. This solution allows to focus the building of the topology according to the most critical aspect of the network. Indeed, if delay, for instance, is very high on the network, it will be used as a metric to improve this aspect. However, if one metric is particularly bad, the algorithm will tend to focus only on this one, ignoring the others. If for instance the ETX is high enough, energy will be ignored and a node might have its battery drained quickly.

This work [112] studies several OFs in order to improve performance on a multi-gateway network in LLNs. The base principle is to rely solely on the hop count to find the shortest route as long as one and only one such shortest path exists. If several paths are possible, tie-breaking metrics are used. One class of OFs uses a greedy approach and the potential parent advertising the best value will be selected. The other class of OFs uses end-to-end values of the tie-breaking metrics if several paths have the same number of hops to the destination. In both classes, if all corresponding metrics turn out to have the same value, the parent will be selected randomly. Selecting the path according to the number of hops is simple and overlooking the tie-breaking metrics in the case of an identical number of hops allows to improve performance. However, there are two main disadvantages to this solution. If the number of hops of all potential parents is always different, a node will only consider this metric and will not be able to take other aspects of the network into consideration. The other main drawback is the case where all corresponding metric values are identical: the parent will randomly be selected so one node could end up being selected by many and it could have its battery drained faster or undergo heavy congestion.

This [113] proposition uses queue utilization to load balance packets in the network in order to improve Packet Delivery Ratio (PDR). According to the authors, monitoring queue utilization allows to detect packet loss due to congestion earlier compared to using ETX. Information on congestion can be transmitted to neighbors in a DIO packet and influences the value of rank.

3.3.3 Mobility support for RPL

Authors in [114] propose to adapt RPL by forcing mobile nodes to be leaves, allowing only static users to act as intermediate nodes. This stabilizes the topology because less disconnections occur due to mobility. The authors also introduce the concept of reverse trickle timer. The principle is to send signaling more and more often because as time passes, the topology changes due to mobile nodes and disconnections will occur with a growing probability. In ME-RPL [115], mobile nodes are flagged and other nodes try to avoid them as potential parent. The rate of signaling is also determined by the velocity of nodes. These solutions tend to generate more traffic and in the case of bad radio conditions, the network can quickly become overloaded. They also avoid using mobile nodes as potential parents which can be problematic in a vehicular network where all nodes are mobile.

In [116], the authors evaluated RPL under mobility and have modified it to improve performance in a VANET setting. Mainly, upon selecting a preferred parent, a node sends a DIO right away to its neighbors instead of waiting on the trickle timer to expire. When a node selects its parent, it sends a DAO to its children immediately as well. The authors also tested an OF close to MRHOF which uses latency instead of ETX. The performance evaluation is done by placing vehicles driving in a linear fashion (similar to a straight road) with a base station in the middle. The simulations were done using up to thirty nodes. However, because of the linear motion of nodes, they always have several choices for a preferred parent and this is not necessarily realistic.

In [117], the authors combine the quality of links between nodes and their mobility using a machine learning approach called AQ-routing. The model learns by being rewarded when a packet reaches its correct destination. The performance

evaluation is done with only one source and the energy consumption is not considered.

Tian et al. [118] propose to limit the depth of topologies to limit the number of hops. They also modify the rate at which signaling is sent according to the velocity of nodes: the faster a node moves, the shorter is the period to send topology-oriented information.

3.4 Motivation

Most of the solutions presented in the last section do not account for mobility or make strong assumptions like having several static nodes, considering low mobility only (i.e. pedestrian walking speed) or having good radio conditions such that sending more signaling would scale up without problem.

It is worth mentioning the velocity of nodes is not part of the OF in the aforementioned works although, when considered, mobility does indeed impact the choice of preferred parent. Some propositions also consider only one or few sources (nodes generating data) or focus on one metric to improve one aspect of the network.

In the IoV, users have a high velocity. This causes devices to constantly enter in and out of range of one another and results in constant connections and disconnections. The topology is thus very unstable as direct links (between two devices) and, incidentally, paths (from a source to a destination which can span for several hops) are always changing. Many transmissions will fail and re-transmissions will occur often, increasing delay and jitter, energy consumption and lowering effective throughput. QoS will drop accordingly.

As we will see in the remainder of this chapter, it is possible to enhance RPL to consider several metrics at the same time, including mobility, and to modify it in order to impact the rank to reflect the fact information from parent is obsolete. This effectively allows RPL to consider several needs at the same time while taking into account mobility.

3.5 Proposed architecture

This section introduces the novel SDN-inspired architecture which is proposed to deal with the high mobility of users in the IoV. As explained in chapter 2, in a SDN the data server is dissociated from the control server. The latter can dynamically create network instances as needed. This architecture allows to modify the virtual network instances without the need to change the physical network. It is thus a very interesting paradigm to use in dynamic networks with a high rate of mobility such as the IoV.

As shown in figure 3.2, the proposition consists of a control server, a data server and several sub-networks[1]. The control and data servers, along with Internet, form the *cloud* part of the architecture. The sub-networks make up the *fog* part of the architecture and are connected to Internet as well. The controller has a global view of all sub-networks and will take macro decisions. For instance, creating a new

[1]for the sake of clarity, the term *sub-network* will be used to design the smaller networks surrounded by blue ellipses in figure 3.2. They are the VANETS.

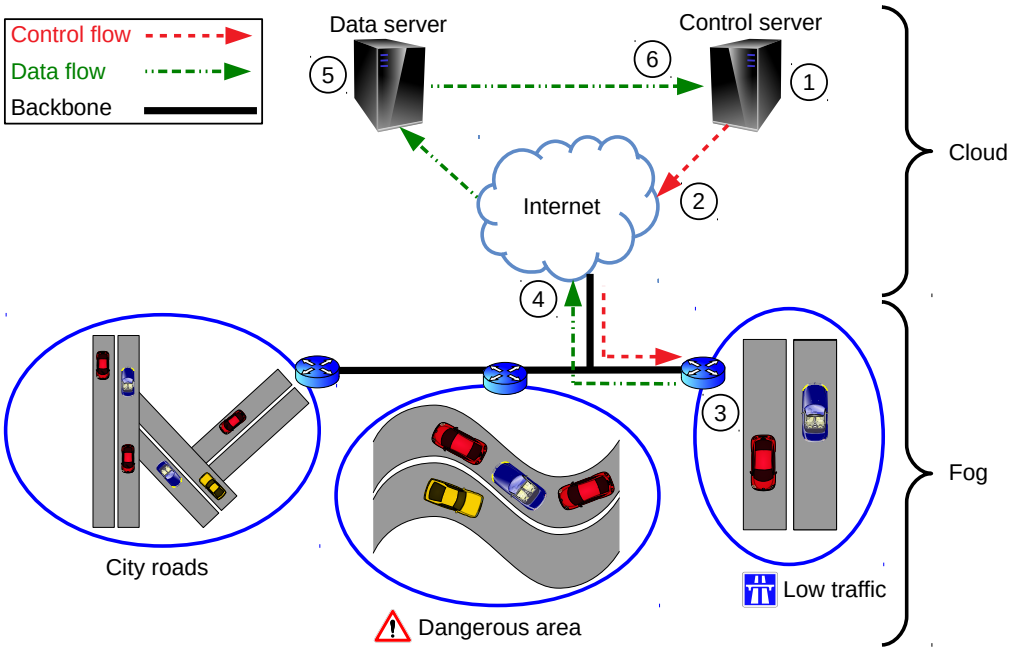


Figure 3.2: The proposed architecture, inspired from a SDN approach.

sub-network when a new need arises, splitting an existing sub-network into two or more if too many users causes QoS to decrease (that is, allocating more resources to better serve them) or changing the communication protocol between the different sub-networks. These decisions are depicted as the downwards "Control flow" arrows in figure 3.2. The data server retrieves aggregated data from nodes in sub-networks, which the controller can then use to adapt the global strategy (upwards "Data flow" arrows in figure 3.2). The control server does not have a local view on sub-networks. It is the role of the APs, or sinks, to manage each one of them.

Several Objective Functions can be designed according to the needs of the sub-networks (see section 3.6). In Figure 3.2 for instance, three different areas are shown: a set of city roads (lower speed, more traffic density), an area with a high rate of accident considered dangerous and a highway (lower traffic, high-mobility vehicles). All of these situations require a different approach when building the topology. For example, the dangerous area requires low delay and high resilience to make sure vehicles are made aware of an accident that is happening as soon as possible. Yet, even on one sub-network the needs might evolve. In the case of the highway, there can be mainly vehicles traveling long distances where passengers watch multimedia streams, so a focus on higher throughput and low delay will be desirable to ensure QoS. However, an exceptional event such as an accident can cause traffic jams: there will be a high density of vehicles driving slowly. In this situation, high interference can be a main concern and the OF might focus on ETX to avoid congestion due to re-transmissions. We will now go through an example run of the solution (see the numbered steps on figure 3.2).

- Step 1: The control server defines the global strategy, how many sub-networks are initially needed and which Programmable Objective Function (POF) (see sec. 3.6) to use for each;

- Step 2: The controller dispatches these decisions downwards to the *fog* part;
- Step 3: Sub-networks are created. Sinks are free to adapt the POF to local needs. More sinks can be created to help manage the sub-network;
- Step 4: aggregated data from the *fog* part is sent upwards to the data server;
- Step 5: The data server processes this information for statistical purposes (for instance, to understand how roads are used to plan maintenance);
- Step 6: The data server can then feedback the control server with the data from steps 4 and 5 to allow it to adapt the global strategy. The adapted strategy is sent downwards (step 1) and the process continues.

Sinks managing sub-networks in the *fog* part communicate amongst themselves and to the servers in the *cloud* part. Several control and data servers can be used for redundancy in case of a failure and default communication protocols are also supposed to be agreed upon so that sinks can always communicate even if the backbone cannot connect to Internet. A sub-network can also be created if no connection exists to the controller. In that case, the information about the newly created network will eventually be sent upwards (step 4) and the controller will be made aware of it.

3.6 Programmable objective function

Now that the architecture has been defined, it is time to explain the proposed OF. As mentioned in section 3.5), it is possible to define several OFs to specific local needs. Here, we will focus on one such instance. The general idea is to overlook all available metrics and take a decision based on all of them (using fuzzy logic). The result will vary based on the value of each metric to prevent any of them to reach a critical level. In any case, all metrics will always be taken into account to avoid having one metric dominating all the others. This fuzzification process allows the algorithm to finely adapt to a dynamic environment no matter what metrics are available.

As mentioned in section 3.3, several solutions exist to approach the problem of routing in the IoV. One possible way to regroup these algorithms is according to the number of considered metrics. The algorithms can focus on:

- using one metric only;
- comparing several metrics but using only one in the end;
- combining several metrics.

Focusing on one metric is useful to reach one specific goal like energy saving or minimum delay. However, looking at only one metric might not be the most efficient way to improve performance of the whole network and different nodes might have different needs. Furthermore, if considering only one metric yields to more than one possible path it could be complicated to decide which path to choose from.

To resolve these issues, one solution is to look at several metrics to select the one that will be used in the end. As only one metric is ultimately used to find the best path, the optimization problem remains simple in terms of complexity and having a look at all metrics allows to take a better decision.

Finally, it is possible to overlook all metrics and take all of them into account. The optimization problem becomes more complex but enables maximum control over the path to select by choosing for any metric which one is to be considered and to what extent. It is possible to focus on energy saving, low delay, high throughput, good reliability or to find a fair compromise amongst all of the them. For instance, if the loss rate is high, the priority could be on lowering the ETX to improve network performance. In practice, this is done by combining metrics in one equation (eq. 3.6 below).

We consider a network N composed of l nodes (from 1 to l included). Each node possess a vector M_i $(0 < i \leq l)$ containing k available metrics (from 1 to k included):

$$M_i = [m_1, m_2, \ldots, m_k] \tag{3.1}$$

For each metric m_j (where $0 < j \leq k$), we use four bounds that will determine which metrics are considered and to what extent:

m_j^{min} : the lowest possible value of m_j;
m_j^{lo} : the lower bound to start considering m_j;
m_j^{hi} : the higher bound before considering only m_j;
m_j^{max} : the highest possible value of m_j.

The lowest possible value of a metric (m_j^{min}) is its minimum physical value. For instance, the lowest delay is 0 ms. The lower and higher bounds (m_j^{lo} and m_j^{hi}, respectively) are defined according to specific needs. The former determines from which value the current metric will start to impact the choice of parent. The latter is the threshold above which the result is impacted only by this metric. Finally, the highest possible value (m_j^{max}) is the maximum physical value of a metric. To continue with the example of delay, this would be $+\infty$.

We define a set of Π functions f^π $(0 < \pi \leq \Pi)$ that are used to combine the different metrics: one function (POF) is implemented for each target strategy π (energy saving in a low density network, minimizing re-transmissions in a high-mobility network, ensuring maximum reliability on yet another network, etc.). We also define the weight of each metric w_j according to their value:

$$w_j = \begin{cases} 0 & \text{if } m_j < m_j^{lo} \\ 1 & \text{if } m_j^{lo} \leq m_j \leq m_j^{hi} \\ +\infty & \text{if } m_j^{hi} < m_j \end{cases} \tag{3.2}$$

The weight is the impact the current metric will have on the choice of parent. It is set to zero if the value of the metric is too low, to one if the value is between the lower and higher bounds and to plus infinity if it is too high. In order to combine the metrics, one must first normalize them to make sure they are in the same range

of values. For this purpose, we define the vector Γ:

$$\Gamma = [\gamma_1, \gamma_2, \ldots, \gamma_k] \tag{3.3}$$

Note that the length of Γ vector is the same as the length of the M_i vectors from equation 3.1. Also, there is only one Γ vector for all nodes because the normalization is the same no matter which node is considered. The normalization can be tricky because the minimum and maximum values of the metric to normalize should be known in advance in order to determine by how much to multiply or divide the metric to normalize it. The minimum and maximum values correspond to the aforementioned bounds (m_j^{min}, m_j^{lo}, m_j^{hi} and m_j^{max}). However, in the case of a metric having no upper limit (such as the delay), a maximum value must be selected. This is achievable by choosing a finite upper limit. In the case of delay, 1000 ms is a fair choice, as many applications consider connection is lost if delay is higher than 1 second. Furthermore, the high mobility in the IoV causes a long delay to be even more impactful. It is also possible to run several simulations to deduce the range of each metric and set the maximum value accordingly. Each metric will then be normalized (equation 3.3) and multiplied by its weight (equation 3.2):

$$\lambda_j = \left(\frac{m_j}{\gamma_j}\right) \cdot w_j \tag{3.4}$$

to produce one impact vector Λ_i per neighbor:

$$\Lambda_i = [\lambda_1, \lambda_2, \ldots, \lambda_j], \quad i \neq \psi \tag{3.5}$$

where ψ is the current node (a node never considers itself as a potential parent). The impact vector is fed to the current POF (f^π) to compute the score of each potential parent in order to sort them. S_i represents the score of neighbor i that the current node is computing:

$$S_i = f^\pi(\Lambda_i), \quad i \neq \psi \tag{3.6}$$

A low score is preferable to a high one. Any node ψ will have to compare the score of all neighbors and select the one with the lowest score to determine its best parent P_ψ:

$$P_\psi = \min_{i \neq \psi} (S_i), \quad i \in N \tag{3.7}$$

P_ψ is thus the best parent for node ψ. Note that the velocity of nodes is included as a new metric in the POFs [92].

To illustrate the concept of programmable objective functions, let us consider two different metrics: ETX (m_1) and delay (m_2). As shown in figure 3.3, we might want to double the value of ETX to create a topology with good link quality and because the value is discrete, it is represented as a step function (the red-continuous line in figure 3.3). We might also want a high delay to have a big impact on the choice of parent, by squaring its value (in figure 3.3, it is the blue-dotted line). Of course, it is possible to have more complex functions to combine several metrics (including velocity) in different ways.

The implementation of the parent selection process is shown in algorithm 1. Each node on a sub-network will run this algorithm. Lines 1 and 2 are the initialization steps. The loop from line 3 to line 20 is to iterate over all nodes. A node is only considered if it is in range (line 4). Line 5 is the selection of metrics for the current candidate and line 6 is the initialization of the impact vector. The loop from line 7 to line 15 is to iterate over all metrics. Line 8 to 13 set the impact of each metric as explained in equations 3.4 and 3.5. Line 16 is the comparison of ranks to select the best parent (equation 3.6 and 3.7), which is then returned (line 21).

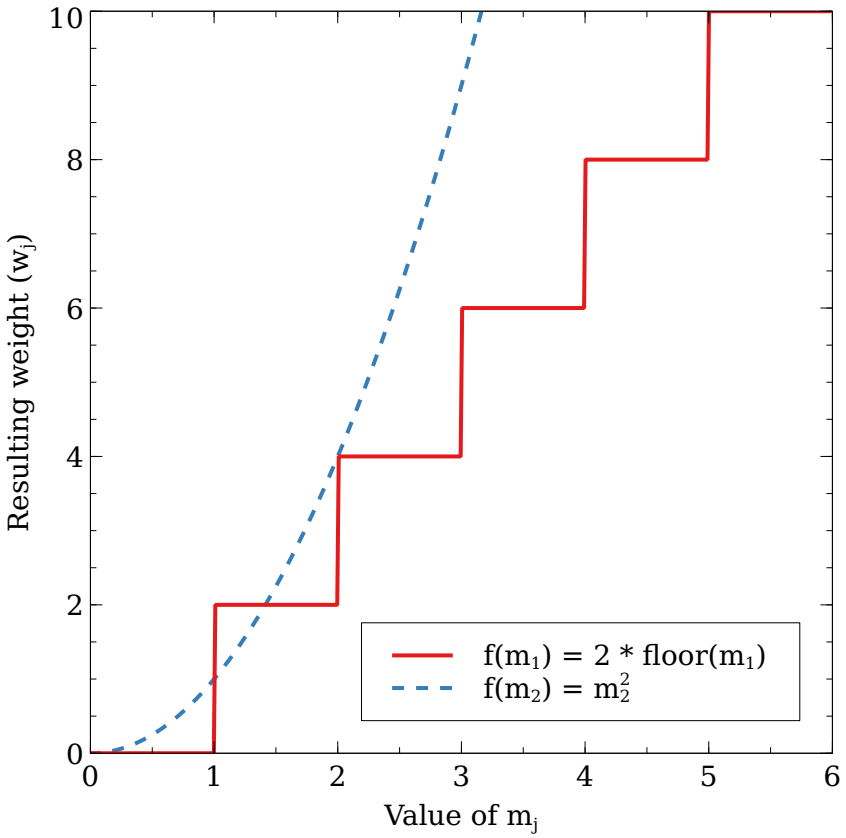


Figure 3.3: An example of two programmable functions. The horizontal axis is the value of the metric and the resulting output is on the vertical axis.

Algorithm 1 The parent selection process.

$preferredParent \leftarrow null$
$currentRank \leftarrow +\infty$
for $i = 0$ to $n - 1$ **do**
  if $node_i \neq this$ && $isInRange(node_i)$ **then**
    $m \leftarrow M_i$
    $\Lambda_i \leftarrow null$
    for $j = 0$ to $k - 1$ **do**
      if $m_j > m_j^{hi}$ **then**
        $\Lambda_i.append(+\infty)$
      else if $m_j^{lo} \leq m_j$ && $m_j \leq m_j^{hi}$ **then**
        $\Lambda_i.append(1 * m_j/\gamma_j)$
      else
        $\Lambda_i.append(0)$
      end if
    end for
    if $f^{\pi}(\Lambda_i) < currentRank$ **then**
      $preferredParent \leftarrow node_i$
    end if
  end if
end for

return $preferredParent$

3.7 Relative velocity and obsolete parent

In order to better account for mobility, the use of the aforementioned model is not enough. Indeed, the fine-tuned bounds and programmable OFs allow more control but one more aspect must be considered to strengthen the model: the obsolescence of rank due to the velocity of nodes. The basic idea is that the faster a node moves relative to its parent, the faster it will leave the connection range of this parent. A node can easily know its own velocity using the GNSS, the rotational speed of wheels (e.g. the speed the driver reads in a car) or the airspeed (for airplanes or UAVs). Detecting the obsolescence of rank for any node is thus introduced here. It is accomplished simply by taking into account the relative velocity of node to its parent. To illustrate this, let us take five different cases:

1. The parent node and the child are static;
2. The parent node is static but the child node is mobile;
3. The parent node is moving but the child node is static;
4. Both nodes are traveling in the same direction and at the same velocity and
5. Both nodes are traveling in a nondescript direction and speed (excluding previous case).

If only the velocity of the child node (i.e. the current node) is taken into account, cases 1) and 2) can be resolved but not the other cases. Of course, if the parent node is mobile like in case 3), its rank will increase and the rank of the child might eventually be impacted. However, if the connection between the parent node and the child node is too bad due to the parent's velocity, the child node might never receive the updated information.

Case 5) will have somewhat the same issue: even though the child node is mobile, its parent's velocity will not be accounted for and the increase in rank will not reflect the real mobility of both nodes. Case 4) is a special case where both nodes travel but keep close to one another. This is the case of a convoy where at least one vehicle follows another one. This could happen if the concerned vehicles know each other and drive to the same destination following the same path. Another case would be that two random vehicles happen to go in the same direction at the same speed for a certain amount of time. In that specific case, only considering the velocity of the child node might lead to a high increase in rank if one message was lost because the child node considers it quickly moves away from the parent node. This phenomenon will be amplified if more than two vehicles are in the convoy and the ETX is more than 1.

This is why considering the relative speed is more reliable: it allows to transform case 4) to case 1) (relative speed is zero) and to transform cases 5) and 3) to case 2) (relative speed is strictly higher than zero). Indeed, the faster a node travels, the faster it will move out of range of the parent it is connected to. Also, because a node computes the obsolescence of rank using only information available to itself, no network overhead is generated. That is, no matter the radio conditions of the current network, there will not be more transmissions required to take the speed factor into account. In the case where a new parent needs to be selected, though, more signaling will be needed (but all algorithms will be impacted the same way).

Of course, the speed vector needs to be transmitted somewhat: this information is added to signaling packets as a new metric [92].

To summarize, a node increases its rank regularly if it does not receive traffic from its parent. Once the parent is heard from, the rank is computed using the current POF. This reflects the obsolescence of rank and the children of a mobile node will end up changing parent if the last transmission is too old. This mechanism proves efficient as depicted in section 3.8.3.

3.8 Performance evaluation

In this section, simulation results are shown. First, PIs (see subsection 1.1.5 for a definition of PI) that will be depicted in all simulations are explained. Then, two sets of simulations are presented: the first one (sec. 3.8.2) concerns results with static nodes only. These scenarios allow to see the impact of combining metrics in a POF without mobility to affect results. ContikiOS [119] was used to obtain these results. It is an embedded operating system that can be uploaded on sensor nodes. The second set of simulations (sec. 3.8.3) introduces mobility and presents results from OMNeT simulations.

3.8.1 Studied performance indices

In this section, PIs used to compare the different solutions are explained. For the sake of clarity, results are regrouped by the following categories: packet delivery ratio, packet loss (dynamic scenarios only), end-to-end delay, throughput usage, energy consumption and network lifetime. Results for static scenarios are shown in section 3.8.2.2 and the dynamic cases are discussed in section 3.8.3.2.

3.8.1.1 Packet delivery ratio

The amount of delivered packets is an important aspect in a network. Packet Delivery Ratio (PDR) shows how many packets reached the destination, in percentage, compared to the number sent. More precisely, the PDR is the amount of packets that have reached the final destination. This is in contrast with the amount of packet loss (see 3.8.1.2), which is an absolute value.

PDR is an important metric to evaluate QoS. If PDR is low, users might not be satisfied and it is possible many re-transmissions occur, ultimately leading to a packet being discarded (maximum of re-transmissions allowed has been reached). However, PDR is not enough to account for the quality of links. Suppose an algorithm is such that nodes are rarely connected. If these nodes manage to send a few packets and all of these packets reach the destination, PDR will be 100%. However, the total amount of packets sent is extremely low and it might be better to have an algorithm with a slightly lower PDR that sends way more packets. The amount of packet loss, combined with PDR, will help to see a better picture. PDR is shown for static (sec. 3.8.2.2) and dynamic (sec. 3.8.3.2) cases.

3.8.1.2 Packet loss

Whenever a node needs to send traffic, it might not do so if it has no connection. A node that is disconnected from the topology will not generate data packets even if it is a source. The amount of packet loss takes into account the number of packets

that should have been sent but were not. It is a less accurate measure compared to PDR because it does not differentiate packets that were lost due to bad radio conditions or due to lack of connectivity, but gives an absolute value.

In a mobile environment like the IoV, it is important to record the connectivity of users: if users are rarely connected at all QoS cannot be achieved, hence the usefulness of the amount of packet loss. However, once we know packets have reached their destination, it is important to understand if they arrived fast or not: this is why we studied the end-to-end delay as well. Packet loss is only shown in dynamic scenarios (sec. 3.8.3.2).

3.8.1.3 End-to-end delay

The end-to-end delay of a packet is the time it takes to travel on the network. A long delay can be caused by several factors: many re-transmissions (high ETX), congestion due to lack of load balancing or low throughput. A long path from source to destination increases the chance of going through a congested node or using a bad quality link. A low delay might reflect a lightly loaded network (even with some re-transmissions the delay can stay acceptable) or good radio conditions in a moderately- or heavy-loaded network.

Several applications are time-critical, such as communication in a Vehicle-to-Vehicle (V2V) setting to propagate information about an accident that has just happened. However, the study of the end-to-end delay is not enough to ensure QoS. Some applications like watching multimedia streams (for instance, passengers on a vehicle watching a movie on a long trip) also require a high throughput. Note that the delay is only significant if the PDR is high enough. Indeed, if too few packets reach their destination for a given algorithm, the sample size can be very small and the resulting value will be inaccurate. Delay is presented in both static and dynamic cases (sec. 3.8.2.2 and 3.8.3.2 respectively).

3.8.1.4 Throughput usage

The usage of throughput, in percentage, represents how much radio links are used. High link usage can result from lack of load balancing or bad radio conditions resulting in many re-transmissions (including signaling packets as well). A link that is lightly loaded generally means less energy usage because the node spends less time transmitting.

In static scenarios (sec. 3.8.2.2), throughput usage depicts the usage of the destination's links (how much the destination node is receiving network traffic), whereas in the dynamic cases of section 3.8.3.2, it represents the average usage of all links (all nodes) on the topology.

3.8.1.5 Energy consumption

The amount of consumed energy can be crucial to consider in many cases. That is to save battery energy to extend node and network lifetime (electric cars), preserve the environment or to consume less electricity on a power grid, the energy consumption is more and more studied.

Although energy consumption is very useful, different nodes may have different amount of residual energy. In that case, it is relevant to also study the amount of

dead nodes. In static scenarios (sec. 3.8.2.2), energy consumption is presented as the maximum amount of energy consumed by one node (the node that consumed the most energy). In dynamic scenarios (sec. 3.8.3.2), it is the averaged energy consumption of all nodes.

3.8.1.6 Network lifetime

A node is considered dead when its residual energy is too low or its battery is completely depleted. The node will not be on the network any longer and the connectivity will get worse: the less nodes there are left alive, the more the remaining nodes will be solicited and have their battery drained faster. The network lifetime is the time taken before the first node runs out of battery power.

Load balancing network traffic, using less throughput or avoiding too much retransmissions are elements that help keep the nodes alive. This PI is only depicted in dynamic scenarios (sec. 3.8.3.2).

3.8.2 Static scenarios

This section shows results from simulations with static nodes only. These nodes are sensors. The objective is too evaluate the performance of the proposed OF without mobility compared to selected state of the art solutions. Simulations were done using the ContikiOS cooja emulator [120] on a Linux desktop computer. In this section, the OF is designated as "Fuzzy-based Objective Function" (F-OF) to differentiate it from POF in the next section (sec. 3.8.3.2) which considers velocity as well.

F-OF considers three metrics: hop count, ETX and energy remaining (or residual energy). However, given hop count is not as relevant as ETX to improve the performance of the network, it is not used (its weight is set to 0, see eq. 3.2 in sec. 3.6). All nodes are supposed to have the same battery capacity. If this was not the case, a more thorough definition of energy would be required. For instance, if two nodes are identical on all aspects except one has twice as much battery capacity as the other one, considering only the remaining power in percentage is not adapted because the node with more battery power could be more solicited due to it having more energy in storage. Also, metrics are additive so they are computed on the whole path and not only on the current node or link.

F-OF overlooks all metrics and uses the threshold as defined in section 3.6 to determine if a node has enough remaining energy. Three cases are possible (eq. 3.2):

- The current node has a battery level greater than the given threshold;
- The current node's battery is below the given threshold;
- The current node has an empty battery.

In the first case, the current node will not take its battery level into account. Using only ETX will yield to better performance in terms of delay, PDR and throughput by sacrificing battery power. The second case takes into account ETX and the remaining battery level of the current node to compute the rank. This case is very important because it will prevent a node with a very good ETX metric to be always

Parameter	Scenario 1	Scenario 2	Scenario 3
Topology size	150 m · 90 m	150 m · 90 m	150 m · 90 m
Simulation time (seconds)	3600	3600	3600
Traffic model (packets/s)	1	**1 to 6**	3 (on average)
Number of sources	**1 to 5**	5	5
Packet count (total)	3300	3300 to 16500	0 to 19800
Packet size (bytes)	16	16	16
Tx* range	30 m	30 m	30 m
Rx† rate (in %)	75	75	75
Batteries capacity (in joules)	1080 J	1080 J	1080 J

Table 3.1: Simulation parameters for the different scenarios. Bold text represents the studied parameter for each scenario. * Tx: transmission; † Rx: reception

solicited and have its battery drained rapidly. In the third case, it is not possible to use this node anymore. Normalizing the metrics can be tricky and one solution is to set the acceptable lower and upper bounds (m_i^{lo} and m_i^{hi} respectively) beforehand. The values of the bounds are discussed in section 3.6.

3.8.2.1 Simulation setup

All devices in the simulations are Wismote nodes as defined in the cooja emulator. The power consumption of all nodes is based upon the datasheets of the CC2520 radio transceiver [121] and MSP430 micro-controller [122]. Nodes are also considered to be powered by 2 AAA batteries (3.0 V) which allows to determine the total energy capacity. Table 3.1 summarizes the base parameters used for all simulations.

The effects of changing several of these parameters are studied, one at a time. The nodes have been placed on a 150 m per 90 m area (see fig. 3.4). As it is not possible to study every possible topologies, the chosen one has been built in order to cover several different cases. It is purposely not symmetric so different paths are possible from any source to the sink and choosing one or another is not obvious. For instance, one path is set to be the shortest in terms of hops for several sources but if all traffic is routed through this one congestion will occur at higher loads. Other longer paths exist so it could be better to select one of them instead if the former path is heavily congested. The simulation time is one hour to allow the system to stabilize, yielding to more accurate results. The sources are set to send 1 packet/s and the packet count (3300 packets) is set to be less than the simulation time (3600 s) to allow sources to have time to resend packets in case of bad transmissions. This 300 seconds extra time is long enough so that a non-significant amount of packets would reach the sink after this delay and will certainly be considered as timed-out by most applications. The traffic load ranges from 1 packet per second per source to 6 packets per second per source. The ETX has been changed by placing nodes closer or farther away from one another. The range between two neighbor nodes goes from 10 m to 30 m, the latter being the maximum transmission range with the worst ETX value. Bar charts (see below, sec. 3.8.2.2) show the various performance of the different OFs on the tested topology with a Variable Bit Rate (VBR). The VBR is such that a source will randomly send between 0 and 6 packets per second, so 9900 packets on average in total.

Given that the MSP430 and the CC2520 are designed for low power consumption, the running time of the simulations would have to be very long for the impact of

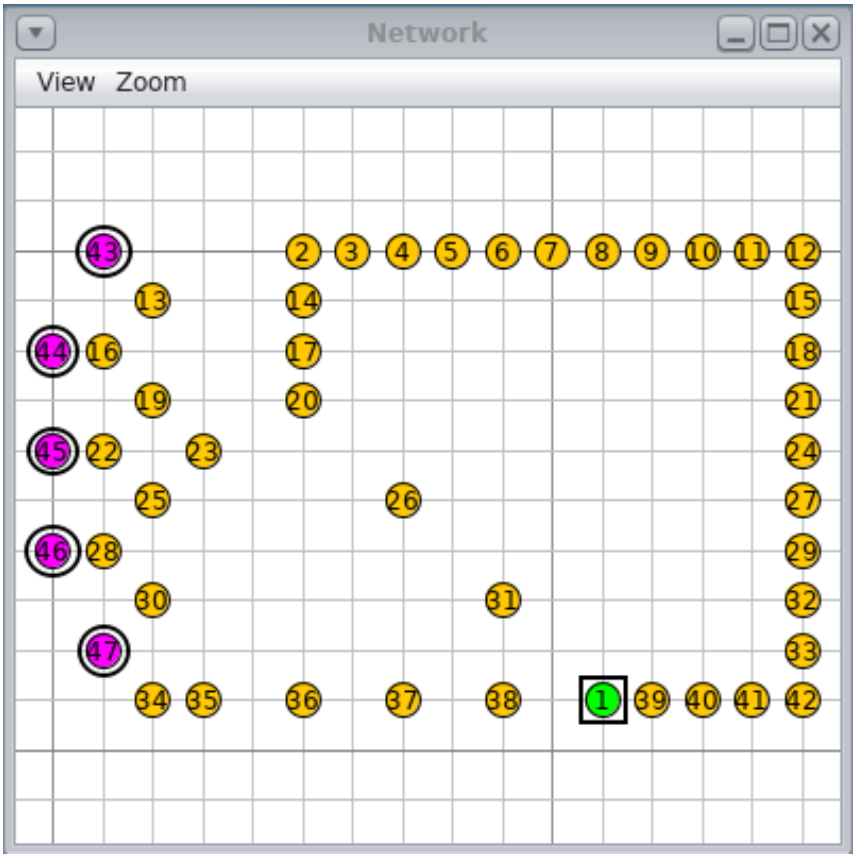


Figure 3.4: The studied topology. The rectangle surrounds the sink, ellipses surround the sources and the other nodes are intermediate sensors. One grid square is 10 m · 10 m in size.

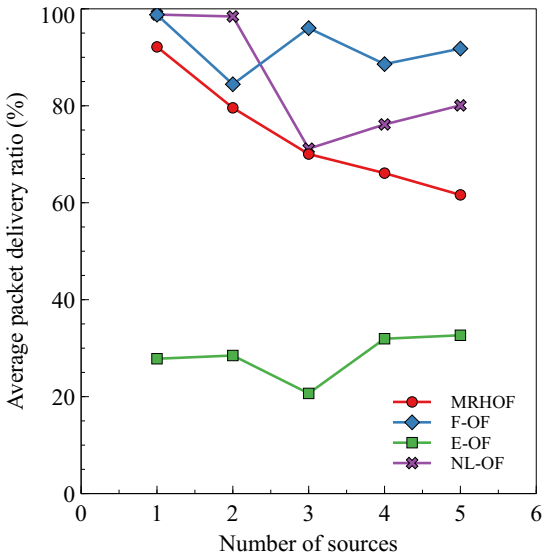
battery depletion to be visible. This is why the capacity of batteries has been divided by a factor 10 to cause a one hour simulation to drain a significant (more or less 20%) amount of all nodes total capacity in order to study the effect of energy-saving algorithms. In view of this static energy capacity, energy starts to be taken into account when the battery level is down by 15% ($m^{lo}_{energy} = 15$). Accordingly, m^{hi}_{energy} and m^{max}_{energy} are set to 100 (the battery is empty), at which point the rank is maximized. The values of ETX and energy can only be positive so their minimum value is set to 0. The ETX is always considered ($m^{min}_{etx} = m^{lo}_{etx}$) and is not constrained ($m^{hi}_{etx} = m^{max}_{etx}$).

The performance evaluation is achieved by comparing the different studied algorithms in terms of PDR, throughput and energy consumption. Given that the resulting PDR varies greatly between the different algorithms, the delay is not necessarily significant when comparing them to one another because of the different sampling sizes (i.e. varying number of packets reaching the destination), leading to inaccurate delay values in some cases. Four algorithms have been evaluated:

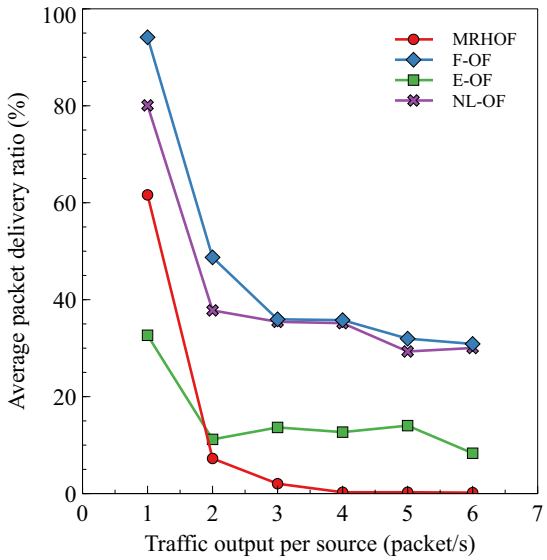
1. MRHOF is the OF as described in RFC 6719 [91]. It uses the squared ETX to influence the rank;

2. F-OF (POF with no mobility) is the proposed solution;

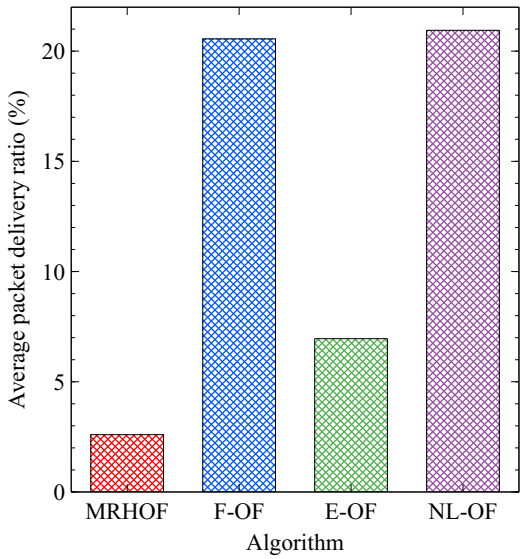
3. E-OF is the Energy Objective Function that bases its rank solely on the energy metric;



(a) The PDR against the number of sources.



(b) The PDR given different traffic densities.



(c) The packet delivery ratio of the different OFs.

Figure 3.5: The effect of changing either the number of sources or the traffic rate on PDR.

4. NL-OF is the Non-Linear Objective Function [111] (see section 3.3.2).

As results will show, E-OF has almost always the worst results[2] of all except in terms of energy where it performs the best. MRHOF mostly has average results. NL-OF and F-OF are comparable in terms of energy consumption and F-OF yields better results in terms of PDR and throughout.

3.8.2.2 Simulation results

[2]Note that this objective function is only intended to serve as a lower bound on energy consumption and is not studied to have the lowest possible energy consumption, nor is it intended to have good performance.

Packet Delivery Ratio Figure 3.5 plots the PDR of the four algorithms given the number of sources (fig. 3.5a) or the traffic density (fig. 3.5b). The packet delivery ratio tends to decrease as the number of sources increases because more packets are generated: when there is one source, 3300 packets are generated on the whole topology and when there are six sources, 19800 ($6 \cdot 3300$) packets are generated in total.

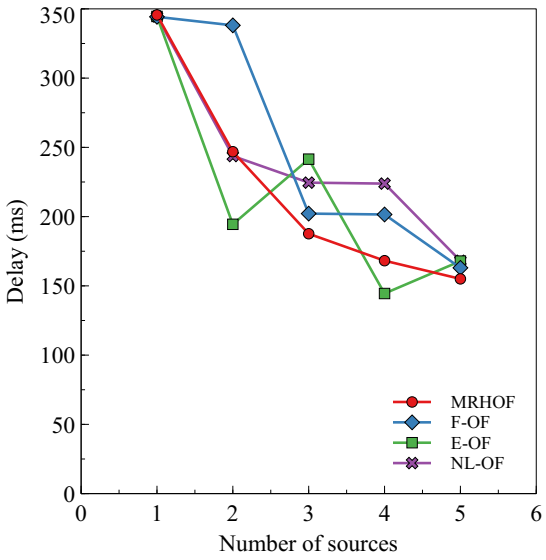
Adding sources one by one also modifies the topology and because of this, all functions have one point where the PDR drops significantly. The drop is seen on F-OF when there 2 sources and when there are 3 sources for the other functions. MRHOF is only slightly affected. MRHOF always only considers ETX which does not change through time, hence it always chooses similar paths when the number of sources change and the PDR drops with increasing congestion. E-OF, NL-OF and F-OF additionally rely on energy and/or the number of hops. The changing level of energy will lead certain nodes to select different parents through time. These parents will then have different additive metric values and their resulting rank will change. Thus, because of the dynamic nature of this selection process, E-OF, NL-OF and F-OF are more affected than MRHOF when adding sources.

In figure 3.5a, E-OF has the lowest PDR with values starting at 27.8% and up to 32.7%. MRHOF drops from 92.2% (1 source) down to 61.6% when there are 5 sources. NL-OF and F-OF both start at 98.8% of PDR when there is 1 source and end up at 80.0% and 91.8% respectively. Thus, F-OF is able to perform better by approximately 30% and 10% compared to MRHOF and NL-OF respectively.

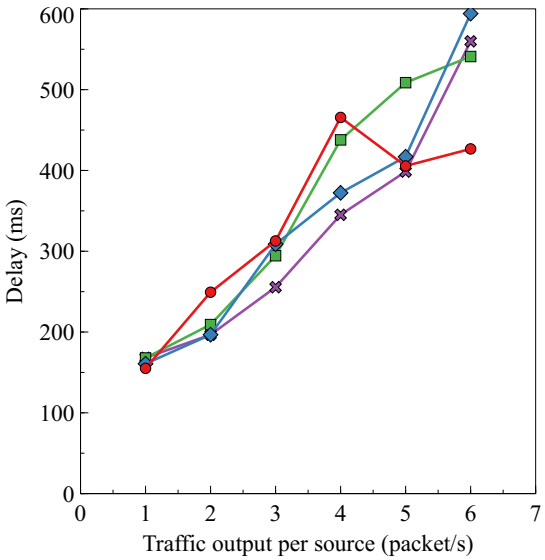
In figure 3.5b, the impact of the traffic density is observed. The more traffic, the harder it is for all algorithms and performance drops significantly. MRHOF tends to keep the same paths through the simulation because of the static ETX. This causes a lack of load balancing and the impact is important: MRHOF performance is very low (PDR of 2%) when traffic is heavy. E-OF performs slightly better because the energy saving will lead to some load balancing. Indeed, if one node is often chosen as a parent its battery will be drained faster, its rank will increase and it will no longer by a parent for a least some time, giving it some respite. NL-OF and F-OF have the best PDR with respectively 80.0% and 94.1% when the traffic is low and 30.0% for NL-OF and 30.1% for F-OF at the highest traffic load. When the traffic is high, the difference between NL-OF and F-OF is not significant.

On figure 3.5c, F-OF has a PDR of about 20.6% which is higher than the PDR of MRHOF which is 2.6%. The relative difference between the PDRs of NL-OF and F-OF is such that F-OF is only slightly better by 1.9%, which is not very significant. The values of PDR are lower compared to CBR scenarios (figs. 3.5a and 3.5b) for F-OF and NL-OF. The variable rate at which packets are generated will cause congestion to change in terms of location and through time. If many packets are suddenly generated for several seconds, energy consumption will suddenly increase accordingly and this will force packets to be routed through another path. In summary, VBR traffic model increases the instability of the topology.

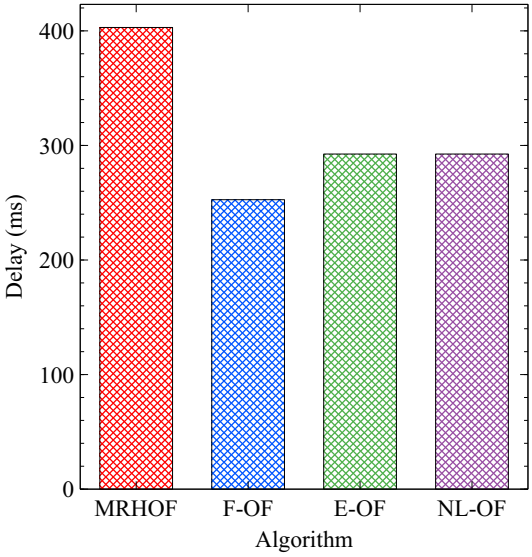
Delay Figure 3.6 shows delay values for the different scenarios. It is important to note that if PDR is low, delay is less significant. Indeed, if fewer packets reach the destination the average delay will have a wider error bar. On figure 3.6a, delay decreases with the increasing of the number of sources.



(a) The average delay with varying number of sources.



(b) The average delay for different traffic loads.



(c) The average delay with a VBR.

Figure 3.6: Study of changing either the number of sources or the traffic load on the average end-to-end delay of packets.

When more sources are present, the average length of the path to reach the destination is shorter, decreasing delay. Additionally, more paths are available to route packets allowing better load balancing. The significant variations in E-OF are due to the low PDR (see fig. 3.5a) and resulting increased variance. All algorithms show similar performance.

On figure 3.6b, the increasing traffic density causes delay to increase because of more congestion. All algorithms have similar results. On figure 3.6c, the delay of MRHOF, E-OF and NL-OF is higher by 59.5%, 15.8% and 12.0% respectively compared to F-OF. The proposed solution performs slightly better compared to the other solutions because of the consideration of all metrics at once, leading to better load balancing and reducing the consequences of the instability created by the VBR.

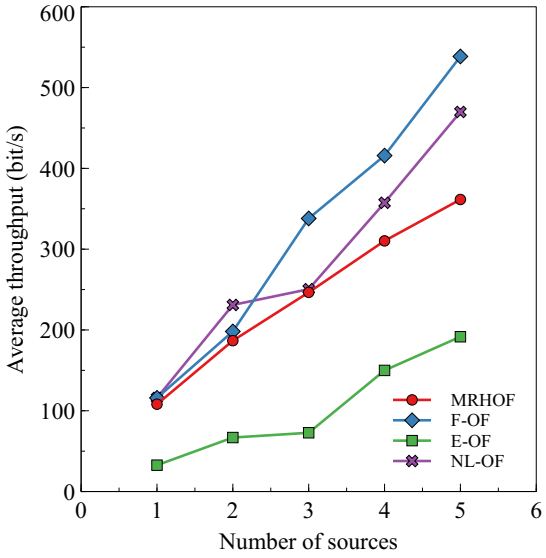
Throughput Throughput values depicted here represent the average throughput at the sink. On figure 3.7a, the throughput increases rapidly as there are more sources because more packets are generated on the topology. However, the rate of throughput increase is different for each algorithm. This rate is related to the PDR (fig. 3.5a) as a higher PDR means more packets make it to the destination and the throughput will be higher. E-OF has both the lowest starting throughput at 32.8 b/s and the lowest ending throughput at 192.0 b/s. The three other functions start off at approximately the same value, that is 108.0 b/s for MRHOF and 116.0 b/s for NL-OF and F-OF. When the number of sources reaches 5, the throughput values are 361.6 b/s, 469.6 b/s and 538.4 b/s, respectively for MRHOF, NL-OF and F-OF. Thus, F-OF performs better than MRHOF by 48.9% and better than NL-OF by 14.7%.

As with the increasing number of sources, the effect of increasing traffic density on throughput is visible on figure 3.7b and is related to the PDR (fig. 3.5b). MRHOF's throughput drops the most followed by E-OF. F-OF yields to the best throughput when the charge is low (552.0 b/s) which is better than NL-OF (469.6 b/s) by 140.8%. However, although NL-OF and F-OF are the best at heavy charges, with respective throughputs of 1057.6 b/s and 1087,2 b/s, the difference is less significant (F-OF is only 2.8% better).

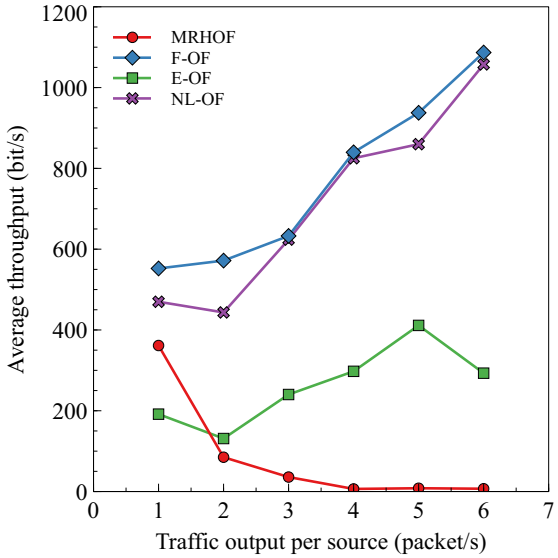
In the case of VBR (fig. 3.7c), F-OF and NL-OF perform the best with similar results. MRHOF has the lowest value with 45.7 b/s followed by E-OF with 122.4 b/s. This results are similar to the one from figure 3.7b.

Energy consumption Here, maximum energy consumption refers to the node with the highest consumption. Figure 3.8a shows that the difference between all algorithms is only marginal (no more than 1%), though E-OF still is the algorithm consuming the least power, as expected.

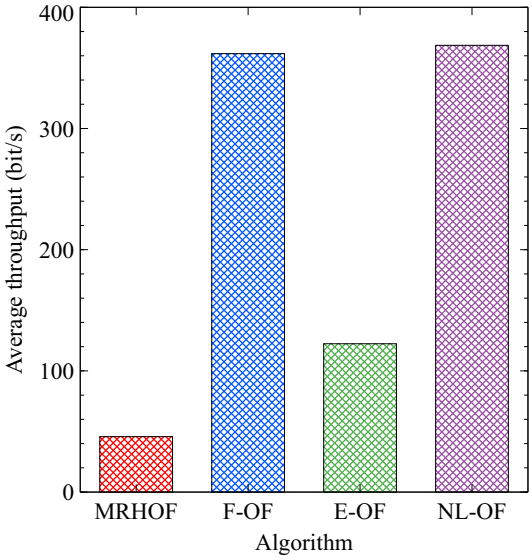
On figure 3.8b the difference is more significant, with NL-OF and F-OF being similar, E-OF performing better and MRHOF consuming the least power. The reason why the consumption of E-OF is decreasing towards the end is because of the decreasing in PDR and throughput (figs. 3.5b and 3.7b). Similarly, the low PDR (fig. 3.5b) and resulting low throughput (fig. 3.7b) of MRHOF explain why its energy consumption is the lowest of all algorithms. The same is seen on figure 3.8c, where the consumption of MRHOF and E-OF are lower compared to F-OF and NL-OF due to lower PDR (fig. 3.5c).



(a) The topology throughput against the number of sources.

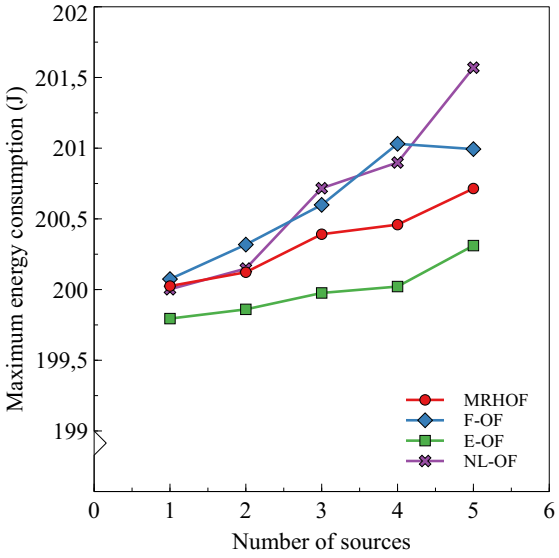


(b) The topology throughput given different traffic densities.

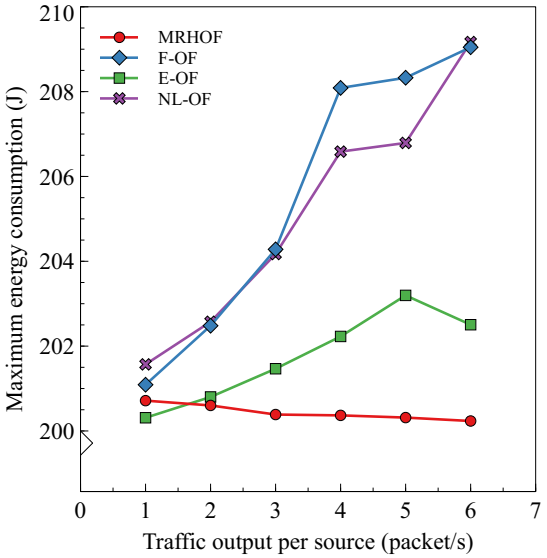


(c) The throughput at the sink of the different OFs.

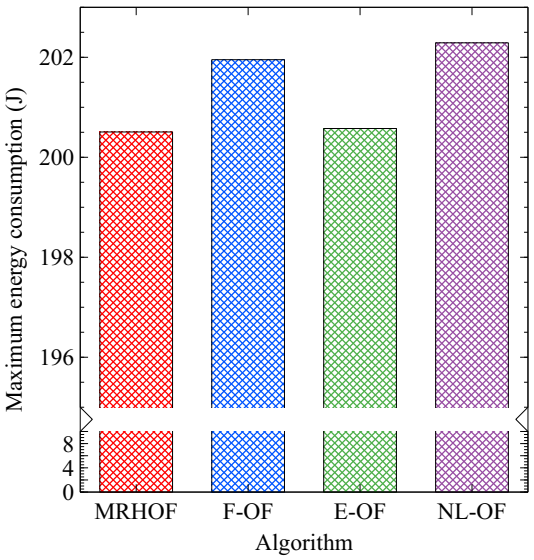
Figure 3.7: Changing the number of sources or the traffic density affect the throughput differently.



(a) The maximum energy consumption with varying number of sources.



(b) The maximum energy consumption for different traffic loads.



(c) The node with the highest energy consumption of all OFs.

Figure 3.8: Study of changing either the number of sources or the traffic load on the maximum energy consumption of any node (Note the ordinates axis-cut).

3.8.3 Dynamic scenarios

This section presents results with the addition of mobility and the concept of obsolete parent (see sec. 3.7). Simulations were run using OMNeT++ 5.4.1. This simulator is used without additional framework (e.g. INet) to keep the simulation lightweight and to allow having more control over the different runs. The proposed solution (POF) is tested against the Objective Function Zero (OF-0) [90], the Minimum Rank with Hysteresis Objective Function (MRHOF) [91] with the squared ETX as a metric and the Non-Linear Objective Function (NLOF) [111] (see section 3.3.2). POF uses a combination of delay, ETX, residual energy and velocity of node to influence the rank. Four scenarios will be presented in section 3.8.3.1 and in the fourth scenario the Simulation of Urban MObility (SUMO) is used to generate realistic mobility patterns of users on a map from OpenStreetMap. As a SUMO instance is running, vehicles are created, moved and destroyed. This information is passed in real-time to the OMNeT++ simulation using the Veins project[3]. Veins is only used to manage mobility without adding other functionalities. We use a $-log_{10}(d)$ based path loss model [123] [124] with a maximum range of about 105-110 m.

3.8.3.1 Simulation setup

Table 3.2 summarizes the parameters of the different scenarios. The simulation time for the first three scenarios is 8000 seconds and there are 10 repetitions. 84 nodes are randomly placed on a 500 m · 500 m playground (the sink is placed in the middle). In scenario 1, 21 nodes generate traffic whereas in scenarios 2 and 3, 84 nodes generate traffic.

The traffic model uses Variable Bit Rate (VBR) for all three scenarios. VBR has been chosen because usage can be anything such as a multimedia stream (watching a movie from within a car), Internet browsing, VoIP, etc. This causes traffic generation and forwarding to be variable from one user to the next. Also, for some specific uses the bitrate can be variable, for instance the Speex codec used in several VoIP applications [125].

In scenario 1, all nodes are static (no mobility), in scenario 2 nodes have a velocity of up the 0.5 m/s (low mobility) and scenario 3 uses high mobility with nodes having a speed of maximum 1 to 15 m/s (see table 3.3). Mobile nodes have a random speed between 0 m/s and the maximum. Their speed is constant throughout the simulation run and when they reach the limits of the playground, they change direction so they stay in the playground.

The average traffic load is the studied parameter in scenarios 1 and 2, whereas the velocity of nodes is the studied parameter of scenario 3 (studied parameters are in bold in table 3.2). The size of data packets is 1280 octets, which is the size of the IPv6 minimum link Maximum Transfer Unit (MTU) [18]. All nodes are supposed to use 802.11 with a throughput of 10 Mbps except for the sink which has 50 Mbps [8]. The maximum radio range of nodes is set to approximately 110 m. Hence, nodes do not require cellular access to reach Internet.

Finally, all nodes have a random amount of residual energy drawn between 54 joules to 162 joules, except the sink which has an unlimited amount of power. The energy capacities were chosen so that the depletion of battery is significant on the

[3] https://veins.car2x.org/

Parameter value	Scenario 1	Scenario 2	Scenario 3	Scenario 4
Simulation time (s)	8000	8000	8000	2000
Repetitions	10	10	10	10
Nodes amount	85	85	85	≈10-100
Sources amount	21	84	84	≈10-100
Average traffic (packets/s)	**1 to 39**	**1 to 39**	10	10
Maximum velocity (m/s)	0	0.5	**1 to 15**	8.33 or 55.55
Data packet size (Bytes)	1280	1280	1280	1280
Links throughput (Mbps)	10	10	10	10
Maximum range (m)	110	110	110	110
Residual energy (J)	[54, 162]	[54, 162]	[54, 162]	[54, 162]

Table 3.2: Simulations setup. The bold text represents the studied parameters for each scenario.

m/s	kmph	mph	Example
1.50	5.40	3.36	Pedestrian
3.50	12.60	7.83	Parking driving
8.50	30.60	19.01	School area
15.00	54.00	33.55	City road

Table 3.3: Instances of typical velocities.

timescale of the simulations [121] (1/1000 the capacity of a typical smartphone, which is 3000 mAh @ 9.0 V.).

Scenario 4 is different because the mobility of nodes is achieved through SUMO/Veins in real-time. This causes runs the be significantly longer (they require more processing power), hence the simulation time of 2000 seconds. The playground is a 1000 m · 1000 m area (from a part of Montreal city, Canada) to have enough streets to generate the random trips. Nodes are created and removed live and at any time several tens of them are present (except at the very beginning where there are fewer nodes) and all those nodes are sources. 16 sinks in a 4 · 4 grid configuration are present during the whole simulation because the rather short lifetime of nodes and the larger area would cause connectivity issues. Those sinks each manage a sub-network as explained in section 3.5 and can be parked cars used as relays like in [83]. Users can move from one sub-network to another one. Nodes are created at a rate of one every 1.5 second. 90% of them are cars driving up to 55.55 m/s[4] (or the speed limit of the road they are currently driving on) and 10% of them are bicycles with a maximum velocity of 8.33 m/s. The other parameters are similar to the aforementioned scenarios.

Table 3.3 shows typical velocities in different unit systems along with an example for each case. As a reminder, 1.0 meter per second (m/s) equals to 3.6 kilometers per hour (kmph) and to roughly 2.24 miles per hour (mph). The studied velocities range from static (0 m/s) to approximately the typical speed limit of a city road (15 m/s).

[4] which is ≈ 200 kilometers per hour or ≈ 124 miles per hour.

3.8.3.2 Simulation results

Four scenarios are studied, as stated in the previous subsection (sec. 3.8.3.1). The first scenario has no mobility and is intended as a base case. The results tend to have a significant error because the random placement of the static nodes on each run has an impact on the possible choice of paths. Scenario 2 adds low mobility (nodes move at a maximum speed of 0.5 m/s) to show how the algorithms react on a changing topology. In both scenarios 1 and 2, the average traffic is increased to study its effect on the performance of the different solutions. Scenario 3 shows the study of the impact of increasing the linear velocity of nodes. Finally, scenario 4 adds realistic mobility thanks to SUMO.

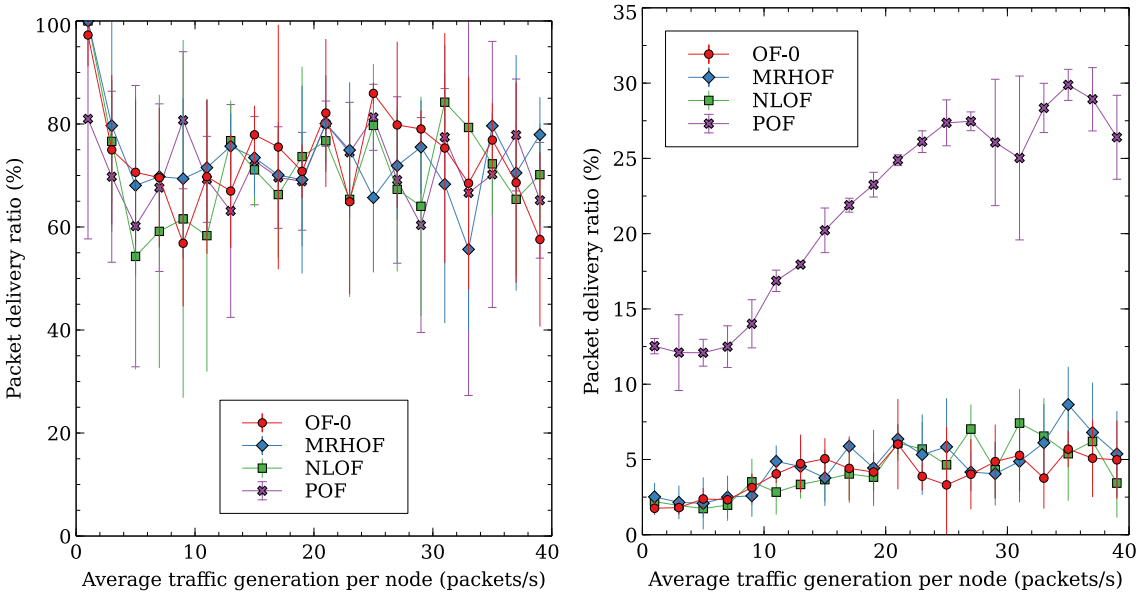
The four OFs that are tested are OF-0 (red circles in graphs), MRHOF (blue diamonds) and NLOF (green squares) and the proposed solution which uses a fine-tuned POF integrating several metrics (see sec. 3.6). Two different versions of the Programmable Objective Function are depicted: POF and POF-rel. POF (purple crosses in graphs) takes into account the velocity of the current node only, whereas POF-rel (orange pluses) uses the relative velocity of nodes (current node and parent) (see section 3.7). POF-rel has only been tested on the third and fourth scenarios. To add clarity, only the error bars of POF and POF-rel have a horizontal marker to delimit them on line charts.

Packet delivery ratio Figure 3.9 shows the PDR on the different studied scenarios (note the different scales on the vertical axes). On the base scenario (fig. 3.9a), no nodes are mobile. This lack of motion results in a finite number of possible paths for each node, limiting the impact of considering several metrics. The PDR gets slightly lower as the traffic charge increases as a result of nodes exhausting their battery (see fig. 3.14a). The PDR stays higher compared to mobile scenarios (fig. 3.9b, fig. 3.9c and fig. 3.9d) because the topology continuously changes, causing instability.

On figure 3.9b, the PDR of OF-0, MRHOF and NLOF are between 2.5% and around 5%. We see the PDR of POF goes from about 12.5% at low charge up to about 28% at higher charge. POF considers velocity and when low, it has a greater impact on the computation of rank than any other metrics. Indeed, whenever the charge is low, the delay (fig. 3.11a) and energy consumption (fig. 3.13a) are rather low and do not impact the rank much. As the traffic generation increases, these will have more impact, improving the PDR of POF.

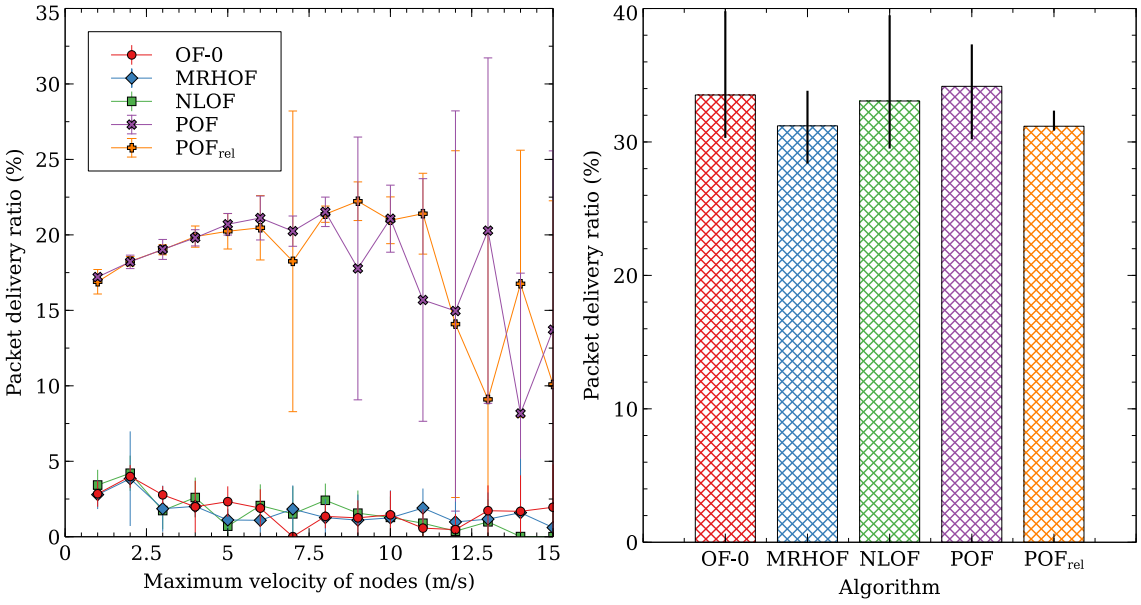
The PDR resulting from third scenario is seen on figure 3.9c. POF and POF-rel perform better than state of the art solutions, thanks to the integration of speed in the computation of rank. The slight increase at lower charges is caused by the same reasons as stated above for scenario 2 (fig. 3.9b). However, at high velocities all algorithms show a decrease in PDR because the high speed of nodes causes connections to last a shorter amount of time. We also see the error increasing due to more instability as speed increases. Yet, the proposed solution performs about up to 5 times better in scenarios 2 and 3. The reason why POF and POF-rel have very similar results is because of the constant speed and direction of nodes. This results in the relative speed not changing much through time (except when nodes reach the limits of the playground).

In scenario 4 (fig.3.9d), the PDR of the different solutions is similar. This is because of the good coverage due to the grid of sinks. POF-rel still is more stable



(a) Scenario 1: no mobility.

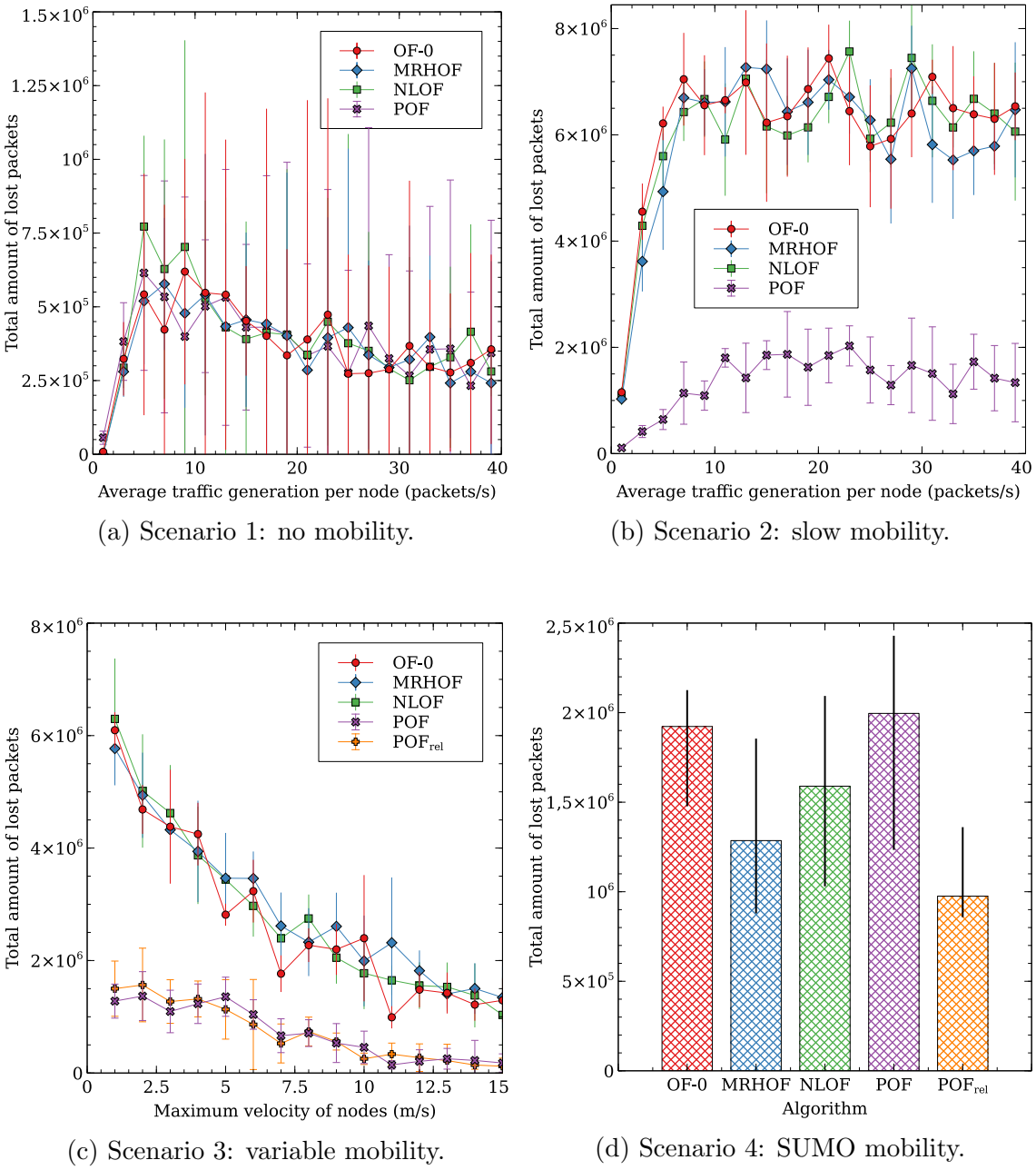
(b) Scenario 2: slow mobility.



(c) Scenario 3: variable mobility.

(d) Scenario 4: SUMO mobility.

Figure 3.9: Study of PDR under different mobility scenarios.



(a) Scenario 1: no mobility.

(b) Scenario 2: slow mobility.

(c) Scenario 3: variable mobility.

(d) Scenario 4: SUMO mobility.

Figure 3.10: Study of the packet loss.

with a lower variance even when compared to POF. Indeed, in this scenario the velocity and direction of vehicles is very variable and it is important to consider the relative velocity of nodes.

Packet loss Figure 3.10 shows the packets loss on the different studied scenarios. In the first case (fig. 3.10a), the losses increase for all solutions from about $7.5 \cdot 10^4$ to roughly $5 \cdot 10^5$, then decline to $3.75 \cdot 10^5$. The error is caused by the lack of choice of paths due to static nodes. The amount of lost packets decreases slightly towards the end as nodes exhaust their battery (see fig. 3.14a).

In scenario 2, state of the art solutions have lower packet loss at low charge and it increases quickly up to around $6.5 \cdot 10^6$. The proposed solution also has an increase in losses earlier on and stabilizes afterwards. The higher PDR (fig. 3.9b) of POF explains why there are less packet losses.

In scenario 3 (fig. 3.10c), as mentioned above, the increasing of speed causes more instability. The number of packet loss decreases for all algorithms because of lower connectivity. POF and POF-rel perform similarly. In scenarios 2 and 3, the proposed solution has about 3 times less packet loss compared to the others.

In scenario 4 (fig. 3.10d), vehicles on the same part of a street often drive either in convoy (one following the previous vehicle on a one lane configuration) or in opposite directions. OF-0, considering only the number of hops, can connect to vehicles driving in opposite directions, increasing losses. Given POF does not consider relative velocity, vehicles with a relative speed of 0 m/s are not favored and it also frequently connects to vehicles in opposite directions. MRHOF performs the second best because only considering ETX helps in the earlier detection of disconnections when vehicles drive in opposite directions. NLOF is somewhat in-between the aforementioned algorithms because ETX will impact rank in some cases. POF-rel favors vehicles with low relative speed, increasing stability: variance is the lowest (like in the study of PDR, fig. 3.9d) and losses are also lower.

End-to-end delay Figure 3.11 shows the delay on the different studied scenarios. In the static case (fig. 3.11a), all solutions have the same performance. Delay increases at higher charge because more congestion occurs.

In scenario 2 (fig. 3.11b), all algorithms also have similar performance, but delay increases with a steeper slope because more disconnections occur. We note the proposed solution is able to deliver more packets (fig. 3.9b and fig. 3.10b) with the same delay compared to the state of the art.

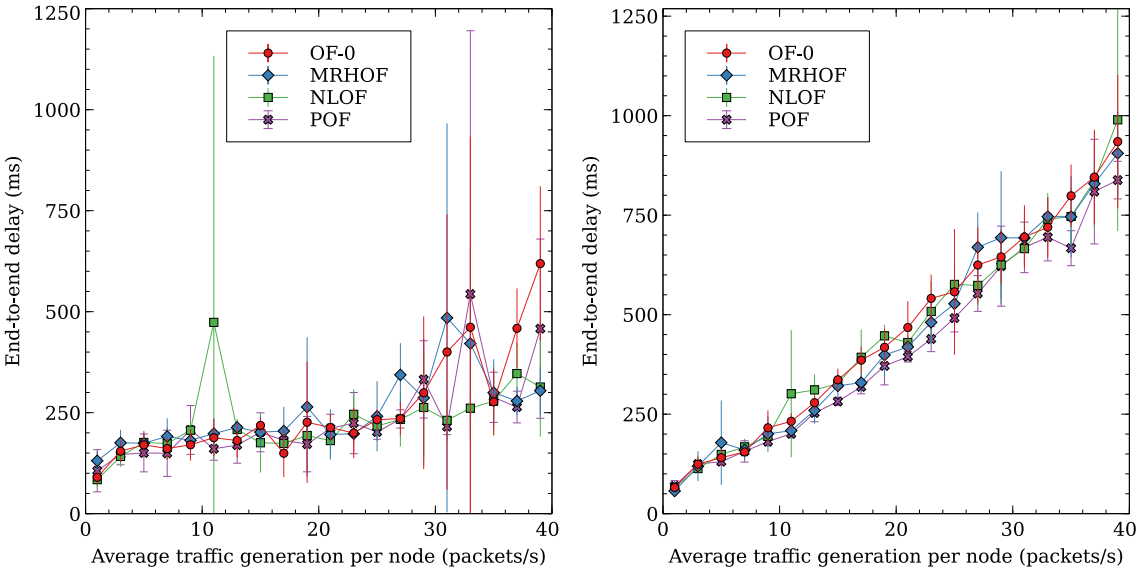
The third scenario depicted on figure 3.11c shows the instability due to higher speed. In this scenario, all source nodes generate an average traffic of 10 packets/s (see table 3.2): at low velocities, the delay is slightly higher (around 200 ms) compared to scenario 2 when the charge is around 10 packets/s. Indeed, in scenario 3 the velocity of nodes is higher than in scenario 2. We see the speed has a lower impact on delay compared to the amount of traffic. This is due to lower connectivity (see fig. 3.10) causing lower network load at higher velocities.

In scenario 4 (fig. 3.11d), delay is shorter compared to other scenarios because there are more sinks. Paths are thus shorter and connectivity is better. However, POF-rel performs significantly better compared to all other solutions. Taking into account the relative velocity causes better connections. Less re-transmissions accounts for shorter delay.

Throughput usage Figure 3.12 shows the throughput usage on the different studied scenarios. High PDR and few re-transmissions explain the very low usage of links in scenario 1 (fig. 3.12a). Note the scale on the vertical axis. As traffic load increases, so does the usage of links.

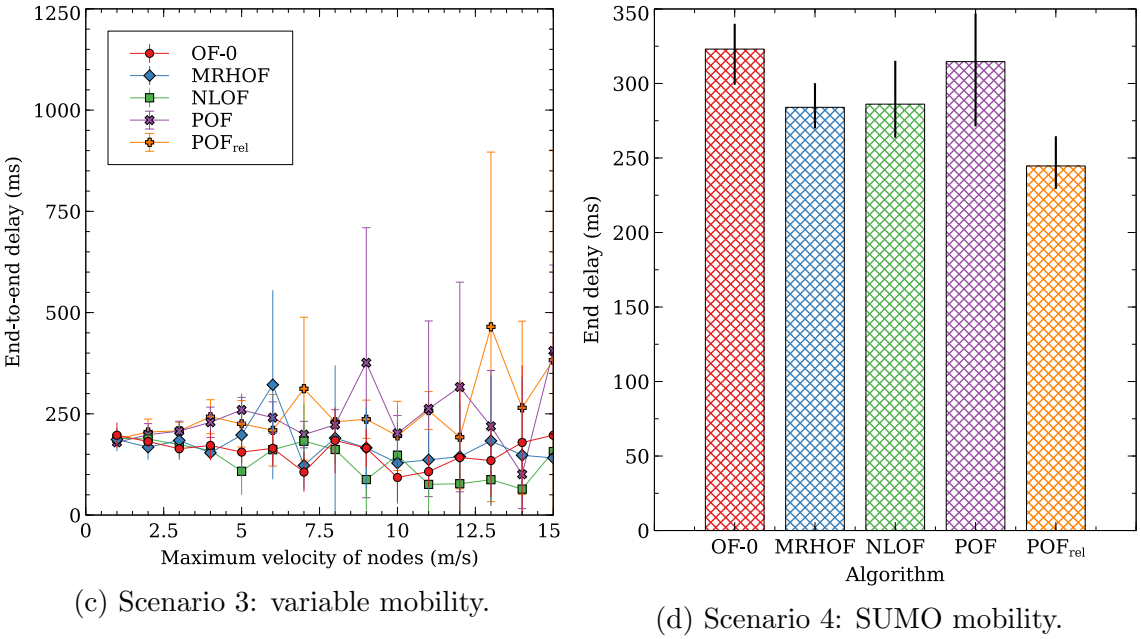
In scenario 2, more re-transmissions cause a higher usage of the links. In the case of POF, the usage of links is similar to others even though its PDR is better (see fig. 3.9b). This is because POF transmits more data packets instead of re-transmitting non-acknowledged packets.

This is the same for POF and POF-rel in scenario 3 (fig. 3.12c). The usage of links decreases at higher velocities, again because of less connectivity.



(a) Scenario 1: no mobility.

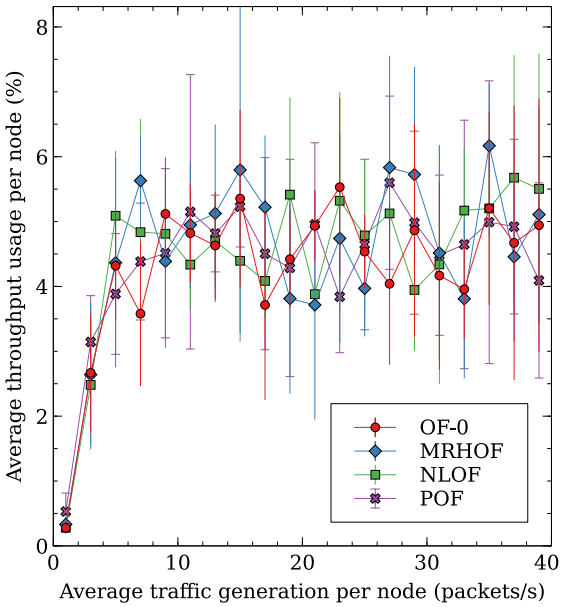
(b) Scenario 2: slow mobility.



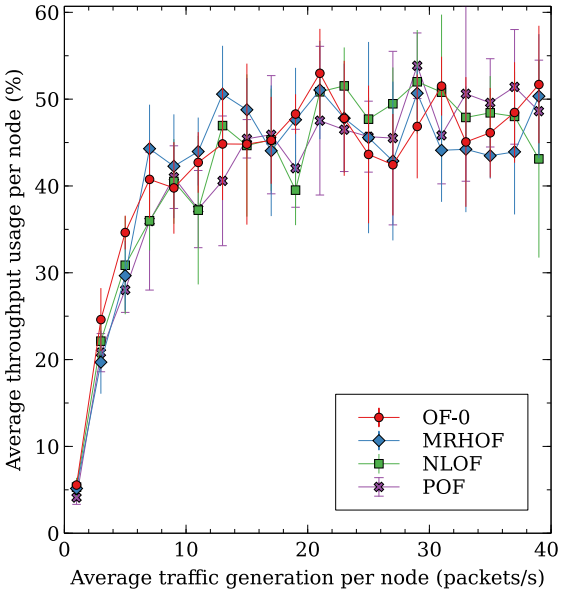
(c) Scenario 3: variable mobility.

(d) Scenario 4: SUMO mobility.

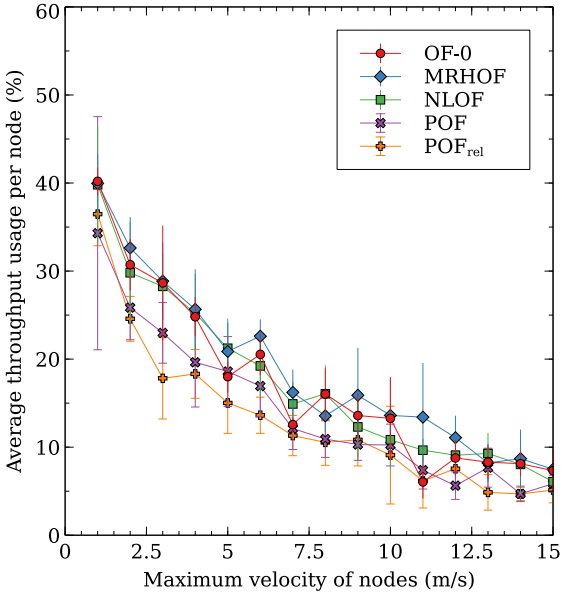
Figure 3.11: Study of delay in different mobility scenarios.



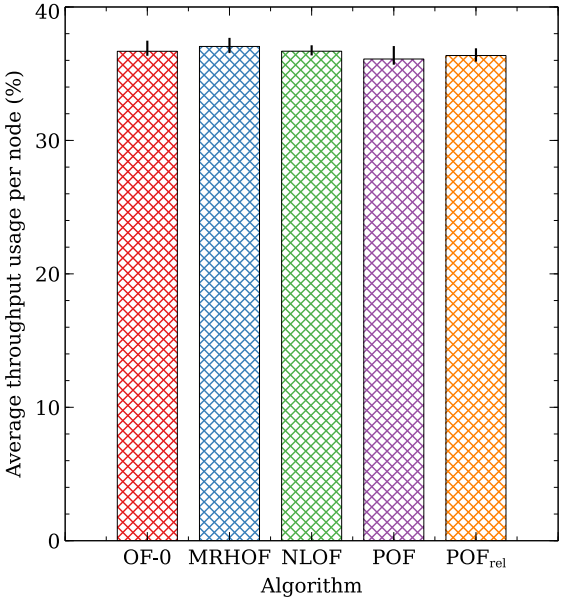
(a) Scenario 1: no mobility.



(b) Scenario 2: slow mobility.

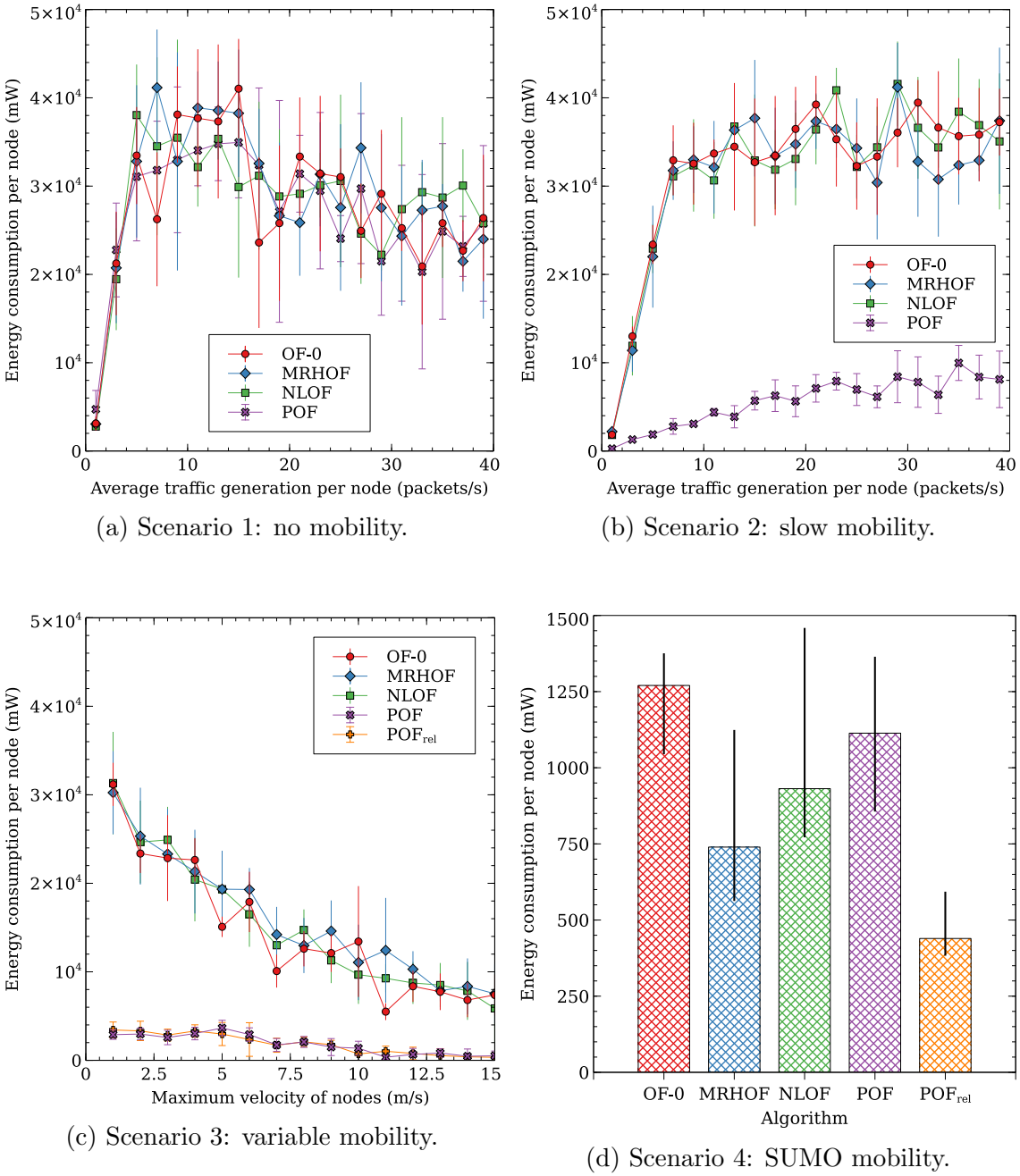


(c) Scenario 3: variable mobility.



(d) Scenario 4: SUMO mobility.

Figure 3.12: Study of link throughput usage.



(a) Scenario 1: no mobility.

(b) Scenario 2: slow mobility.

(c) Scenario 3: variable mobility.

(d) Scenario 4: SUMO mobility.

Figure 3.13: Study of energy consumption.

In scenario 4, the use of links is very similar between all algorithms (see fig. 3.12d), even though POF-rel has less losses (fig. 3.10d). Source nodes will consume less throughput because less re-transmissions are required, but sinks will have to absorb significantly more traffic. For the other algorithms, more re-transmissions between mobile nodes will occur and sinks will be less solicited. On average results are thus comparable.

Energy consumption The average energy consumption per node is shown in figure 3.13. The consumption increases rapidly with traffic in the no mobility scenario (fig. 3.13a) and the low mobility scenario (fig. 3.13b). As more packets need to be sent, more throughput is used and more energy is spent sending/receiving data packets. There is a decline at higher loads in figure 3.13a and the same behavior is seen in the amount of dead nodes (see fig. 3.14a). Indeed, the stability of the static scenario allows more packets to be sent and the nodes will deplete their battery

faster, especially at higher load.

Nodes will also consume more energy as traffic increases in scenario 2 (fig. 3.13b), depleting their battery (see fig. 3.14b) and causing the consumption to stabilize on average. The better PDR (fig. 3.9b) and lower packet loss (fig. 3.10b) of POF explain the lower power consumption.

As depicted before, scenario 3 (fig. 3.13c) is more unstable, so less packets are sent as velocity increases. Still, POF and POF-rel perform better compared to others by consuming 5 to 15 times less energy.

In scenario 4 (fig. 3.13d), POF-rel consumes significantly less energy because less packets are lost (see fig. 3.10d) and less re-transmissions are required.

Network lifetime The network lifetime can be studied through the amount of dead nodes: figure 3.14 shows the amount of nodes running out of battery before the end of the simulation. More packets are sent when there is no mobility (fig. 3.14a) so more nodes are dying. At higher loads, the amount of dead nodes decreases because the most solicited nodes have their battery depleted much more quickly, breaking the connectivity of the network. When the connectivity is low, few nodes can send and receive packets so nodes consume less power on average.

When mobility is low (fig. 3.14b), the amount of dead nodes increases with traffic load. Indeed, whenever a node runs out of energy, the mobility allows other nodes to connect to a new parent (unlike in scenario 1). We see the proposed solution has no dying nodes except at higher loads because it does more load balancing by considering more metrics. The inclusion of speed in the OF also allows less losses (thus less re-transmissions) as depicted before.

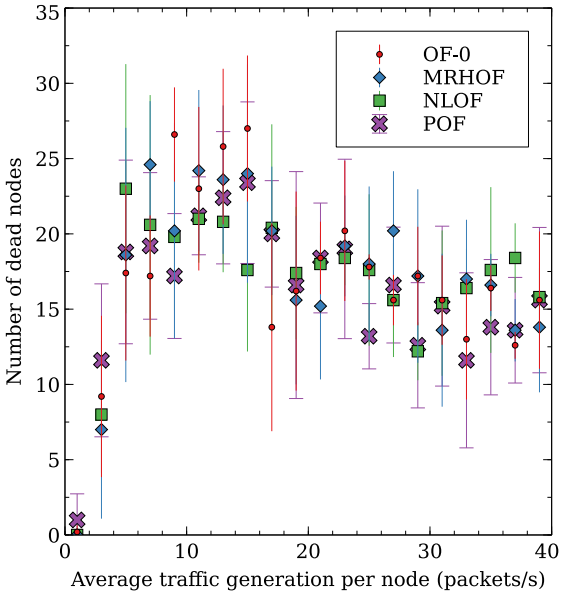
The same is seen in scenario 3 (fig. 3.14c): POF and POF-rel have no dead nodes. The other solutions also have fewer nodes running out of energy (the vertical scales are not the same on the graphs) because less transmissions occur as discussed previously.

Given nodes in scenario 4 have a short lifetime, none run out of battery so no graph is shown in this section for this scenario.

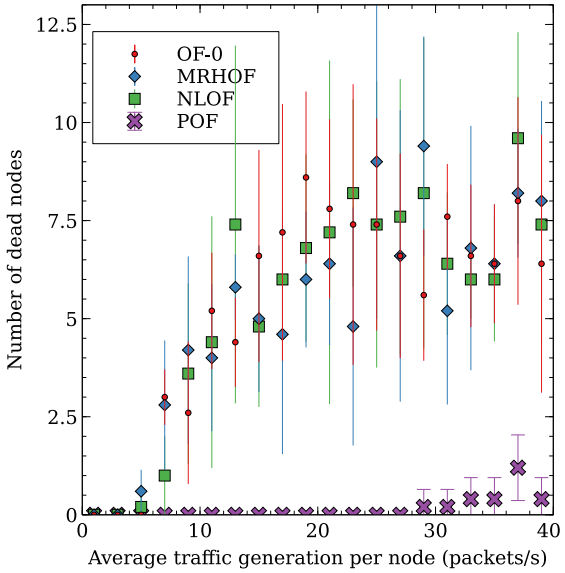
3.9 Conclusion

In this section, we discussed about a new SDN-inspired architecture in which the control server can send OFs to local areas to answer to users' needs. Once in place, an OF can evolve to follow what users expect in terms of QoS. The proposed programmable objective function allows to merge all available metrics together with different weights to favor certain aspects of the network. This enables to consider several needs, whatever they are, to include all devices and users in the topology. The first series of results (sec. 3.8.2) show POF performs better in cases with no mobility. The second series of results (sec. 3.8.3) represent the POF with the inclusion of velocity as a new metric and the increasing of rank through time to reflect the obsolescence of rank due to mobility.

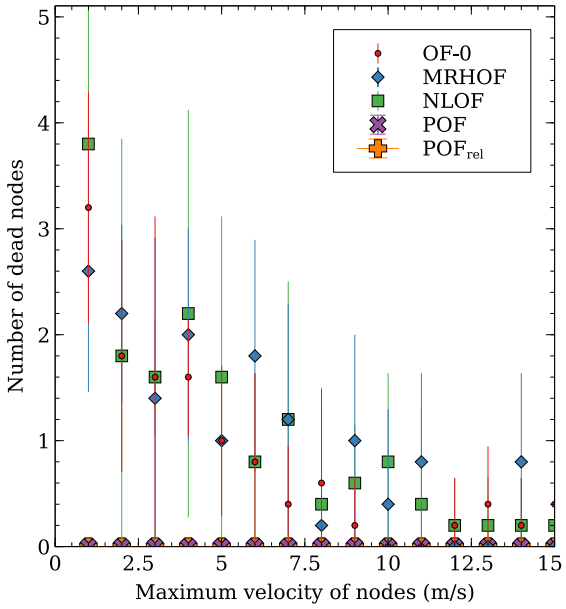
It is possible to further improve performance by also considering the mobility-type of users. A slow traveling user does not necessarily have the same requirements



(a) Scenario 1: no mobility.



(b) Scenario 2: slow mobility.



(c) Scenario 3: variable mobility. POFs have 0 dead nodes.

Figure 3.14: Study of the network lifetime. The vertical scales are different on the three graphs.

as a fast traveling one. Low velocity devices disconnect less often because it takes more time before they move out of range from one another. The next chapter will cover the aspect of determining the mobility-type of users. This will be used to deduce which device on a topology is the best choice to serve as a sink for other devices to connect to Internet through this device.

Chapter 4

Selection of relay nodes based on the classification of user mobility profile

Résumé français

Ce chapitre porte sur la sélection de relais au sein d'un réseau local pour permettre aux noeuds en faisant partie de se connecter à un réseau plus vaste, comme Internet. Pour ce faire, nous utilisons l'apprentissage artificiel pour classifier les noeuds en fonction de leur profil de mobilité, et identifions ceux étant le plus adaptés à remplir ce rôle. Ensuite, nous employons des métriques réseaux localisées pour identifier les zones critiques où placer un relais est prioritaire. Nous définissions une nouvelle métrique (la densité de paquets) pour permettre une meilleure identification ce ces zones.

L'intérêt de la solution proposée est d'élire un relais parmi les noeuds de la topologie. Contrairement à la plupart des solutions de l'état de l'art, nous n'utilisons pas un appareil dédié pour remplir cette fonction. Ceci permet une grande flexibilité quant au nombre de relais à élire et la fréquence à laquelle ce choix s'opère dans le temps. Une infrastructure dédiée n'est donc pas nécessaire, mise à part une antenne cellulaire à laquelle se connectent les relais (permettant ainsi aux noeuds connectés au relais d'avoir un accès à Internet).

L'architecture est donc composée d'appareils (capteurs, *smartphones*, vélos et véhicules motorisés connectés, etc.) et d'un point d'accès à Internet (l'antenne cellulaire précédemment mentionnée). Les appareils ayant un accès cellulaire font partie du *fog* et un serveur de contrôle est situé dans le *cloud*. Le serveur récupère les données utilisateurs liées à la mobilité et exécute l'algorithme d'apprentissage artificiel avant de sélectionner le ou les relais. Une fois les relais élus, ceux-ci initient la construction d'une topologie en arbre basé sur RPL. Un protocole de routage tel qu'expliqué au chapitre précédent peut désormais être utilisé pour le transport des données.

La classification du type de mobilité est basée sur des données comme l'accélération, la vitesse, la position (c'est-à-dire le type de route où le noeud se situe, etc.). Il faut attendre une certaine durée pour accumuler ces données de mobilité pour avoir une meilleure précision sur la prédiction. En effet, un temps trop court ne permet pas de déduire qu'un noeud est une voiture, par exemple, dans la mesure où celle-ci n'a peut-être pas eu le temps d'atteindre une vitesse

suffisamment élevée pour l'identifier en tant que telle.

Dans l'évaluation de performance, nous étudions le comportement de la solution proposée et d'autres algorithmes en fonction du temps de sélection des relais ou de la densité de trafic. L'évaluation est effectuée dans des scénarios à faible et haute densité d'appareils. Il en ressort que la solution proposée permet d'obtenir un taux de délivrance des paquets plus élevé, une stabilité plus grande de la topologie ainsi qu'une économie en terme d'énergie.

4.1 In a nutshell

In this chapter, the problem of finding a suitable relay to allow users to connect to Internet is addressed. A mobile relay can be any user with Internet connectivity that can forward back and forth data to other users on the network. Finding a suitable relay is first achieved through ML using a classification algorithm to deduce the mobility-type of users (soft-mobility users like pedestrians or high-mobility users like cars). Then, critical areas are found with a novel metric to decide where relays will be elected. The main aspects of this chapter are:

- The use of a machine learning classification algorithm to deduce the mobility-type of users in order to select the best relay;
- The addition of localized network metrics such as delay to help in the selection process of relays;
- The definition of a new metric, the Expected Packet Count (EPX).

4.2 Context

As explained in the previous chapter, mobility causes connectivity issues because of the increased instability of the topology: users move out of range of one another, breaking links and new links become available as well. Furthermore, not all users might have a direct Internet access through a cellular antenna (3/4/5G). Multiple reasons can explain such a situation like the lack of infrastructure (deployment costs too high or low populated area not being profitable for an Internet Service Provider (ISP)), user not having a subscription plan (too expensive or person traveling abroad), device not compatible with local frequency bands (different frequency bands might be available depending on the location [126] [127] [128]) or because radio conditions are bad (path loss, shadowing, multipath fading due to obstacles like buildings or bad weather).

In the IoV, the high mobility of users exacerbates the problem of cellular connectivity due to devices leaving range of base stations (BSs) or of one another (in the case of ad hoc networks). One possible solution is to deploy either Static Relays (SR) or Mobile Relays (MR). A relay is a device that extends the capacities of a BS by connecting to it and by offering connectivity to other devices. The traffic of those devices is forwarded back and forth. They can thus access Internet (for instance) without the need to directly connect to the BS. The relay can be a dedicated unit or a standard device that is part of the network and can use different wireless technologies. For instance, a smartphone equipped with LTE and WiFi technologies can serve as a relay by forwarding WiFi traffic from nearby devices to

a cellular antenna using LTE. A common example of this is when using a smartphone with tethering to connect a computer to Internet. However in tethering, there is only one WiFi hop.

4.3 Related work

Several solutions have been proposed that make use of static relays (dedicated architecture such as Road Side Units (RSU)), Mobile Relays (MR) [129] or parked vehicles [83]. The use of static relays has limitations such as deployment cost, lack of flexibility and mandatory regular maintenance of the equipment to ensure reliability. On the other hand, the use of MRs allows for more flexibility and can lower deployment costs.

We categorize state of the art solutions about the selection of a relay in three groups: general approaches, ML based and Cluster Head (CH) based. General approaches (subsection 4.3.1) concern the selection of relays based on a mathematical model. They can be based upon a maximization problem, Markov chains or cost function. ML based approaches (subsection 4.3.2) use machine learning to select the relay. Finally, CH based solutions work by electing one relay for each group (or cluster) of users, hence the appellation "cluster head" (subsection 4.3.3).

4.3.1 General approaches

The solution proposed in [130] aims to maximize the system throughput by satisfying QoS requirements of users by considering a power constraint. The model is composed of one antenna, several fixed relays and users. Users far from the antenna and suffering from a low quality link are eligible to connect through a relay. The maximization problem allows to find the best relay for each user. However, given relays are dedicated units and no mobility is considered, the solution is not adapted in a highly mobile environment with potential relays moving in and out of the studied area.

Authors in [131] propose a solution to select mobile relays amongst vehicles to allow other vehicles to connect to Internet through them. Neighboring vehicles are connected through WiFi and relays connect to Internet using LTE. Several metrics are used for the selection process such as speed, Received Signal Strength (RSS), stability of links and distance between vehicles. Though this solution is good, it does not consider several mobility types and simulations are done in straight lines on highways.

In [132], the authors propose a relay selection scheme where Mobile Users (MU) select relays according to a cost. The model is composed of a base station, several relays and MUs. The mobility of users is represented by a transition matrix and the problem is modeled using a constrained Markov decision process. Though the mobility of users is accounted for in this proposition, the use of dedicated relays constrains the problem.

Authors in [133] propose a mechanism of relay selection to assist nodes showing high interference. The model is made of a macro cell BS, a few pico cell BSs and several users. When a user is detected to endure high interference from several pico cell BSs, the macro cell BS asks the user to search for a close neighbor to act as a

relay. The best relay is then selected to forward network traffic from/to the user. This solution is interesting as any node can potentially relay data to help another node. However, it uses static nodes and is not intended to use in a context with high mobility.

4.3.2 Machine learning based solutions

In [134], the authors propose to use deep learning to select the best relays to enhance dissemination of data. The studied case is made of a road network with static RSUs near the roads and vehicles driving around. Although the use of several features is good to make a more relevant choice, relays are static and dedicated units. This makes this solution non-adapted if the deployment cost of the infrastructure is too high or if it is not possible to deploy a dedicated architecture at all.

In [135], the authors propose a powerful model using the K-nearest neighbors ML algorithm. They consider a vehicular network composed of several RSUs and many vehicles driving around. The idea is to connect a vehicle directly to a RSU if one is in range, otherwise it tries to find a path through other vehicles to connect to a RSU using several features. Though the model uses many relevant features in the context of mobility, it relies on static relays and only motorized vehicles are considered.

Authors in [136] propose a Fuzzy-based Q-Learning Routing Protocol to find the best next hop based on the node's residual energy, movement and buffer space. Q-learning uses a reward system as reinforcement learning to optimize the selection of the next hop to forward packets. The proposed solution is tested on different scenarios with several tens of nodes reaching velocities up to 1.5 m/s. However, in the context of IoV there can be up to hundreds of users and they can move significantly faster.

4.3.3 Electing a relay through cluster head selection

The solution proposed in [137] exploits the proximity of buses at intersections to forward data. Bus density, road connectivity and the path of bus lines are used to determine on which bus the data is forwarded. This solution relies on the presence of a bus network with a minimum density to efficiently forward data. Also, the delivery delay from simulations ranges from 7 to 10 seconds. These make the proposed solution inadequate in areas without bus lines or for applications needing a real-time latency.

Al-Kharasani et al. [138] propose a clustering solution to connect nodes in a VANET. Several metrics like bandwidth, connectivity, velocity and distance are used to create a cluster of vehicles. However, the proposed solution is tested in scenarios using constant speed and motorized vehicles (i.e. high-mobility users) only.

In [139], the authors propose a scheme where vehicles with similar velocity are regrouped in clusters. The cluster head is also connected to the infrastructure using a cellular link. The clustering allows to regroup vehicles with a similar trajectory together so only the head need to be connected to the infrastructure or to other clusters' head. However, in the context of urban mobility several different types of

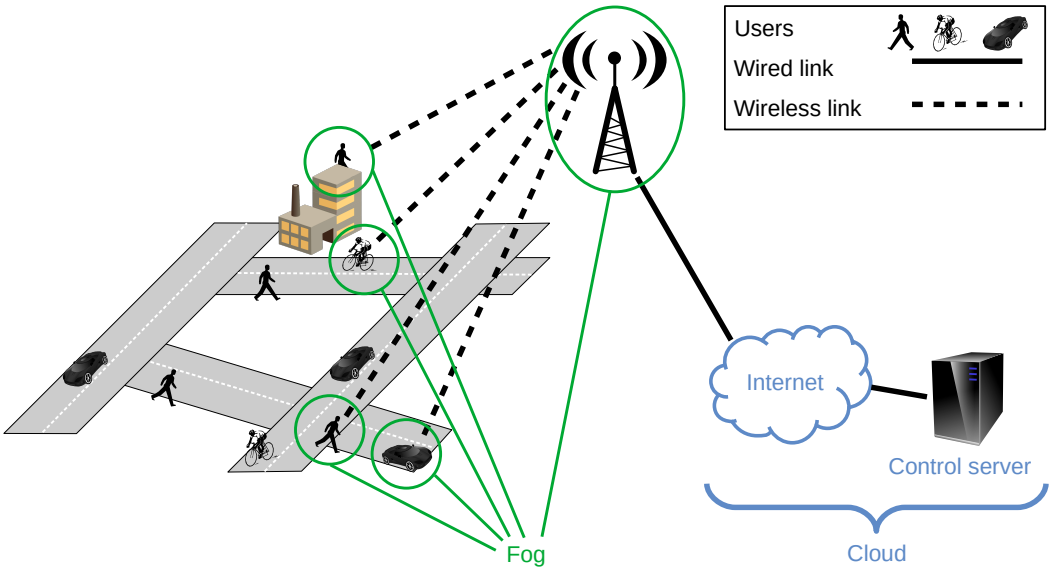


Figure 4.1: The proposed architecture made up of the *Fog* part and the *Cloud* part. Here, 4 users have access to Internet through the cellular antenna.

users are present and many intersections will cause users to move apart with a high probability.

4.4 Motivation

Most solutions from the previous section focus on the selection of relay that users can connect to. Relays - mobile or not - are usually dedicated devices as well or make use of some information considered as known beforehand. They tend to focus on one type of mobility (for instance cars only) or all users are considered to move slowly.

The placement of relays is a challenging issue in the context of IoV because of the difficulty in foreseeing where the users will be located and what they will require in terms of service. Placing static relays may not be optimal as some (or many) of them could be located where only a handful of users are present. On the other hand, other relays could be very much saturated, being placed in an area with a dense amount of users. Furthermore, static relays induce a cost if they are dedicated units.

One solution is to use mobile relays which can greatly help in adapting the topology to very dynamic contexts, without incurring a higher cost. If mobile relays are elected amongst standard users the solution becomes even simpler because a dedicated infrastructure is not required any longer. Only a cellular antenna must be present so relays can connect to it.

4.5 Proposed architecture

The proposed architecture is depicted on figure 4.1 and it is made up of two parts: the *Fog* part and the *Cloud* part. Different types of user are supposed to be present

in the environment: soft-mobility users such as bicycles or pedestrians and high-mobility users like cars, trucks or buses. A "user" can be a smartphone, a smart device like bicycles and e-scooters part of a sharing service or on-board computers found in motorized vehicles. Users with Internet connectivity and the cellular antenna (all circled in green on fig. 4.1) compose the *Fog* part of the network. On figure 4.1, fog users have a wireless link to the antenna represented as a black-dashed line. Internet and the Control Server are the *Cloud* part of the architecture (in light blue on fig. 4.1).

The algorithm will run either using a two step mode (*initialization mode*) or a four step mode (*on-going mode*). The *initialization mode* is as follows:

1. Users that have Internet access connect to the controller to send their mobility data (such as velocity, acceleration, type of road the user is on, etc.) as depicted on figure 4.1;

2. The controller uses the data to determine the mobility profile of the users. Users with the appropriate profile are selected as relays.

This mode is executed when there are no users and the network is yet to be built. Once one or several relays are elected (depending on the total amount of users and local needs), the placement of MRs can be further optimized as required by running the second mode. The *on-going mode* is executed when there is already a topology in place but changes or users' needs evolve and adjustments are required:

1. Users send to the controller their mobility data (such as velocity, acceleration, type of road the user is on, etc.) and their network metrics (such as the average delay of their packets and the average amount of transmitted packets), along with their location as depicted on figure 4.1;

2. The controller uses the mobility data to determine the mobility profile of users. Users with the appropriate profile are pre-selected and tagged as potential relays;

3. The controller uses the network metrics to find the most critical locations. A critical location is one where the values of metrics are considered as too high;

4. Relays are elected in priority at critical locations. Then, as long as there are locations considered as critical other relays are selected as well.

Figure 4.2 is an example of the topology once the algorithm has been run and relays have been elected. Relays are users directly connected to the antenna. As a reminder, there were in this case four users with Internet access (see figure 4.1), labeled from 1 to 4 on figure 4.2: bicycle user 1 and pedestrian user 2 are the elected MRs. Pedestrian user 3 was not elected because her radio conditions are too bad (she is located inside a building). Car user 4 was not elected as a MR because he is a high-mobility user, not suitable for the stability of the topology. The resulting topology is a tree where relays act as root of the topology they manage (it is a RPL-based topology).

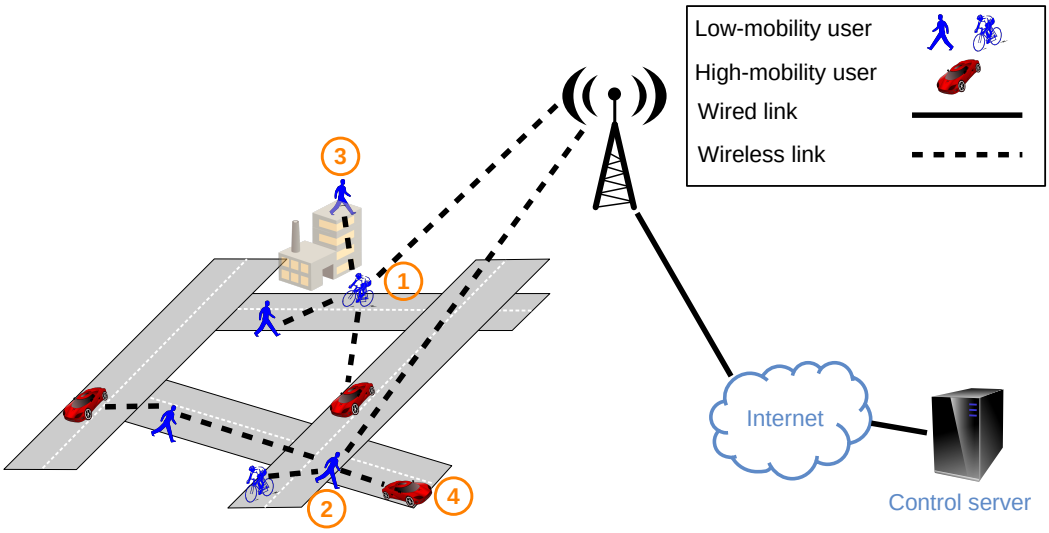


Figure 4.2: The established topology after a run of the proposed solution.

4.6 Classification of mobility and selection of relay

The algorithm uses machine learning to classify users according to their mobility profile. Then, it uses network metrics to find the best amongst users with the correct profile. The classification of mobility allows to separate high-mobility from soft-mobility users. A soft-mobility user is more suitable to serve as a relay because his lower velocity means the resulting topology will be more stable. Note that a user on a train is considered a high-mobility user because a train's speed is comparable to motorized vehicles. However, trains transit from one area to the next faster than road vehicles because they are not affected by traffic jams and intersections. Thus, they are not considered here.

Although the classification of mobility yields to better results compared to not classifying users, there are some shortcomings. For instance, if several users are suitable to act as relays it can be hard to select which ones will become relays if it is not desirable to elect too much relays (because of interference, for instance). The proposed solution allows to refine the selection by considering network metrics as well. It can be better to select a relay where the delay, for instance, is the lowest in order to improve QoS.

The proposed solution is made up of two principal parts:

1. The ML algorithm, used to classify users according to their mobility profile and
2. The users' network metrics to determine which specific users from the above step are the most appropriate.

For the first step, we used a decision tree algorithm to classify users. Decision trees are adapted because the decision at each branch can potentially lead to a leaf

(solution found). A user with a velocity of 100 km/h is certain to be a motorized vehicle, so no more computations are required if this condition is met. However, in the case of a heavy traffic jam cars can be harder to differentiate from bicycles and even pedestrians because of their low speed.

We define a network composed of U users. Each user u possess one vector of mobility-oriented metrics M:

$$M_u = [m_1, m_2, \ldots, m_\mu] \tag{4.1}$$

Where μ is the number of mobility-oriented metrics. These metrics can be the current or maximum velocity of the user, his location (i.e. the type of road he is on) and so on. The user periodically records these metrics. Each user u also stores one vector of network-oriented metrics N:

$$N_u = [n_1, n_2, \ldots, n_\nu] \tag{4.2}$$

Where ν is the number of network-oriented metrics the user is recording (such as delay, throughput, ETX and so on). The metrics are periodically recorded as well.

Mobility-oriented metrics (eq. 4.1) are used by the ML algorithm as features for the classification. The accuracy in determining the mobility type of the user (e.g. pedestrian, bicycle, car, etc.) depends on the choice of features and specific algorithm. We thus introduce the vector of features for each user u:

$$F_u = [f_1, f_2, \ldots, f_\phi] \quad \text{s.t.} \quad |F_u| > 0 \quad \forall u \quad \text{and} \quad \phi \leq \mu \tag{4.3}$$

That is, the features in equation 4.3 are a subset of the available mobility-oriented metrics from equation 4.1. The same ϕ features are used from each user and at least one feature must be used. These features are recorded for a length of time equal to τ. The purpose of τ is to choose the amount of time during which mobility-oriented metrics are gathered. Indeed, gathering metrics for a longer amount of time tends to increase accuracy, but it means waiting more before establishing the profile of the user. This can lower QoS, especially in time-sensitive applications so a compromise must be found between accuracy and the delay before predicting the class of a user. The mobility class predicted by the algorithm for each user is thus:

$$c_u^p = L(F_u, \tau) \tag{4.4}$$

In equation 4.4, L is the selected machine learning algorithm, F_u are the features of user u (see eq. 4.3) and $c_u^p \in C$ is the predicted class of user u. Given the prediction is not always correct, we introduce the real class of user u, $c_u^r \in C$, such that:

$$c_u^p = \begin{cases} c_u^r & \text{if the prediction is correct} \\ c^w \in (C \setminus c_u^r) & \text{if the prediction is wrong} \end{cases} \tag{4.5}$$

If the prediction is correct, the predicted and real class of user u are the same. On the other hand, a failed prediction will classify the user in the wrong class c^w which is any class other than the real one. The best mobility class C_b is determined as the most appropriate class to serve as relay. For instance, a slow-mobility class (e.g. pedestrian) can be desirable to increase topology stability.

To act as relay, a user must have access to Internet (for instance via 5G). We define $I \subseteq U$ the set of users with Internet access. Thus, only users within the

correct mobility class and with access to Internet can potentially be selected as relays:

$$R = I \cap C_b \tag{4.6}$$

Where R is the set of potential relays. Once the mobility class of users and the set of potential relays are determined, the network-oriented metrics (from eq. 4.2) are used to find the best location for electing a relay. First, we divide the studied area into X per Y tiles. The dimensions of each tile can be as small as to have one user maximum per tile (e.g. 1 m $\cdot$ 1 m) or as big as to have only one tile for the whole area and for all users:

$$T_{Y,X} = \begin{bmatrix} t_{1,1} & t_{1,2} & \cdots & t_{1,X} \\ t_{2,1} & t_{2,2} & \cdots & t_{2,X} \\ \vdots & \vdots & \ddots & \vdots \\ t_{Y,1} & t_{Y,2} & \cdots & t_{Y,X} \end{bmatrix} \tag{4.7}$$

Where each $t_{y,x}$ is one tile. We now define a scoring function S to compute the score of each tile given the values of network-oriented metrics of users:

$$t_{y,x} = S(N_u) \quad \forall u \in t_{y,x} \tag{4.8}$$

The purpose of tiles is to identify the most critical locations (e.g. tiles with high delay, high ETX, low throughput, etc.) to elect one or more relays inside them. The vector of the most critical locations is introduced:

$$T^{crit} \subseteq T_{Y,X} \quad \text{s.t.} \quad |T^{crit}| > 0 \tag{4.9}$$

That is, the vector of the most critical locations (eq. 4.9) is a subset of the matrix of scores (eq. 4.7). A location can be considered critical when QoS requirements are not met, for instance. Finally, the set of selected relays R_s is computed once these locations are determined:

$$R_s \subseteq (R \cup T^{crit}) \tag{4.10}$$

That is, a user will serve as relay if it is located in any critical cell (see eq. 4.9), has Internet access (eq. 4.6) and is of the correct (best) mobility class (eq. 4.4 and 4.5).

Note that in the case where a critical tile has no user that can serve as relay, these users will eventually be connected to a neighboring tile. This neighboring tile will become critical and relays will be elected to address the issue.

Algorithm 2 shows the relay selection process. It is run whenever the maximum number of relays is not reached, for instance when a MR has disconnected due to low battery life or because it does not need to be connected to the network any longer. The parameter of the algorithm (*arg[0]*) is the maximum number of relays. Lines 1 to 3 are the initialization steps. The gathering of users' metrics is done in lines 4 to 6 (eq. 4.1 and 4.2). In lines 7 to 11, the classification algorithm is run for each users to determine if its mobility profile is suitable to serve as a MR (eq. 4.4 and 4.5). If so, it is added to the set of potential relays (line 9). Then, in lines 12 through 19, the score of the potential relays is computed to find the ones that will best increase performance of the network (eq. 4.9). The score is computed in line 13 (eq. 4.8) and, if the user is satisfying, it is added to the set of relays in line 14 (eq. 4.10). The adding of relays can be interrupted if enough relays are present (line 16). Finally, the set of relays is returned in line 20. Note that a user must be in the correct location to become a potential relay.

Algorithm 2 The selection relay algorithm.

```
1: maxRelays ← arg[0]
2: potentialRelays ← NULL
3: relays ← NULL
4: for all users do
5:    users.gatherMetrics()
6: end for
7: for all users do
8:    if A(user) == optimal then
9:       potentialRelays.append(user)
10:   end if
11: end for
12: for i = 0   to   potentialRelays.size() do
13:   if computeScore(potentialRelay_i) > criticalScore then
14:      relays.append(potentialRelay_i)
15:   end if
16:   if relays.size() ≥ maxRelays then
17:      break
18:   end if
19: end for
20:
21: return relays
```

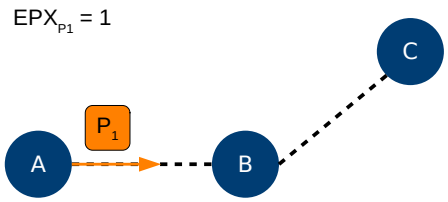
4.7 New metric: Expected Packet Count

The Expected Packet Count (EPX) is a new metric that represents the total amount of (re-)transmissions needed for packets sent from a location to reach their destination during a certain time frame. Each hop in the path of these packets counts towards EPX including re-transmissions. It is the "density of packets" of a given location. A high EPX is the consequence of long paths to reach a relay, bad radio conditions (many re-transmissions) and/or simply high traffic.

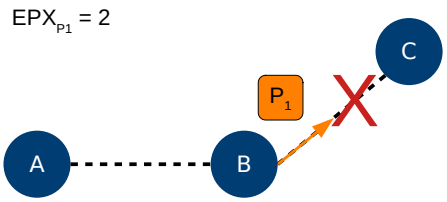
Figure 4.3 shows a simple example of the computation of EPX. One packet is emitted from node A and its destination is the sink node C. First, when the packet is about to be sent after its creation, the value of EPX is incremented to 1 (fig. 4.3a). The location of the packet's source is kept in the packet. When the packet is received on B, its EPX value is incremented as it is about to be forwarded to C (fig. 4.3b). The transmission fails due to interference and is not acknowledged by C. B increments the EPX value again and resends the packet (fig. 4.3c) to C. The value of EPX is now 3. Finally, C successfully receives the packet (fig. 4.3d). Note that the value of EPX from P_1 concerns the area where the source (here, A) is located, not the source itself. Thus, if a sink receives multiple packets from a same location, the total EPX is the sum of all EPX values from all packets from this area.

4.8 Comparison of different ML classification algorithms

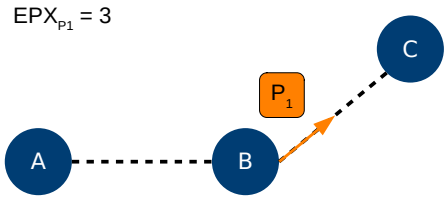
Different classification algorithms are presented in this section. They are tested and table 4.1 shows their accuracy. The ML models are trained using a real map different from the one used to test the model (i.e. run the simulation) to assess the



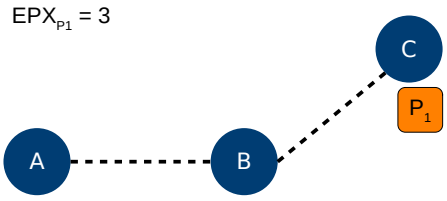
(a) As A is about to transmit P_1, the ETX is incremented.



(b) B received the packet and tries to send it to C, thus incrementing the EPX. The transmission fails.



(c) B retries to send P_1 to C. EPX is incremented to 3.



(d) The packet reaches its final destination, C. The packet reads an EPX of 3 and originates from A.

Figure 4.3: An example of the computation of EPX: packet P_1 is emitted from A and has destination sink C. The EPX, stored in P_1, concerns the area where A is located.

		Classification delay (in s)						
		2	4	6	8	10	12	14
K-nearest	Training	98	98	98	98	98	98	98
	Testing	65	84	87	80	85	84	85
SVM	Training	94	94	94	94	94	94	94
	Testing	57	70	77	77	70	77	77
Decision tree	Training	99	99	99	99	99	99	99
	Testing	70	84	94	92	94	95	94

Table 4.1: The precision in classifying the correct type of mobility against the allowed delay. The table shows the values for the training and testing sets.

generalization capabilities of the studied algorithm.

The accuracy of the algorithms depends on the delay allowed to gather data. When the delay is very short (2 s for instance), the error is rather high because the user was not active long enough. For example, a car just starting might stop at a red light right away and it is not possible to differentiate it from a bicycle on such a short notice. However, although a longer delay allows for better accuracy, it also means a longer time before selecting a new sink. During this time, no connectivity will degrade QoS and might not be the right choice. This is why a compromise must be found between accuracy and delay to yield the best QoS possible.

Table 4.1 shows various values of accuracy for three different algorithms: K-nearest neighbors, Support Vector Machine (SVM) and decision tree. For each

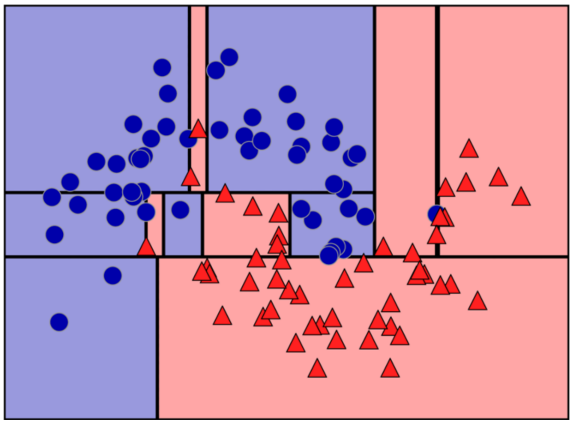


Figure 4.4: An example of a decision tree deduced from supervised learning. The image comes from [4], p.76. Blue circles and red triangles represent 2 different classes of objects.

algorithm, two lines show the accuracy in classifying users as pedestrian, bicycle or car for the training and testing sets. Decision tree yields the best accuracy with 99% for the training set and ranging from 70% to 94% during testing. K-nearest is the second most accurate with a training accuracy of 98% and a testing accuracy from 65% (2 s) to 85% (10 s and 14 s). SVM is the least accurate with correct classification in 94% of cases for the training set and between 70% and 94% when testing.

For the longest delays, the accuracy does not improve significantly and stays stable (with small variations). This means a very long delay is not necessarily better.

As mentioned at the beginning of section 4.6, decision trees work well in this case because the algorithm can split the classes into "tile" regions if one criterion is met. Figure 4.4 shows such an example. For instance, if the speed of a user is 100 km/h, it is most certainly a motorized vehicle and the algorithm can stop.

4.9 Performance evaluation

OMNeT++ and SUMO are used for the performance evaluation. The results show two batches of scenarios. The first batch is composed of two scenarios: one with a dynamic selection of relays amongst normal users and the other without a selection process, where relays are randomly placed on the studied area. The goal of the first batch is to compare the selecting of relays based on the mobility type of users to a random selection of relays. The scenario with static relays is used to compare the performance of a dynamic selection of relays with a static dedicated infrastructure.

The second batch goes one step further. The dynamic selection of relays is compared to a random selection (like in the first batch of scenarios) but it is also compared to the dynamic selection of relays with the consideration of localized network-oriented metrics to refine the selection of relays. In this batch, two scenarios are studied: one with a low traffic density and one with a heavy traffic density.

The remainder of this section presents the first batch of scenarios and their associated results (comparison of random selection of relays, dynamic selection of relays and static relays) in subsection 4.9.1 and the second batch of scenarios

Delay (s)	2	4	6	8	10
Accuracy (%)	20	27	88	79	86

Table 4.2: The accuracy of the predictive algorithm given the delay to gather data.

Parameter	Value
Playground size	1 km^2
User generation period	1.5 s
Num. of users, total	±4800
Num. of users, inst.	10's to 100's
Num. of relays†	5
Car:bike ratio	9:1
Simulation duration	8000 s
Traffic model	VBR
Traffic density	10 packets/s
Data packet size	1280 bytes
Max. user throughput	10 Mbps

Table 4.3: Values for the most relevant parameters of the simulations. †Number of relays is set for the predictive case and varies from 2 to 7 in the case of static dedicated relays.

(comparison of the random, dynamic and dynamic with metrics selection of relays in light and heavy traffic cases) is depicted in subsection 4.9.2.

4.9.1 Dynamic selection of relays

4.9.1.1 Simulation setup

The simulation is run using OMNeT++ 5.4.1 and SUMO 0.32.0. The Veins framework interfaces OMNeT with SUMO. A real map from OpenStreetMap[1] is imported into SUMO and realistic urban traffic is generated with a tool from the SUMO package. As a remainder (see section 3.8.3), Veins allows to run an OMNeT simulation using the mobility patterns of users (cars, bicycles) generated from SUMO to create and move nodes in real-time. That is, each user in SUMO has a corresponding entity in OMNeT (a "node" in this case). Thus, OMNeT creates one "node" from each user in SUMO and destroys them when they reach their final destination.

Table 4.2 shows the accuracy of the decision tree in correctly predicting the mobility class of users. A good prediction means a car, for instance, is predicted as a car. A wrong prediction means a bicycle, for instance, is predicted as something different from a bicycle. The "Delay" column in table 4.2 corresponds to the time during which data is gathered from users before making the prediction (see section 4.8).

The predictive algorithm is compared to a random selection of sinks. The random selection serves as a base case and has the advantage of a zero delay to choose a new sink. The random choice is not represented in table 4.2. The simulation is also run without prediction using static relays (from 2 to 7) to compare the ML algorithm with a solution using a dedicated static infrastructure. Relays are placed randomly

[1]The real map is a sub-area of the *Saint-Laurent* borough, Montreal, Canada.

on the simulated area. The large studied area compared to the total coverage area of relays is deliberate to study the impact of the choices from the different algorithms.

All users are supposed to be able to connect to Internet (directly via cellular network or not) to send their mobility-oriented data to a control server running the ML algorithm. Users without cellular connectivity will connect to other users and eventually transmit these data to the controller.

4.9.1.2 Studied performance indices

This subsection presents the different PIs for the first batch of scenario. The structure is similar to subsection 3.8.1.

Packet delivery ratio The packet delivery ratio (PDR) represents the percentage of generated data packets that reach their final destination. Some factors such as frequent disconnections of nodes from their neighbors or very bad radio conditions (causing the amount of re-transmissions exceeding the maximum number of retries) will cause PDR to decrease. Note that even if packets are being re-transmitted many times on each hop of a path, PDR can still be high, though other metrics such as delay will increase. The higher the PDR the better.

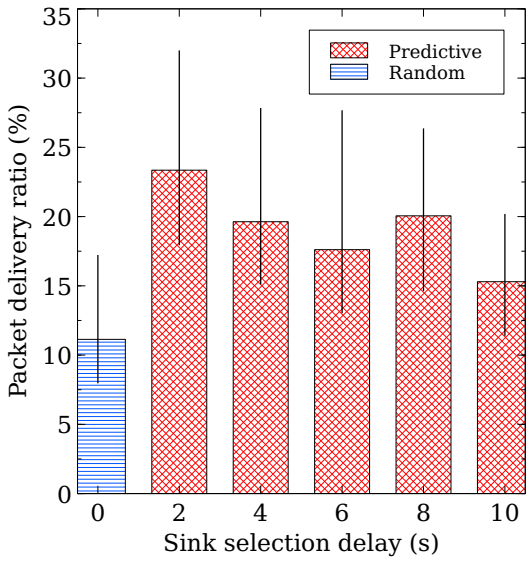
Connectivity ratio The connectivity ratio is the percentage of time a user is connected to a sink during her trip. For instance, a connectivity of 25% on a 8000 s run means the user was connected during 2000 s in total. Connectivity is linked to PDR, as more connectivity increases the chance for packets to reach their destination though it is not necessarily the case, as a non-connected user will not try to send packets. This metric allows to see the coverage of the studied area and if relays are well placed. A higher connectivity ratio is better.

Amount of signaling The amount of signaling represents the number of packets sent on the topology excluding data and acknowledgment packets. Signaling is mostly used to set up and maintain the topology. Sending a signaling packet again because it was not acknowledged counts in the amount of signaling. This metric shows how much overhead is generated and how good transmissions are. A lower amount of signaling is better.

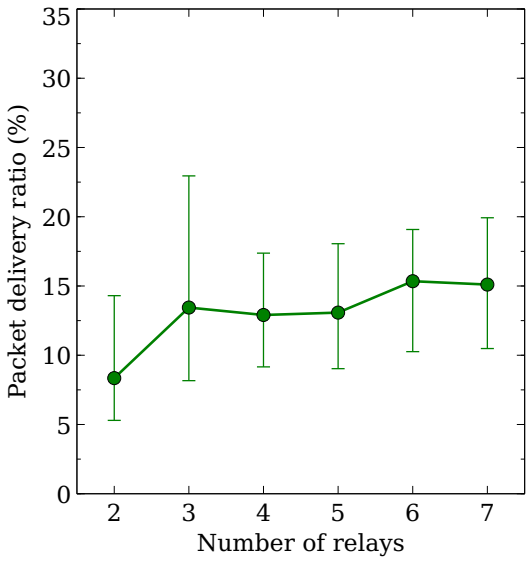
Amount of sink change When dynamic selection of sinks happens, this metric shows how many times sinks are replaced. When an elected sink leaves the area of interest or fails due to low battery, a new one will be chosen. This metric shows the stability of the topology. A lower number of changes is better.

4.9.1.3 Result analysis

In the following, bar charts (figs. 4.5a, 4.6a, 4.7a and 4.8) represent results using dynamic relay selection amongst users. The horizontal axis is the time taken to make the prediction. The leftmost bar is the random selection (selection delay of 0 s) and other bars represent the ML algorithm (selection delay of 2 s to 10 s). The line plots (figs. 4.5b, 4.6b and 4.7b) represent results without prediction using static dedicated relays. The horizontal axis is the number of relays placed on the area. As a remainder, simulations with dynamic relay selection are achieved using 5 relays (see table 4.3).



(a) The PDR given the allowed prediction time.



(b) The PDR in the case of static relays.

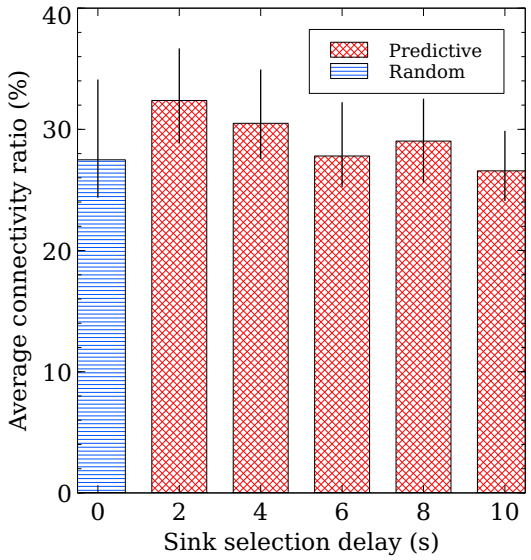
Figure 4.5: The average PDR of the data packets of users.

Packet Delivery Ratio Figure 4.5 shows PDR results of the different simulations. The random selection (fig. 4.5a) yields a PDR of about 11%. Considering error bars, the proposed solution performs significantly better when the selection delay is 2s (fig. 4.5a) because the selection of soft-mobility users as relays allows the topology to be more stable, reducing the amount of packet loss. PDR is potentially higher when using ML and tends to decrease as the selection delay increases because waiting longer to select a new sink causes data packets to be rerouted after a longer delay, increasing the probability of packet loss.

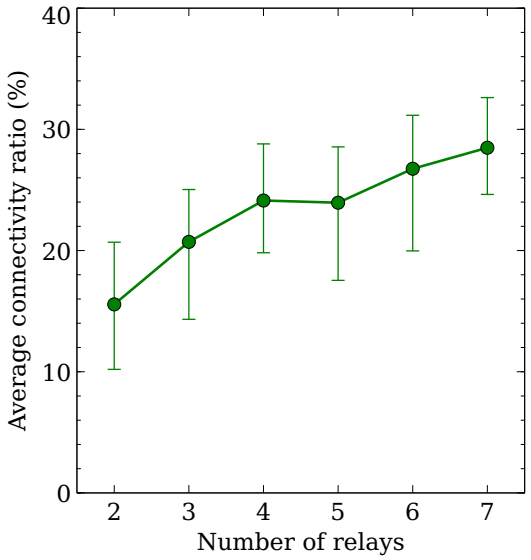
On the other hand, static relays (fig. 4.5b) have lower PDR than the predictive solution because users move around and those relays eventually service no users at all. Those results are comparable to the random selection. When 2 relays are used, results from figure 4.5b show a PDR significantly lower than the proposed solution with a delay of 2 s and 4 s: static relays have a limited coverage, so only 2 yields low performance. More relays, in this case, help to increase performance, as shown on figure 4.5b. Note that on figure 4.5a the lowest PDR value with a selection time of 2 s is 17.9865 % whereas on figure 4.5b the highest PDR value with 5 relays is 18.0529 %, so there is a slight overlap.

Connectivity ratio The connectivity ratio is depicted on figure 4.6. The proposed solution is comparable to the random selection (fig. 4.6a) with a connectivity of about 30 %. Indeed, mobile relays are located such that there is similar connectivity no matter the selection method. However, with a prediction time of 2 s or 4 s, the proposed solution performs significantly better when compared to static relays (fig. 4.6b) when using 2 or 3 relays. This is caused by the fact static relays (fig. 4.6b) eventually spend some time not being in range of users. A longer selection delay will negatively impact the predictive algorithm because users will spend more time disconnected.

Amount of signaling We can see on figure 4.7 the random selection performs significantly better (about 1.5 packets per minute per user on average) than the

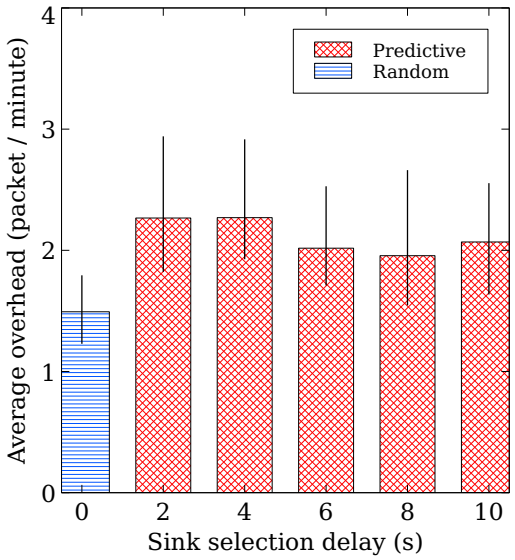


(a) The connectivity given the allowed prediction time.

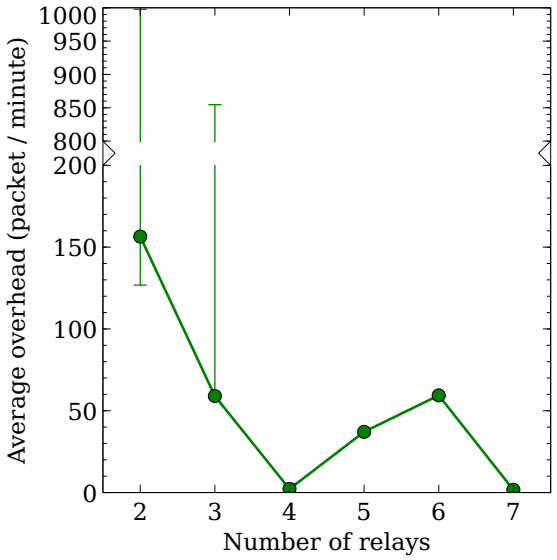


(b) The connectivity when using static relays.

Figure 4.6: The average connectivity of users during their trip.



(a) The amount of signaling given the allowed prediction time.



(b) The amount of signaling in the case of static relays. Note the cut on the vertical axis.

Figure 4.7: Study of the amount of signaling packets (excluding acknowledgments) in two different scenarios. Note vertical axes have different scales.

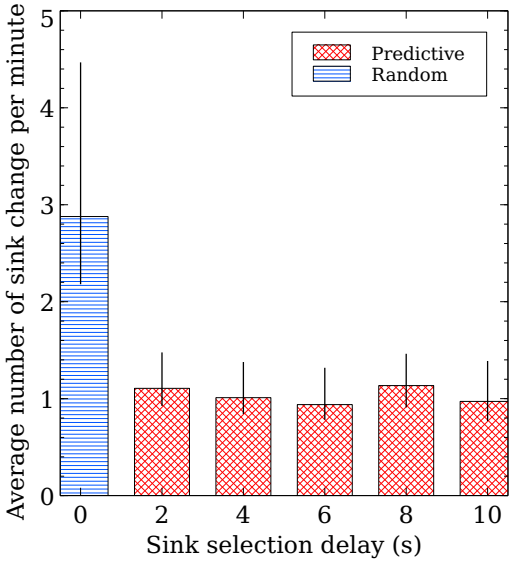


Figure 4.8: Stability of the topology in terms of the rate of sink change.

proposed solution (around 2.25 packets per minute per user on average) when the selection delay is 2 s or 4 s (see fig. 4.7a). The lower overhead of the random selection is due to lower performance in terms of PDR (fig. 4.5) and connectivity (fig. 4.6). Indeed, less signaling will be sent in absolute if users are less connected to one another.

In figure 4.7b, the amount of signaling is very high when there is (are) 1 or 2 relay(s). This is due to low coverage causing long paths (i.e. high number of hops from any user to the sink) with higher probability of re-transmissions. Long paths also increase the probability of any user moving away causing the need to repair the topology. Transmission distances are longer as well because relays are static, causing many losses. The proposed solution is significantly better than the case of static relays (fig. 4.7b) except when the number of relays is 4 or 7, with overheads of about 2.2 packets per minute and 1.7 packets per minute respectively. A high number of overhead does not necessarily means performance is bad (see PDR on fig. 4.5). However, the scaling will be problematic if overhead is very high, as is the case of 1 and 2 relay(s) (fig. 4.7b).

Number of sink change Figure 4.8 shows the predictive algorithm performs significantly better in all instances. The random selection process causes more changes of sink. This is due to selecting high-mobility users (cars) more often that leave the area of interest quicker, so a change is required. The proposed solution yields more stability as the number of changes is below 1.5 per minute on average. This means that one new sink is chosen approximately every 40 s on average, compared to the random case (around 3 changes per minute) where one sink is chosen every 20 s on average.

4.9.2 Dynamic selection of relays with metrics

The evaluation of performance is done by comparing three different algorithms. Two scenarios are studied and two parameters vary (thus, there are 4 series of graphs). The two first graphs of each series (figures *a* and *b* in section 4.9.2.3) represent the scenario with a low density of users (column *Low density scenario* in table 4.4). The third and fourth graphs of each series (figures *c* and *d* in section 4.9.2.3) refer

to the scenario with a high density of users (column *High density scenario* in table 4.4). The line charts depict the variation of traffic generated by users, whereas the bar charts represent the change of selection delay (that is, the time during which network-oriented metrics are gathered for the purpose of classifying the mobility-type of users). The studied algorithms are:

1. *Without classification* does not use any classification. Relays are selected randomly. In line charts it is represented by red circles. In bar charts it is represented as red-crossed bars with a selection delay of 0 s;

2. *With classification* uses classification of mobility profile to select a new relay. Relays with low mobility are favored. In graphs, this algorithm is represented by blue diamonds (line) or blue-lined bars (bar charts);

3. *With classification and metrics* uses mobility classification (as *With classification*) and a score function (see sec. 4.6). It is represented in line charts as green squares and in bar charts as green-filled bars.

Given the classification requires some time to gather data to determine the mobility profile of users, the selection delay of the algorithm without classification is always 0 s (no delay). The score function (see eq. 4.10) for the solution using classification with metrics combines the delay of tiles with the EPX (defined in section 4.7).

4.9.2.1 Simulation setup

Simulations are done using OMNeT++ 5.6.1. Users' trip are generated using SUMO (Simulation of Urban MObility) 1.3.1. The studied area is the location of the University Gustave-Eiffel (France) and was downloaded from OpenStreetMap (see figures A.3 and A.4). Users follow roads from the downloaded OpenStreeMap area (sidewalks in the case of pedestrians) and obey traffic rules. Once the trips of users have been created in SUMO, they are exported to a XML file. Then, the OMNeT simulation will create one "module" (or node) for each user that was generated by SUMO. To train the ML algorithm doing the classification, another simulation is run on a different location, allowing to test the generalization of the algorithm. The delay of the prediction allows to gather more or less data from users. A lower delay is less accurate (about 70% accuracy for 2 s) because fewer data from a user's position, acceleration, speed and so on are accumulated. On the other hand, waiting a longer period of time to make the prediction means a higher precision (about 94% for 6 s) but users are not connected during that delay, so a compromise has to be found.

The list of the most important parameters is presented in table 4.4. The size of the area is about 1.96 km^2. Users are created on the network each 3 s (low density case) or each 0.75 s (high density case). The total and instantaneous number of users represent, respectively, the amount of users created on one run and how much of them there are at any time. About 5% of the instantaneous number of users are elected as relays. Most of the users are in cars and the smallest part of users are bicycles. Simulations are 1000 s in length with 10 repetitions to increase results accuracy. All users except relays generate data following a Variable Bit Rate (VBR). Data packets are set to the minimum size of an IPv6 packet [18] and users have a maximum throughput of about 10 Mbps [8]. The maximum range of users is about

Parameter Value	Low density scenario	High density scenario
Playground size	1400 m · 1400 m	1400 m · 1400 m
User generation period	≈ 3 s	≈ 0.75 s
Num. of users, total	≈ 330	≈ 1250
Num. of users, inst.	30-50	100-150
Ratio of relays	max. 5 %	max. 5 %
Car:bike:pedestrian ratio	3: 1: 1.25	2.75: 1: 1.5
Simulation duration	1000 s	1000 s
Repetitions	10	10
Traffic model	VBR	VBR
Traffic density†	10 to 15 packets/s	10 to 15 packets/s
Relay selection delay†	[0, 2, 4, 6] s	[0, 2, 4, 6] s
Data packet size	1280 bytes	1280 bytes
Max. user throughput	10 Mbps	10 Mbps

Table 4.4: Values for the most relevant parameters. †The variable parameter is either traffic density (from 10 to 15 packets/s) or the relay selection delay (from 2 to 6 s). The solution without classification always uses 0 s).

105-110 m with a path loss based on a $-log_{10}(d)$ (d is the distance) model [123] [124].

4.9.2.2 Studied performance indices

Packet delivery ratio As explained in section 4.9.1.2, the PDR is the amount of data packets that reach the destination divided by the amount of data packets sent. The higher the PDR the better.

End delay The end delay is the time taken for a packet to reach its final destination. It depends on how good radio conditions are (many re-transmissions will increase delay) and on the amount of congestion (buffer occupancy). A low throughput means nodes require more time to send packets, increasing delay. The amount of nodes a packet has to go through on its path (hop count) does not necessarily increase delay, as a long path composed of nodes with high throughput and low congestion will perform better than a short congested path. The lower the delay the better.

Energy consumption The energy consumption per node represents the amount of energy spent by each node on transmitting and receiving packets, including signaling and data packets. Bad radio conditions resulting in many re-transmissions and topology instability causing more signaling are examples of causes increasing energy consumption. The lower the consumption the better.

Throughput usage The throughput usage represents the percentage of the total throughput a node can achieve that is in use. For instance, if a node can achieve a maximum throughput of 10 Mbps, a throughput usage of 20% means the node sends on average 2 Mbps worth of packets. This includes signaling and data packets. A lower value is achieved when good radio conditions (fewer re-transmissions) are present. More stability in the topology will also lower the amount of signaling and the usage of throughput. The lower the throughput usage the better.

Relative amount of packets reaching destination PDR, end delay and throughput usage are not enough to see how well the different solutions perform. Indeed, if a solution generates a very low amount of packets due to lack of connectivity, the PDR of these packets can still be very high, their delay very short and the usage of throughput will be low as well. The relative amount of packets reaching destination is the number of packets that reached the final destination compared to the total amount of packets created. It is the PDR times the total amount of generated packets and represents how well nodes are connected to a relay. The higher the better.

4.9.2.3 Simulation results

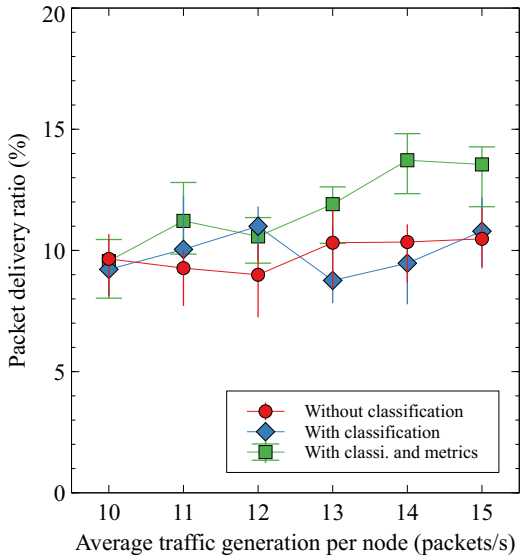
The following results are regrouped by the studied metric. In line charts (where traffic load varies), the selection delay is set to 2 s. In bar charts (representing the variation of selection delay), it is the traffic load that is set to 15 packets/s.

Packet delivery ratio Figure 4.9 shows the PDR of the different algorithms. When studying the impact of traffic in the low density scenario, (fig. 4.9a), we can see the proposed solution performs significantly better at higher loads. This is because taking into account network metrics allows the solution to adapt the location of relays where the need is greater. On the other hand, at lower traffic loads (figs. 4.9a) all solutions perform similarly. In the case of a higher density of users (figs. 4.9c and 4.9d), the proposed solution almost always performs significantly better because of more traffic generated in absolute. Indeed, in the case of a higher density of traffic, the use of network metrics allows to select relays in the most critical locations.

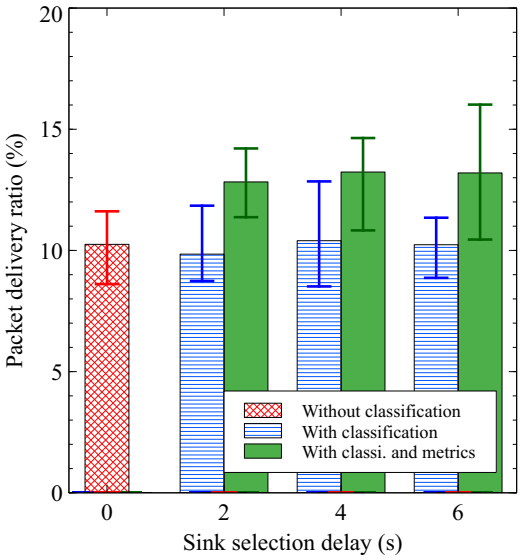
Algorithm *with classification* has similar results compared to *without classification* on figures 4.9a and 4.9c because the low amount of relays (5%, see table 4.4) and the fact that users within the correct class of mobility are not differentiated from one another makes it harder to select a candidate in a critical location. The same is seen on figures 4.9b and 4.9d. The variance of the proposed solution increases with the selection delay in the low density scenario (fig. 4.9b) because the network metrics used for the scoring function are older (less accurate). This is not the case in the second scenario 4.9d because the higher density of users greatly increases the amount of network metrics and better locations are available to select a relay.

End delay Figure 4.10 shows the average end delay of data packets for the different solutions. It is interesting to see that although the PDR was not better in the low density scenario (fig. 4.9a) for the proposed solution, the end delay is better (fig. 4.10a). This is because even though there are fewer choices to select a potential relay, the network metrics still help in determining the most critical locations.

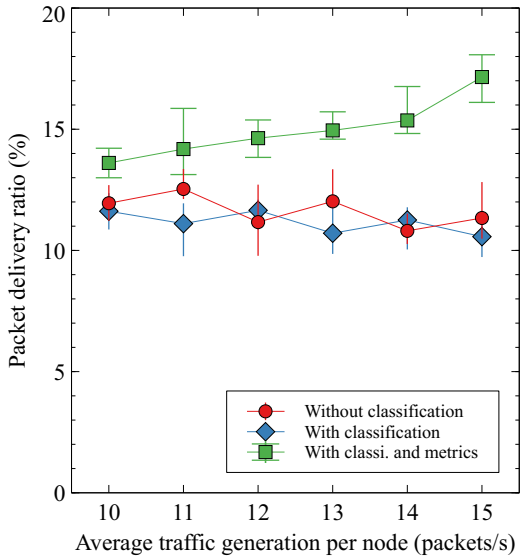
When studying the selection delay in the low density scenario (see fig. 4.10b), the proposed solution is better, except when the selection delay is 4 s. As stated at the beginning of section 4.9.2.1, this is due to the fact an average selection delay will decrease the accuracy of the network metrics but it will only increase the classification accuracy by a moderate amount. On the other hand, either a short selection delay (less accurate network metrics and shorter waiting time) or a longer selection delay (much more accurate classification) are preferable. For the high density scenario (figs. 4.10c and 4.10d), the delay is comparable to the two other algorithms. In this case, the high amount of users makes the delay and EPX



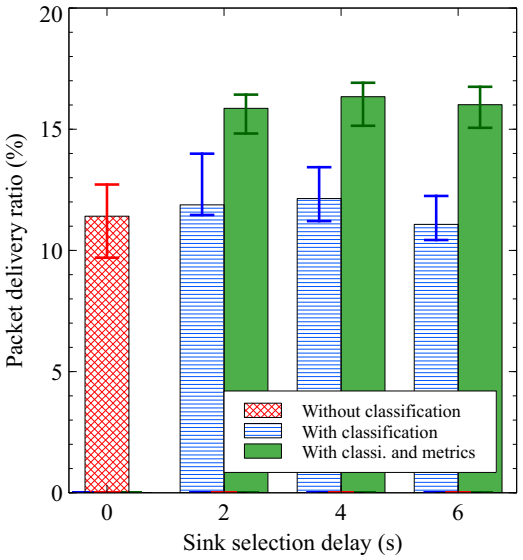
(a) The impact of the average network load in the low density scenario.



(b) The impact of the selection delay in the low density scenario.

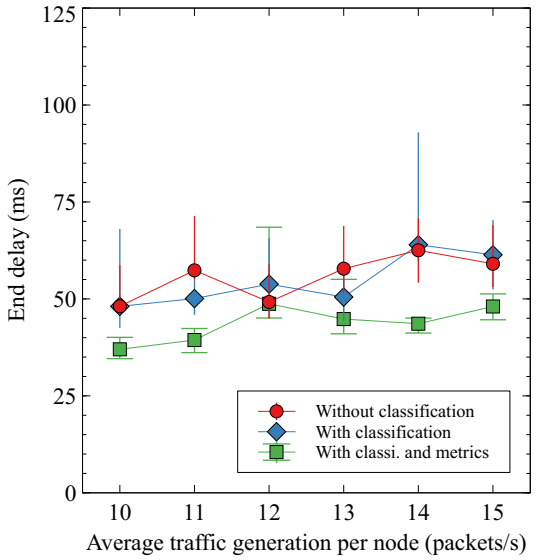


(c) The impact of the average network load in the high density scenario.

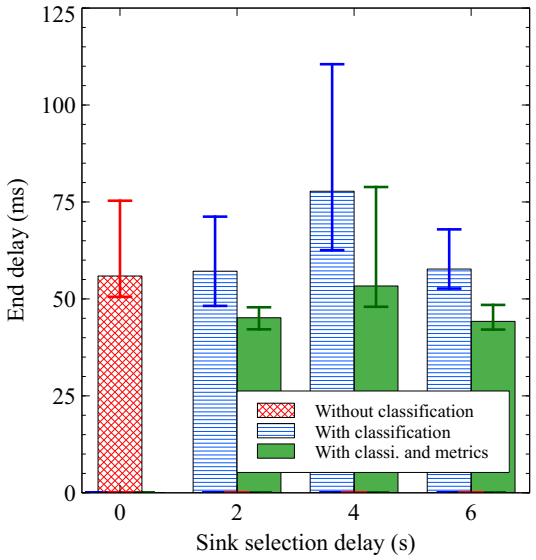


(d) The impact of the selection delay in the high density scenario.

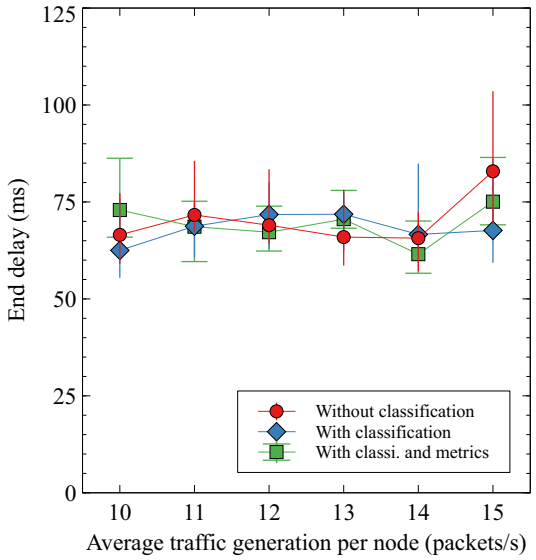
Figure 4.9: The packet delivery ratio of the different scenarios. The vertical axis has the same scale for all 4 graphs.



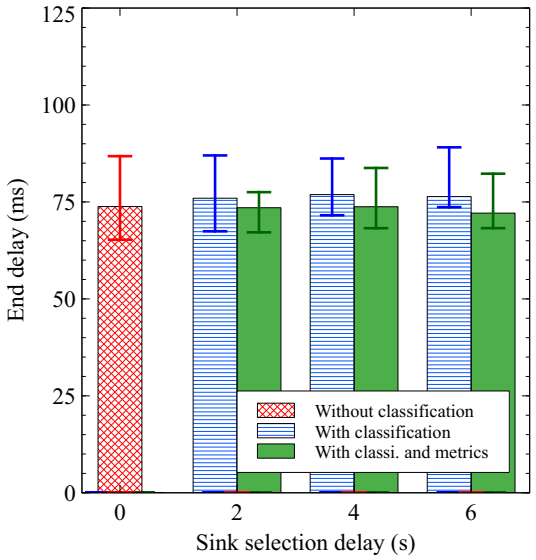
(a) The impact of the average network load in the low density scenario.



(b) The impact of the selection delay in the low density scenario.

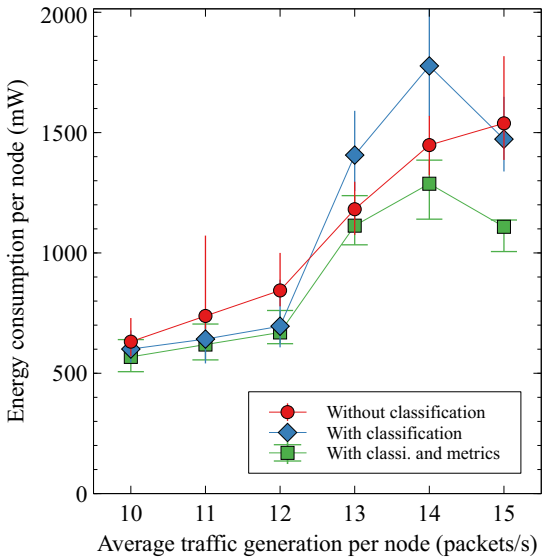


(c) The impact of the average network load in the high density scenario.

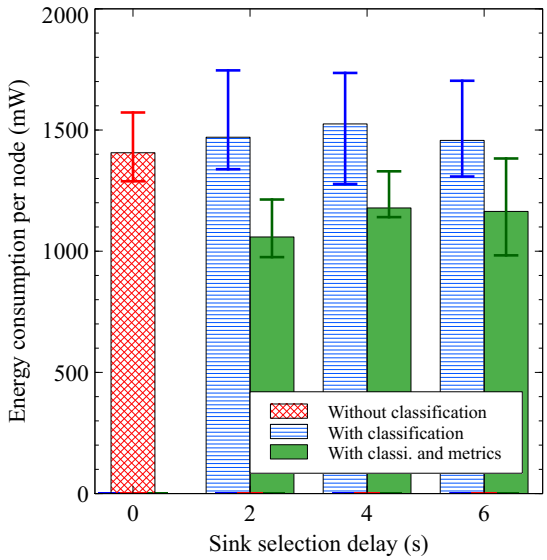


(d) The impact of the selection delay in the high density scenario.

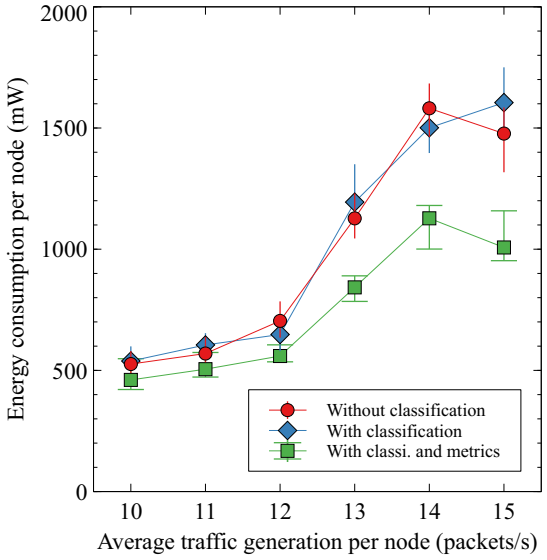
Figure 4.10: The average end delay results of the different scenarios. The vertical axis also has the same scale on all graphs.



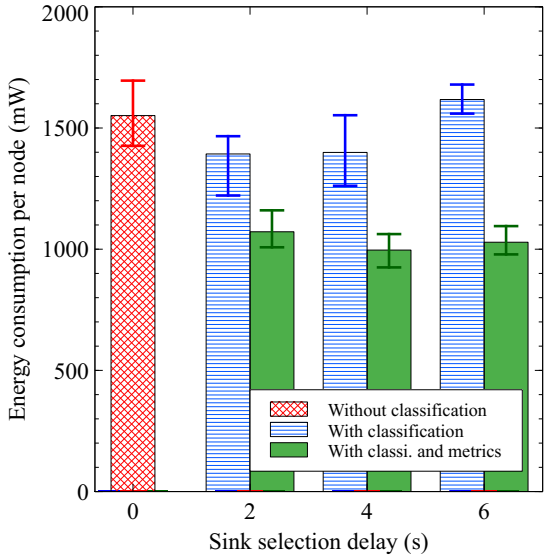
(a) The impact of the average network load in the low density scenario.



(b) The impact of the selection delay in the low density scenario.



(c) The impact of the average network load in the high density scenario.



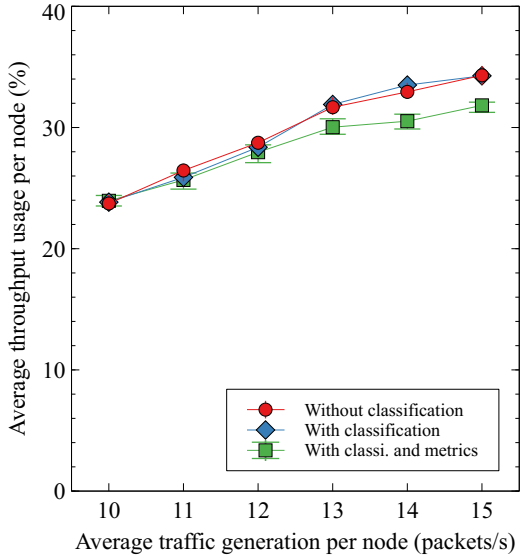
(d) The impact of the selection delay in the high density scenario.

Figure 4.11: The average energy consumption per node in the different scenarios.

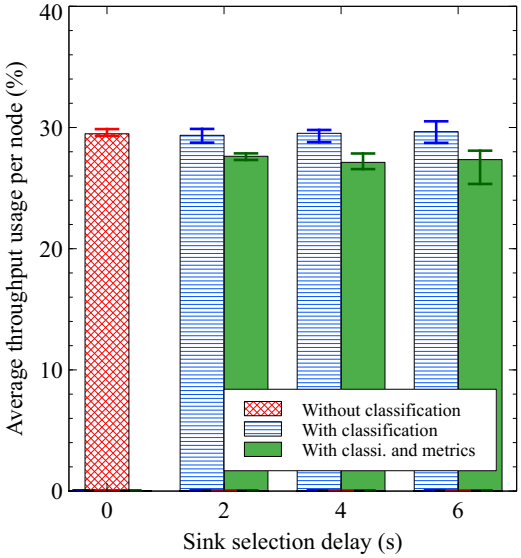
increase in all locations. The other solutions perform similarly (figs. 4.10a, 4.10b, 4.10c and 4.10d) for the same reasons mentioned before.

Energy consumption The average energy consumption per user is depicted on figure 4.11. In all scenarios (figs. 4.11a, 4.11b, 4.11c and 4.11d), the proposed solution consumes less energy. The higher PDR (fig. 4.9) and the similar or lower end delay (fig. 4.10) show that there are less re-transmissions, resulting in a lower consumption of energy. It is worth noting that even though there are more users and packets generated on the high density scenario (see fig. 4.13), the consumption per user is not necessarily higher. This is due to the fact that more paths are available, so the PDR is higher (as depicted on fig. 4.9c) and there are less re-transmissions.

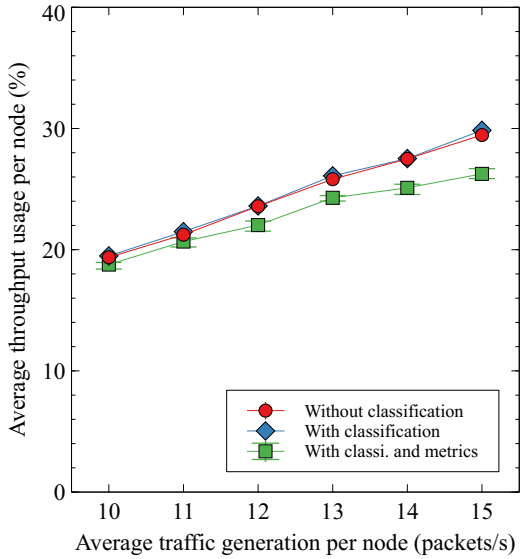
Throughput usage Figure 4.12 shows the average throughput usage per node for the different solutions. In the low density and high density scenarios (figs. 4.12a



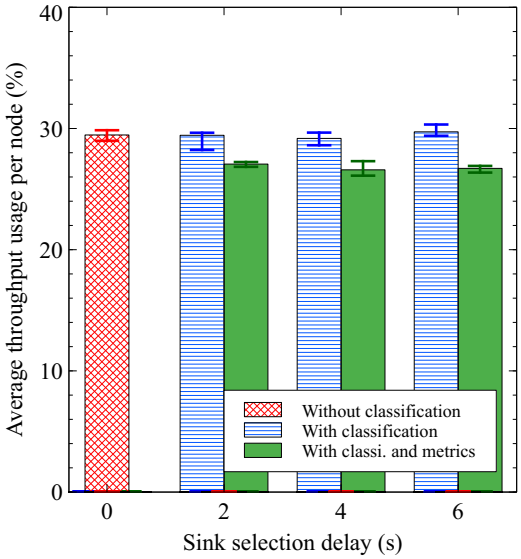
(a) The impact of the average network load in the low density scenario.



(b) The impact of the selection delay in the low density scenario.

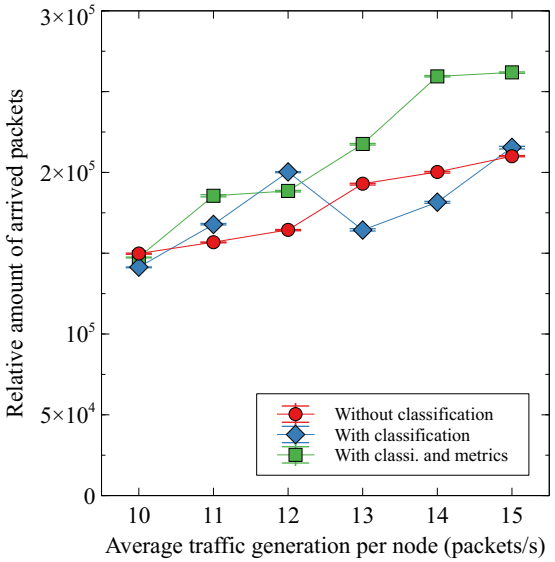


(c) The impact of the average network load in the high density scenario.

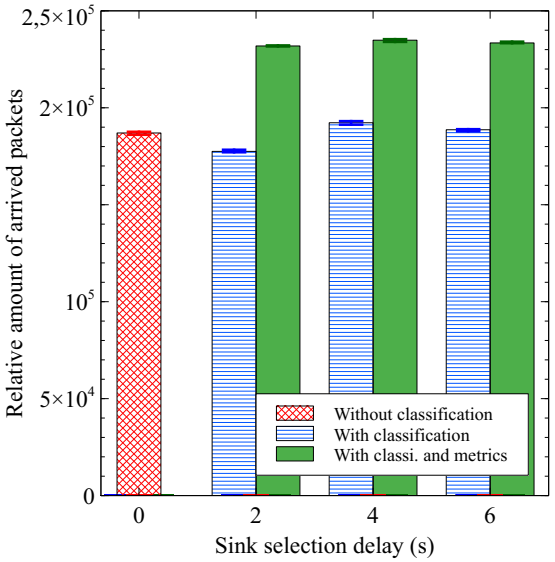


(d) The impact of the selection delay in the high density scenario.

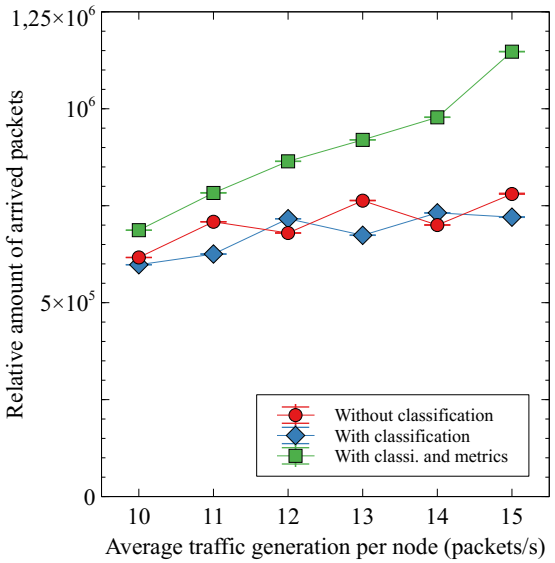
Figure 4.12: The average throughput usage in the different scenarios.



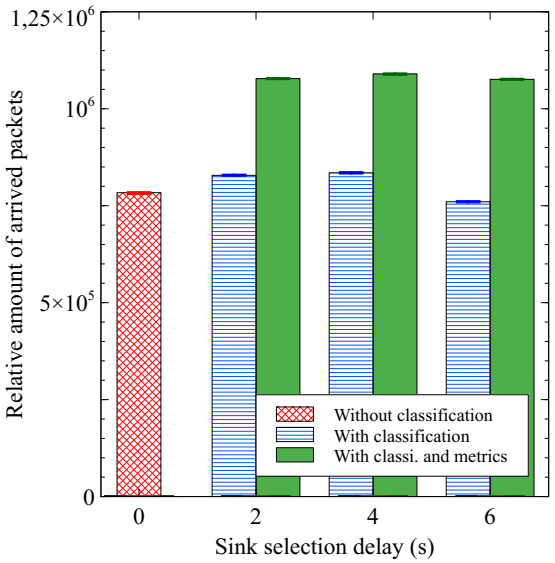
(a) The impact of the average network load in the low density scenario.



(b) The impact of the selection delay in the low density scenario.



(c) The impact of the average network load in the high density scenario.



(d) The impact of the selection delay in the high density scenario.

Figure 4.13: The relative amount of packets that reached their final destination.

and 4.12c), the usage increases with load for all algorithms. The usage is higher in the low density case compared to the high density case due to a lower PDR (fig. 4.9a compared to fig. 4.9c) and more re-transmissions. The selection delay (figs. 4.12b and 4.12d) does not impact much the throughput usage, though the variance increases slightly for the algorithm *with classification and metrics* in the low density case (fig. 4.12b) because of the decreased accuracy of the network metrics.

Relative amount of packets reaching destination The relative amount of packets reaching their final destination compared to the amount of generated packets is presented on figure 4.13. The proposed solution shows better results in the low density scenario at higher loads (fig. 4.13a) and in the high density scenario (fig. 4.13c). This follows the results on the PDR (fig. 4.9a and fig. 4.9c). The selection delay does not impact the relative amount of arrived packets for the proposed solution in both scenarios (figs. 4.13c and 4.13d).

4.10 Conclusion

In this section, we discussed about a ML-based solution to classify users according to their type of mobility with the aim of electing mobile relays amongst users. The proposed architecture is made up of a fog part and a cloud part. In the former, users with cellular access send their mobility-oriented data to the control server, located in the cloud. Then, the server runs the machine learning classification algorithm to deduce which users are the best to act as relays and elects them. Once selected as relays, these users start the building process of the topology to allow others to connect to Internet. As the network evolves with users traveling, connecting and disconnecting, the control server locates critical locations with the help of localized delay and the new metric EPX to elect more relays in those areas.

The OF presented in chapter 3 can be combined with the classification presented in this chapter. Indeed, once relays are selected by the controller, a programmable OF can be sent at the same time depending on local needs to build the topology. It is possible to go one step further by predicting the future position of devices and users, thanks to ML.

Chapter 5

Placement of relay drones based on the prediction of devices position

Résumé français

Ce dernier chapitre porte sur le placement d'un drone-relais pour donner aux appareils et utilisateurs une connexion vers Internet. Les drones peuvent être déployés en appoint pour gérer une situation normale mais exceptionnelle (par exemple, un festival de musique ou tout autre événement rassemblant un grand nombre de personnes) ou en cas de situation anormale (des dégâts sur l'infrastructure de télécommunication ou une panne électrique). Pour placer le relais à l'endroit optimal, nous faisons de nouveau appel à l'apprentissage artificiel, mais cette fois-ci pour prédire la position future des noeuds. Leur emplacement actuel et futur sont représentés par un score qui permet de déterminer l'endroit le plus intéressant où placer le drone.

Les solutions de l'état de l'art ne considèrent généralement pas plusieurs types de mobilité en même temps. Nous proposons un modèle où se trouve des noeuds pouvant se déplacer à vitesse faible (à l'arrêt ou à vitesse de marche), des vitesses intermédiaires (vélos ou trottinettes, voitures dans des zones d'écoliers) ou encore à vitesse plus élevée (typiquement 50 km/h, qui est la vitesse maximale autorisée en agglomération dans plusieurs pays). De plus, nous tenons compte non seulement de la position actuelle des noeuds pour placer le drone, mais aussi de leur position future.

L'architecture est composée d'appareils faisant partie du réseau local et qui ont besoin de se connecter à Internet. Le drone est positionné à proximité de ceux-ci (à portée de WiFi, par exemple) et est connecté à une antenne cellulaire. Une fois en place, il peut se déplacer au besoin.

Nous commençons d'abord par séparer la région d'intérêt en tuiles. La taille des tuiles va dépendre des besoins, mais ici nous avons évalué la solution proposée avec des tuiles carrées ayant des dimensions de l'ordre de quelques dizaines de mètres. La position des noeuds est donc discrétisée. La région d'intérêt est de ce fait représentable sous forme de matrice. La valeur de chaque case de la matrice correspond au nombre d'appareils présents dans cette zone. Ensuite, les cases adjacentes (incluant les diagonales) sont regroupées ensemble. Le score d'un groupe est le nombre de noeuds le composant. Ceci permet de savoir combien

d'appareils sont connectés dès lors que l'un d'entre eux peut se connecter au drone. Une structure en arbre similaire à ce que l'on a vu précédemment est utilisée pour connecter les autres membres du groupe. Une fois tous les groupes déterminés, le drone est placé sur la case qui permet de déservir le plus de noeuds possible.

Pour déduire la position future des noeuds se déplaçant sur la région d'intérêt étudiée, nous utilisons les tuiles précédemment visitées par tout appareil. Cela permet de voir leur trajectoire. Cet historique de tuiles est fourni en entrée à l'algorithme d'apprentissage artificiel qui doit déduire la prochaine tuile sur laquelle l'appareil se trouvera. En utilisant les positions prédites, il est possible de créer une deuxième matrice de score.

Finalement, le score global pour placer le drone est donné par une combinaison linéaire des deux matrices (positions présentes et futures). Un poids est appliqué sur la matrice des positions futures qui peut dépendre de la confiance en la prédiction. Si la prédiction a une probabilité faible de correctement déduire la position future des appareils, le poids sera proche de zéro, alors que si la fiabilité est excellente ce poids peut être arbitrairement haut (une valeur très élevée rendra le score des positions présentes négligeable; nous ne tiendrons compte que du futur).

L'évaluation de performance a permis de déterminer que la précision de la prédiction dépend de la taille des tuiles, mais surtout du nombre de tuiles passées considérées. Notamment, passer d'une à deux tuiles augmente significativement la précision puisque deux tuiles nous permettent de savoir la direction vers laquelle un noeud se déplace.

5.1 In a nutshell

This chapter focuses on the optimal placement of relay drones to allow connecting devices to the network. The use of drones is useful in alleviating locations with an unusual high density of devices or if infrastructure is failing locally. Drones are considered to have Internet access and act as root of a tree-like topology (such as RPL, see section 2.4) and devices connect to the drone directly or *via* other devices. The placement of drones is determined using an original scoring system and by predicting the future position of devices with ML. This contribution is still work in progress. The key aspects covered in this chapter are:

- The creation and implementation of a novel scoring system to determine the best locations to place relay drones and
- The use of machine learning to predict the future position of devices. Current and future positions are combined to place the UAV;

5.2 Context

The solutions proposed in chapters 3 and 4 are assumed to have at least some devices that can connect to a cellular antenna directly. Once connected to the antenna, they allow others to connect through them to also get Internet connectivity. However, in the case where a failure occurs on an antenna or a sudden increase in network traffic (such as an event) happens, the cellular infrastructure might not be sufficient to provide all devices with connectivity. It is possible no device can connect to

Internet at all (e.g. sensors equipped with a short range technology only). High costs of cellular connectivity or difficulties in deploying an antenna to specific places can also be the reason why a location lacks Internet access.

As explained in previous chapters, mobility of devices adds complexity to the problem. One solution is to deploy UAVs to act as relays and place them to critical locations. However, because of the mobility of devices drones can end up being at a location where there is no one left to service. On the other hand, by the time an area is deemed critical in terms of number of devices (causing high traffic density) and the drone travels to it, QoS will drop during that period of time.

5.3 Related work

The authors in [140] propose to put in place mobile relays in the studied area to help fixed relays during periods of peek network traffic. Those heavy traffic periods are predicted using Markov chains. Then, the best paths for mobile relays to follow are computed and those relays are sent to patrol on them, ultimately helping fixed relays to absorb network packets. However, end users can not connect to other end users, which limits the flexibility of the proposed solution. It can also be a problem if there are way more users compared to the amount of relays, as several of them might not be covered at all.

Tabatabai et al. [141] have developed an algorithm to opportunistically select relays in the context of vehicular network using different threshold based online algorithms. The idea is to divide a location (such as a city) in a grid and to elect one Local Community Broker (LCB) per tile. Nodes are connected to the elected LCB in the current zone and can subscribe to any service in the whole grid through this LCB. However, if there are tiles within the grid with a substantial amount of users, the local LCB can be congested. Although the service time is studied for the performance evaluation, network metrics such as delay, throughput, expected transmission count (ETX) or energy consumption are not covered.

Liu et al. [51] propose to use matching game to predict future radio conditions of UAVs. Each UAV determines its own needs and adapts the data transmission strategy using matching game. This allows to dynamically adapt mode selection, time scheduling and channel allocation to improve the performance of the network. However, this solution is designed such as only a subset of devices are generating data and simulations are done with a few tens of them at the same time. These make the proposed algorithm inadequate for a context where several hundreds of nodes are transmitting at the same time.

Authors in [142] consider a deep learning approach to select a relay for nodes suffering from bad radio conditions due to shadowing. A user in a network will take into account several metrics as input features for the ML algorithm to find the most suitable relay. However, any user can only connect through one intermediate relay to reach the access point. This limits the flexibility of the solution, especially if the usage of a short range wireless technology is desired to connect devices to a drone.

The solution presented in [143] uses UAVs as base stations to assist ground users in case of an emergency. The solution is based upon a deep reinforcement learning algorithm to find the best position of UAVs to cover as many users as possible. However, the algorithm is tested using static ground users and they are

directly connected to the UAVs. This makes the proposed solution non adapted to a context where all users are assumed to have mobility.

In [144], the authors propose a solution to compute the trajectory of UAVs acting as base stations by taking into account the current and future positions of users. The current location of users is determined when they tweet (using the Tweeter API). Then, an echo state network prediction algorithm predicts the future position of users according to their current position. Finally, a multi-agent Q-learning algorithm allows to use this information to compute the position of the UAVs. The proposed solution relies on users actively using a social network and only walking users are considered (no high-mobility). This prevents the solution to work with devices such as sensors, actuators or connected vehicles that do not use Tweeter.

5.4 Motivation

As we presented in last section, the use of drones as relays is something that has already been done in several contributions. However, considering low-mobility devices (such as pedestrians with smartphones and connected bicycles), high-mobility devices (like connected cars) and the current and future positions of devices was not achieved. The constantly changing positions of mobile connected devices on the network imply the critical location where the density of devices is highest is changing as well. It is possible to find this location given the position of all devices, but once it is identified there is a delay for the drone to reach it.

This is why we propose a mechanism to identify the current and future critical locations to allow the placement of drones on time to prevent a drop in QoS. The area of interest is divided in tiles and a scoring system reflects the amount of devices present on the different tiles. This allows to find the ones with the most devices. We also use ML to predict the future location of devices which makes it possible to determine the future score of the tiles and place the UAVs before congestion occurs.

5.5 Proposed architecture

The architecture is inspired from SDN (similar to the one presented on figure 3.2, section 3.5). The first part is located in the cloud. It is composed of the control and data servers that are connected to Internet (fig. 5.1). The second part is composed of the networks managed by Secondary Controllers (SCs, see fig. 5.1). In this chapter, we focus on one such instance. In [5], a gateway (SC) is an edge router and the topology managed by it is built using the Connected Dominating Set (CDS) algorithm. In this scheme, dominating nodes are elected according to metrics like the connectivity degree and link quality. The objective is to create a local backbone of connected dominating nodes such that every other node (the dominated ones) can directly connect to one dominating node.

However, in this chapter we use a somewhat different approach. In our case, the gateway (depicted as SC on fig. 5.1) is a drone and, instead of building the local topology using the CDS approach, we use the tree-like topology covered in previous chapters. That is, each Local Controller (LC on fig. 5.1) is a RPL sink (see section 2.4). Each sink is directly connected to the UAV.

The local topology is depicted on figure 5.2. Several connected users and devices

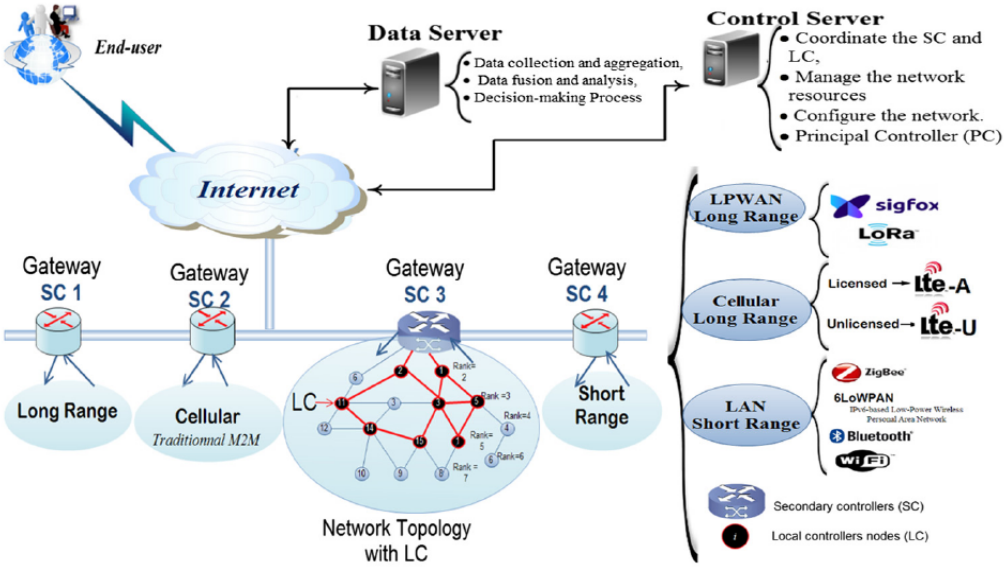


Figure 5.1: The architecture as depicted in [5]. We focus on one topology managed by a secondary controller (SC) or gateway.

are traveling on a road network: they can be soft-mobility users like pedestrians, bicycles or scooters or high-mobility users such as motorized vehicles. Devices such as sensors placed for instance on e-scooters to measure the quality of the air are considered as well. It is assumed devices are not able to directly connect to Internet so a drone (corresponding to a gateway on fig. 5.1) is deployed to allow them to do so. Devices use WiFi to connect to the drone and the drone uses cellular access to connect to Internet.

The drone allows to forward network traffic from devices to Internet back and forth and it is also connected to control server (as depicted on figure 5.2). The drone gathers the positions of devices and transmits it to the control server, which runs the ML algorithm to determine their future positions. We suppose the algorithm has already been trained on the area of interest. Once enough data from devices has been accumulated, the algorithm is used to determine the current and future positions of devices, and the server sends the coordinates to the drone which relocates accordingly. After some predetermined amount of time, the new location is sent to the drone and it relocates again, and so on.

5.6 Position prediction of mobile devices to place relay drone

To predict the future position of devices, we first divide the studied area into a number of square tiles. We then record the tiles a device travels on. These tiles are used as features for the ML algorithm to compute the prediction. Considering more tiles is more accurate because the trajectory of the device is better accounted for. For instance, let us take the limit case of using only the tile from which a user with a smartphone starts her journey. Let us also suppose that on this tile there are roads leading to each neighboring tile. In this case, it is likely the probability of the user traveling in any neighboring tile is the same because it is not possible to know the direction towards which the user is traveling. Using at least two tiles adds this information. However, if too many tiles are used the complexity will increase

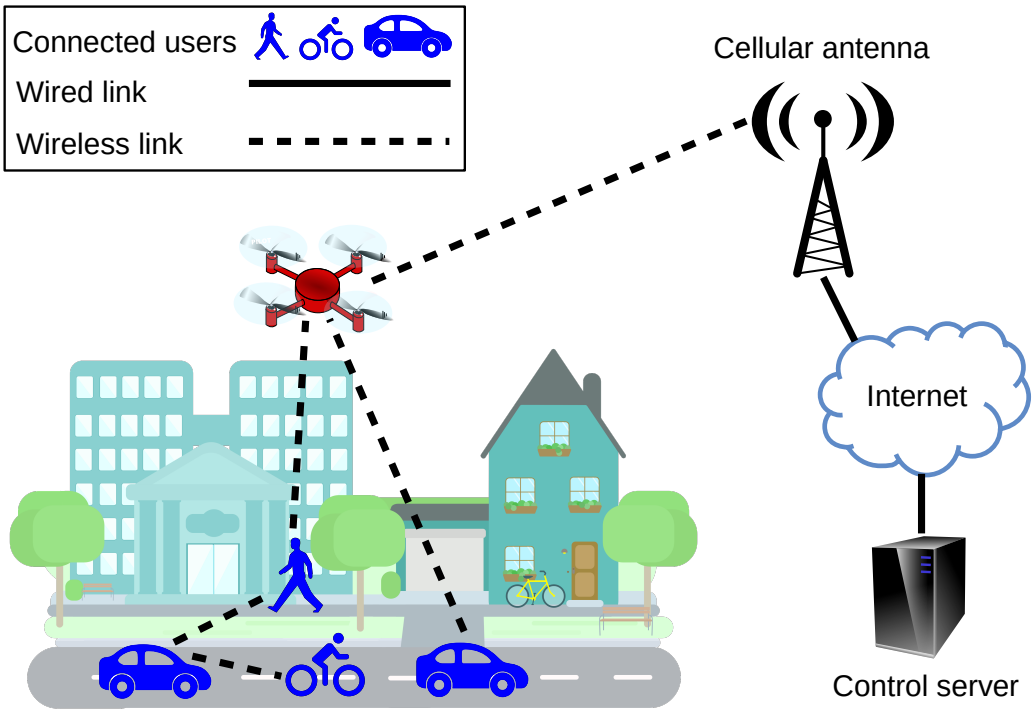


Figure 5.2: The proposed architecture uses a drone (or UAV) to serve as a relay to connect devices to Internet.

significantly. Also, it will take more time to predict the future positions and this could cause the drone to relocate too late, decreasing QoS.

Figure 5.3 illustrates the difference in predicting the future position of a pedestrian using two or three features. We consider the algorithm has already been trained. The real trajectory of the user is depicted as a green continuous arrow. The pedestrian starts in tile T_0, then walks to tiles T_1, T_2 and finally T_f. The prediction based on 2 features (blue dashed line on figure 5.3) takes as input tiles T_0 and T_1, and it is supposed it predicts that the user will be located in tile T_{p1}: it is anticipated the pedestrian will walk in a straight line, because during training that was the most probable path. Now, we suppose the prediction based on 3 features (orange dotted/dashed line on the figure) predicts the user will go to tile T_{p2}, after walking on tiles T_0, T_1 and T_2, which is more precise. This is because when considering two features, the fact a user travels through tiles T_0, T_1 and T_{p1} or through T_0, T_1 and T_2 is seen as the same (only the two first tiles are used, that is T_0 and T_1 in both cases). The reasoning is the same when considering four features instead of three, etc.

We define a geographic location containing roads, sidewalks, bicycle lanes, etc. separated into a grid of $G = G_x \cdot G_y$ tiles. Tile $g_{x,y}$ is the one with coordinates $[x; y]$, with $1 < x \leq G_x$ and $1 < y \leq G_y$. One tile has a size of $(\gamma_x \cdot \gamma_y)$ m^2. There are also D devices and one device is referenced as d_λ, with $1 < \lambda \leq D$. The relay (drone) is R. The relay and devices have a wireless range of w meters (for the short range technology).

We suppose the relay is always placed in the middle of a tile. Given we want all 8 adjacent tiles to be in range of the relay, we can see from figure 5.4 that the most

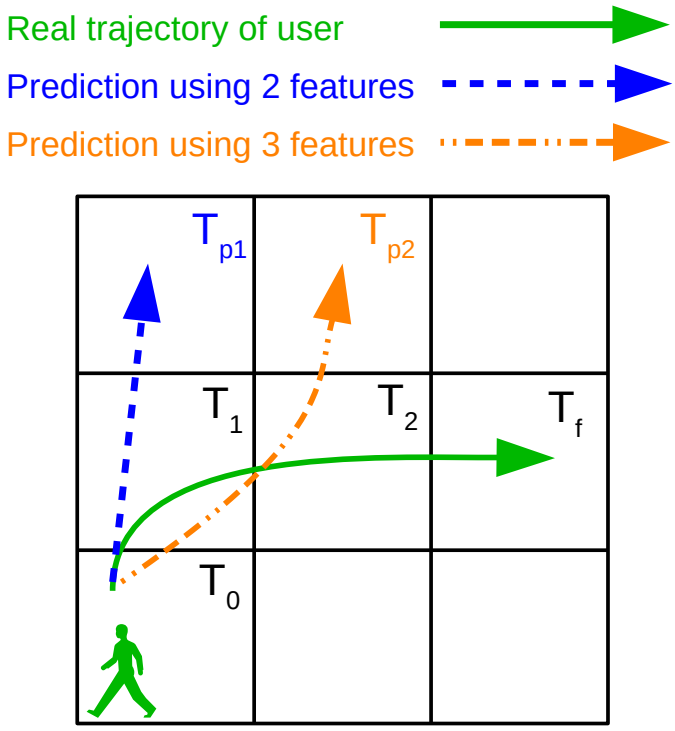


Figure 5.3: An example of prediction given 2 or 3 features compared to the real trajectory of user.

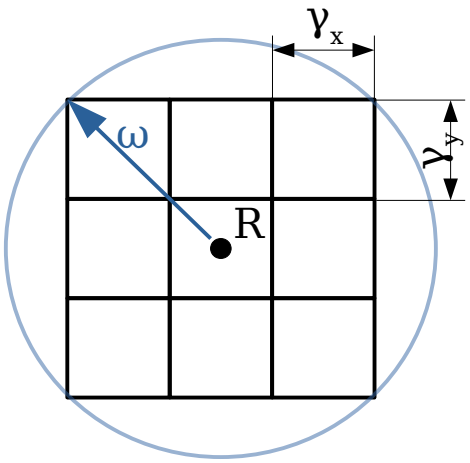


Figure 5.4: The most distant points are found when the circle representing the minimum wireless range w (in blue) intersects the square formed by the 9 tiles.

distant points from the central relay R are the four vertices of the square formed by the 9 tiles made from the central tile containing the relay and its 8 neighbors. Those points are located at 1.5 times the hypotenuse of 1 tile. Thus, the relation between the size of a tile (γ_x and γ_y) and the wireless range is given by:

$$1.5 \cdot \sqrt{(\gamma_x^2 + \gamma_y^2)} < w \tag{5.1}$$

We see from equation 5.1 that all points in the 9 tiles must be within range of the relay, hence the inequality. Note that in figure 5.4, $\gamma_x = \gamma_y$ (square tiles).

Devices travel through tiles and some will eventually end up in another one. To track the path of devices, we use discrete timesteps and record the last Θ tiles the device went through. For any device λ, we have:

$$H_\lambda = [h_i, h_{i-1}, h_{i-2}, ..., h_{i-\Theta+1}] \tag{5.2}$$

where H_λ is the vector of tiles the device passed through (its "history") and i represents the current timestep (or iteration).

We remind the reader that the objective is to predict the future position of devices: that is, to determine for all devices tile h_{i+1} using present and past tiles (vector H_λ, see eq. 5.2). For any device λ at each timestep i, we define the vector of tiles P_λ used for the positions prediction:

$$P_\lambda = [p_i, p_{i-1}, ..., p_{i-\Pi+1}] \quad | \quad \Pi \leq \Theta \quad \text{and} \quad P_\lambda \subseteq H_\lambda \tag{5.3}$$

The prediction can be based on all tiles the device has visited (in that case, $\Pi = \Theta$ and $P_\lambda = H_\lambda$, see eq. 5.2) or some of them only ($\Pi < \Theta$ and $P_\lambda \subset H_\lambda$). We point out that tiles are ordered from the most recent (the present at iteration i) to the oldest. We thus have:

$$h_{i-j} = p_{i-j} \; \forall \; i \quad \text{and} \quad 0 \leq j < \Pi \tag{5.4}$$

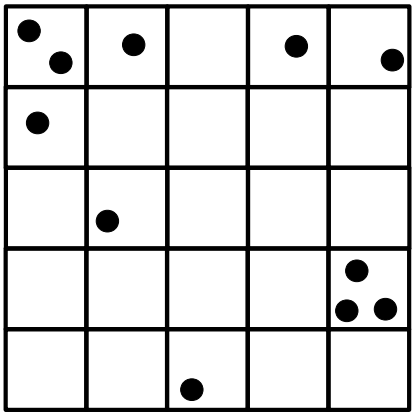
using H_λ and P_λ as defined in equations 5.2 and 5.3. The future position of devices is computed with a ML algorithm that uses the past position of devices as input features:

$$F_\lambda = A(P_\lambda) \quad | \quad \forall \; \lambda \tag{5.5}$$

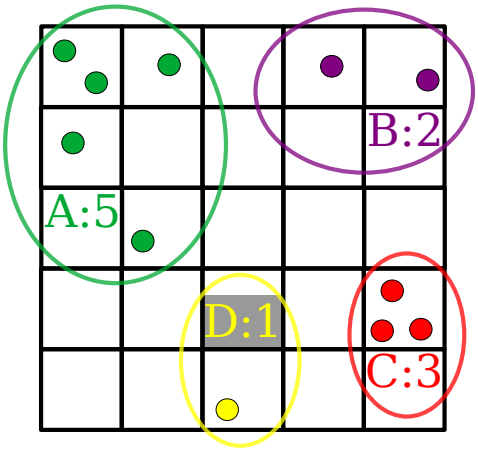
where F_λ is the future position of device λ, A is the ML algorithm used and P_λ is the vector of tiles as defined in equation 5.3. We define the score matrices $S^i_{x,y}$ and $S^{i+1}_{x,y}$ (for present and future position of devices, respectively) such that the value of each cell (tile) is the total number of devices that would be serviced if a relay was on this cell. A device is considered serviced if at least one device of the same group is in range of a relay placed on a neighboring cell (see fig. 5.4). The placement of the relay can finally be computed:

$$R_{x,y} = max[S^i_{x,y} + \phi^{i+1} \cdot S^{i+1}_{x,y}] \quad | \quad \forall \; x, y \quad \text{and} \quad 0 \leq \phi^{i+1} \leq 1 \tag{5.6}$$

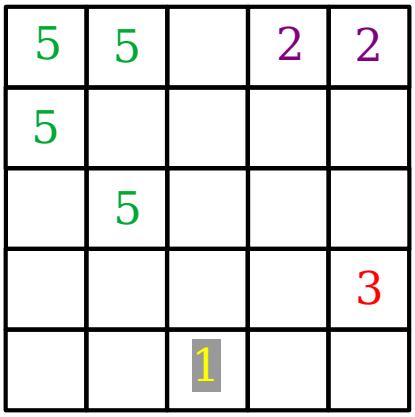
where $R_{x,y}$ is the best position to place the relay. It consists of taking into account the present and future position of devices. ϕ^{i+1} is the confidence factor of the prediction. That is, if the prediction has a low accuracy it is possible for the score of future (predicted) positions $S^{i+1}_{x,y}$ to have a lower impact on the location of the relay. Note that it is possible to set this value manually as well, for instance at 0 to remove any consideration of the future or to a big value to only consider the future.



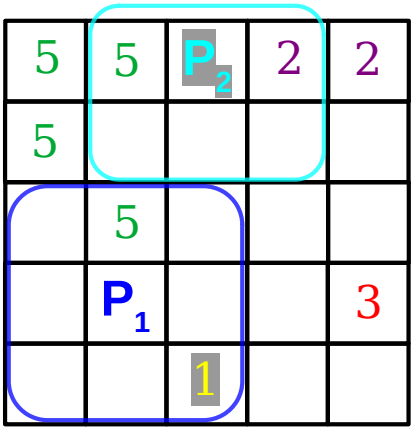
(a) Ungrouped devices on a 5 by 5 grid.



(b) 4 groups of devices are created : A, B, C and D with respectively 5, 2, 3 and 1 device(s).



(c) Each cell has a value worth the amount of devices in the group. Empty cells have a score of 0.



(d) Two of the best locations for a relay. P_1 is in range of groups A and D whereas P_2 is in range of A and B.

Figure 5.5: An example of grouping of devices.

As an example, let us consider figure 5.5. We start with a grid containing several devices (fig. 5.5a). If adjacent cells have at least one device, they are grouped. From figure 5.5b, we see that four groups are formed (A, B, C and D). Note that two devices in diagonal tiles are considered neighbors because they are in wireless range (see fig. 5.4). The number of devices of each group is counted and represents the score of the group. Still in figure 5.5b, we see that group A has a score of 5, group B has a score of 2, group C has a score of 3 and group D has a score of 1. Then, the score of each cell is set equal to the number of devices of the current group (fig. 5.5c). If a cell has no device, its score is 0. Finally, the relay is placed in the cell such that the maximum number of devices are serviced. In figure 5.5d, two of the best locations are shown: P_1 is in range of groups A and D for a total of 6 devices serviced. P_2 is in range of A and B and can thus service 7 devices. Thus, the relay will be placed at P_2.

To consider the future as well, a second matrix of (future) positions will be used, similar to the one depicted on figure 5.5. The computation of groups and score

Number of features	Size of the table
1	N
2	$\approx N^2$
3	$\approx N^3$
4	$\approx N^4$

Table 5.1: The impact of the number of features on the size of the resulting prediction table.

is the same as aforementioned, but the predicted positions are used instead of the current ones. The position of the relay is determined by both matrices.

5.7 ML algorithm analysis

We remind the reader that this chapter is still work in progress so we only present the analysis of the ML algorithm in this section.

Table 5.1 shows the impact of the number of tiles (features) on the size of the table required to store the prediction data. We suppose the studied area is divide into $N = X \cdot Y$ tiles of the same size. Each central tile has eight neighbors, edge tiles (excluding corners) have five neighbors and corner tiles have three neighbors. When there is one feature, each tile is mapped to one other (predicted) tile: there are N entries in the table.

When there are two features, we must consider all possible pairs of tiles: for any tile, the combinations are the current tile plus one neighbor (as explained above, there are eight, five or three neighbors). However, we must also consider that a device can stay on the same tile (because its mobility is low or null). The amount of entries E in the table is thus:

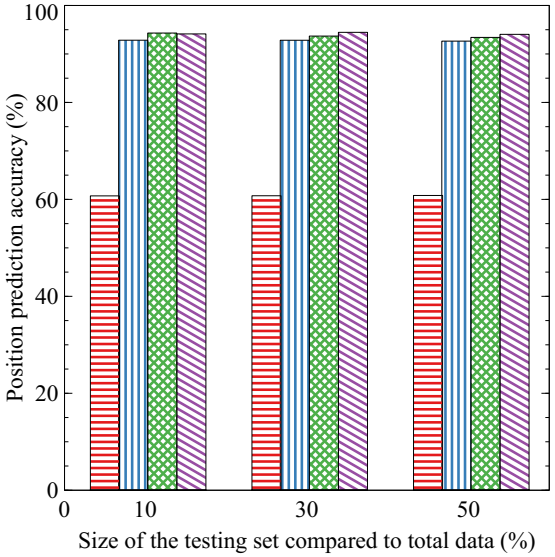
$$E = N \cdot [9 \cdot [(X-2) \cdot (Y-2)] + 6 \cdot [2 \cdot (X-2) + 2 \cdot (Y-2)] + 4 \cdot [4]] \quad (5.7)$$

That is, a device can start on any of the N tiles. Then, if the device is on a central tile, it can stay where it is or move to any of the eight neighbors (9 times the amount of central tiles). If the device is on a border tile excluding corners, it can stay on the tile or travel to any of the five neighbors (6 times the amount of border tiles). Finally, if the device is on a corner tile, it can also either stay on the tile or move to any of the three neighbors (4 times the amount of corner tiles). Rounded up (big oh), there are approximately N^2 possible pairs of (feature) tiles (the second line in table 5.1). The reasoning is the same when considering 3, 4, etc. features.

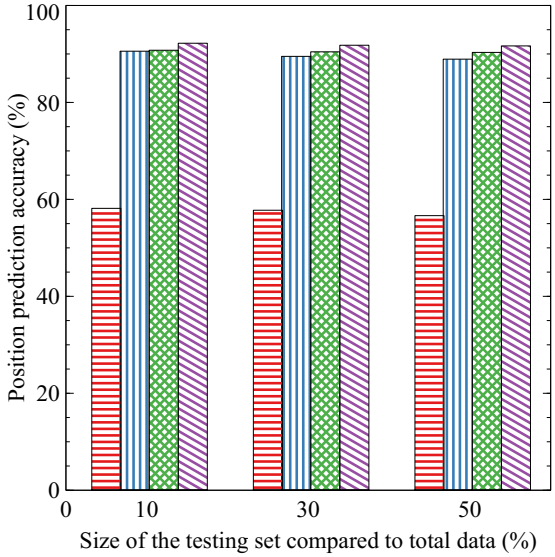
Decreasing the size of tiles and considering more of them improves accuracy at the cost of higher complexity. Three tiles is a good compromise between precision and complexity (see fig 5.6). It allows to differentiate between straight and curved trajectories more accurately (compared to using two tiles) without generating too much complexity if more tiles are considered (table 5.1).

Graphs on figure 5.6 plot the accuracy of the decision tree algorithm with a different number of features to predict the position of users. SUMO has been used to generate the trips of users and the prediction was made with the scikit-learn Python toolkit[1]. There are three different types of users: pedestrians, bicycles and

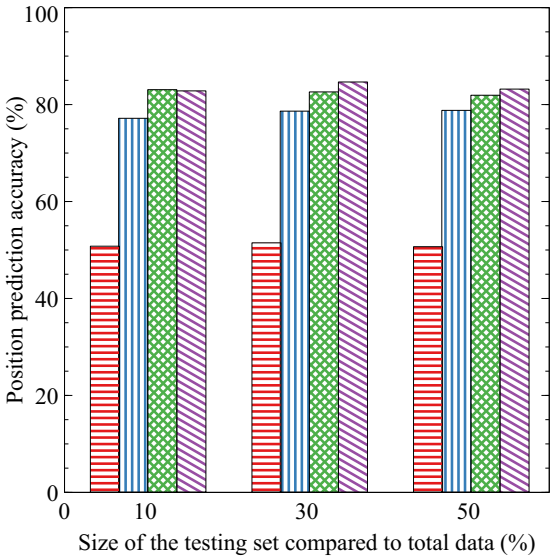
[1] https://scikit-learn.org/stable/index.html



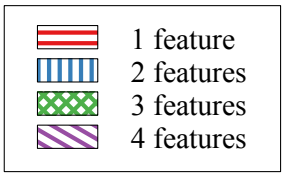
(a) The accuracy of the decision tree algorithm with a tile size of 20 m · 20 m.



(b) The accuracy of the decision tree algorithm with a tile size of 40 m · 40 m.



(c) The accuracy of the decision tree algorithm with a tile size of 100 m · 100 m.



(d) The legend of the three graphs.

Figure 5.6: The accuracy of position prediction for tile size and number of features given the size of the testing set.

cars. Users are generated about each 0.5 s and the running time is about 2000 s.

Graph 5.6a depicts the accuracy of the prediction with a tile size of 20 m · 20 m. Graphs 5.6b and 5.6c show the results of the prediction accuracy for a tile size of, respectively, 40 m · 40 m and 100 m · 100 m. 40 m is the "base" size because the range of WiFi used to connect users to the drone is about 105-110 m (as explained in section 3.8.3), so a tile size of 40 m allows a drone to cover all eight neighboring tiles (see figure 5.4). Studying smaller (figure 5.6a) and larger tiles (figure 5.6c) allow us to make a comparison with the "base" size.

The most accurate predictions are obtained with a smaller tile size (20 m, figure 5.6a) and 4 features (accuracy of about 94%). The smaller size implies there are less roads inside one tile, reducing the possibilities of where the device will travel next. More features allow to better account for devices' trajectory and increases accuracy as well. The precision drops significantly with only one feature compared to 2, 3 and 4 features. As explained at the beginning of section 5.6, this is due to the fact 2 features are enough to deduce the direction the device is traveling towards, which greatly increases the resulting precision. Indeed, considering 1 or 2 features increases accuracy by approximately 30% with a tile size of 20 m (figure 5.6a) and 40 m (figure 5.6b) or 25% when the size of tiles is 100 m (figure 5.6c).

5.8 Conclusion

In this chapter, we explained our solution to place relay drones based on a novel scoring system and a machine learning prediction algorithm to anticipate the future position of devices. The aim of the solution is to place relay drones where the density of devices is projected to be highest (critical location), preventing congestion. The drone is connected to Internet through a cellular link and devices connect to the drone directly or through other devices in a tree-shaped topology. The drone gathers position data from devices and communicates with a control server on Internet. The server runs the machine learning algorithm and sends the coordinates of the most critical location to the drone. The drone then travels to the location where the density of devices will be highest.

We created a scoring system where the studied area is divided into tiles to find the one that is best-suited to place the drone. The scoring system is based on the present and future locations of the devices. Their future position is predicted with a machine learning algorithm that we analyzed in terms of prediction accuracy, given the size of tiles and the number of features considered. It was concluded that the accuracy of the prediction is as high as 94% when using 20 m · 20 m tiles and 4 features. These are encouraging results and simulations are work in progress to evaluate the performance of the proposed solution.

Chapter 6

Conclusions and perspectives

Résumé français L'augmentation toujours plus importante des objets mobiles connectés mène à une génération massive de données. Ces appareils connectés sont différents du point de vue système, réseau et applicatif. Qui plus est, leur mobilité est variable et potentiellement différente pour chaque appareil. Cette hétérogénéité rend le problème de transport des données (le routage) complexe. Le travail de cette thèse vise à améliorer la qualité de service dans ce contexte.

Nous avons créer une architecture inspiré du paradigme SDN et nous avons développé une nouvelle fonction objectif qui tient compte des différents aspects des noeuds participant au réseau. La fonction objectif considère la vitesse relative entre les noeuds. Nous avons ensuite utilisé l'apprentissage artificiel dans le but de déduire le type de mobilité des noeuds afin de trouver ceux qui ont le profil le plus adapté pour servir de relais. Nous avons également ajouté des métriques localisées, telle que la nouvelle densité de paquets qui permet de trouver les zones critiques où sélectionner un relais est prioritaire. Finalement, nous avons développé une solution qui se sert de l'apprentissage artificiel dans le but prédire la position future des noeuds sur une topologie afin de placer un drone relais à l'endroit optimal. Cette solution permet de considérer soit la position actuelle des noeuds, leur position future ou une combinaison des deux en fonction de la précision de l'algorithme d'apprentissage artificiel.

Il serait intéressant d'étendre les métriques de la fonction objectif en ajoutant, par exemple, l'altitude du noeud courant, ainsi que d'ajouter le type de mobilité "volant". Ceci permettrait de mettre en place une topologie terrestre et aérienne à la fois et donnerait par exemple la possibilité de mesurer la qualité de l'air à diverses altitudes pour créer une carte en 3D de la qualité de l'air dans une zone géographique donnée. Que les appareils soient en l'air ou au sol serait transparent du point de vue utilisateur.

En poussant le concept de prédiction de la position future des noeuds pour placer un drone relais, il serait possible de développer une solution permettant d'assurer une connexion fiable en contexte dégradé (lors de catastrophes naturelles, par exemple). Ceci pourrait être rendu faisable en étendant aussi le concept de fonction objectif que nous avons utilisé. Ajouter des métriques comme la technologie sans fil disponible à un endroit donné, les besoins des gens bloqués ou l'emplacement de zones dangereuses pourrait être utilisé pour diriger les gens vers des endroits sécuritaires en attendant les secours.

Finalement, en connaissant et prédisant les besoins utilisateur, il peut être

possible de faire de l'allocation de ressources prédictive en prévoyant la "météo du réseau". Les ressources peuvent être allouées en temps réel en fonction des besoins, non seulement au niveau réseau (le *slicing*) mais aussi au niveau système ou applicatif. Par exemple, en anticipant qu'un appareil uniquement doté de WiFi va arriver à un emplacement précis, il pourrait être possible de choisir un relais (un autre appareil) doté de WiFi et 5G qui se trouve au même endroit pour transférer les données de/vers Internet. Du point de vue utilisateur, cela signifie qu'il serait possible d'avoir un accès au réseau peu importe la technologie sans-fil disponible sur l'appareil. Aussi, la qualité de service serait potentiellement plus élevée car le besoin du noeud pourrait également être anticipé et les ressources requises allouées lorsque l'utilisateur arrive à l'endroit prévu.

6.1 Conclusions

The ever increasing amount of connected mobile devices such as smartphones, motorized vehicles, bicycles and scooters, sensors and actuators, etc. is generating a massive amount of data. Dealing with this increase is a challenge for the near future as it is important to ensure quality of service and quality of experience to all devices and users connected to the network. They are heterogeneous on several aspects: system (battery capacity, processing power), networking (technology available, throughput capacity), application (transmission of one packet every hour versus voice over IP with strong delay constraint) and needs (save battery power, preference for high quality videos). Furthermore, many devices are mobile which adds complexity in establishing and maintaining a network topology. Anticipating their future position is tricky and they can potentially travel to locations with high interference, high fading or out of range of an access point. New devices can connect to the network and others can suddenly disconnect, breaking the connectivity of the topology. The present work focuses on addressing the aforementioned issues by providing a flexible approach that considers several aspects related to the system, network and application to ensure quality of service and quality of experience in infrastructure-less networks.

In this regard, we developed an architecture inspired from software defined networks, where the data and control planes are separated. A control server manages the dynamic creation of virtual network instances to improve the performance on the data part of the architecture. An instance is a local network topology where one or several root nodes called sinks act as a gateway between devices of the local network and a broader network like Internet. A topology is shaped like a tree and nodes connect to the sink or to other neighboring nodes. This is inspired from the well-known RPL routing protocol. Sinks can be connected to the data server of the global architecture and statistics on the data can be aggregated and sent back to the control server to adapt the global strategy in real-time (adding or removing instances). Examples include the creation of a new service needed at a specific location (such as streaming during a social event) or if increased congestion due to a sudden rise in the number of devices in an area.

Instances created by the control server are based on a programmable objective function, which is the next contribution of this work. The programmable objective function is transmitted to a local sink to create a new topology. The function is used to compute a score known as "rank" to order nodes on the topology according to the performance they provide on the network. We implemented it

to take into account several needs and aspects of the devices participating in the topology. Network-oriented metrics allow to improve performance by considering delay, available throughout or the expected transmission count (ETX) on the links. System-oriented metrics account for aspects specific to the device such as available memory or residual energy. These are important to save battery life and improve network lifetime. They also impact network performance because a node with less computational power compared to another one should not be favored. Finally, application-oriented metrics like a constraint on the maximum allowed delay for a Voice over IP application is important to improve the quality of experience of users. In the programmable objective function, we also included the velocity of users and the obsolescence of the rank to account for connected mobile devices.

Then, we detailed our novel machine learning solution which further improves support for mobility by selecting the best relay node amongst devices present in the studied area. The classification of the devices' mobility-type is the foundation of this contribution. Its purpose is to identify and classify them to differentiate soft-mobility from high-mobility devices. Then, this allows to select the nodes considered as best to serve as relays. A best node can be a soft-mobility device with a smartphone, considering his low velocity will yield to a more stable topology compared to selecting a connected car, which travels faster. This contribution is interesting as the only infrastructure required is one cellular antenna to which at least a few devices can connect to. On the other hand, the solution can still work without Internet access if local nodes need only to communicate to one another. To further refine this contribution, a new metric was created: the expected packet count (EPX). It permits to locate critical areas where adding a relay is a priority.

We then moved on to the two last contributions of this thesis, that are still work in progress. The objective is to address the problem of optimal placement of relay drones on an area to give network access to users. We first detailed the proposed scoring system. Once the area of interest is divided into tiles, the number of devices in each one is computed. Then, groups of devices are composed according the their proximity (devices in neighboring tiles - in range of one another - are part of the same group). This method allows to create a matrix to find the best locations to place the drone to service as many devices as possible. Finally, we discussed the machine learning prediction algorithm we designed to foresee the future position of users. The predicted positions are used to build a second matrix of scores to place the drone in locations that will be critical in terms of number of users and devices. The precision of the algorithm was analyzed and it can be used to apply a weight on the matrix of future positions to account for the uncertainty of the future.

It is worth mentioning that we considered in this work that the devices part of the network topologies are cooperative. That is, all devices participate to improve network performance even if it means, for instance, sacrificing their residual energy to some extent (except devices that must save energy at all cost). An improvement could be made in order to ensure the protection of users' data (e.g. guarantee anonymity). The precision of the machine learning algorithm could also be improved by entering as an input a larger amount of data. Using real mobility data that was gathered could yield to even more realistic results that would give a better insight about the performance of the deployment in real-life of the proposed solutions.

6.2 Perspectives

Several key aspects to address mobility in the Internet of Mobile Things have been covered in this thesis. There are many other issues that still require research and the following lists some leads for future work.

Unlocking full cooperation between aerial and vehicular networks

Throughout this manuscript, we discussed vehicular and aerial networks (made up of drones connected to one another). Although we considered the use of drones to alleviate congestion in networks composed of ground devices (see chapter 5), drones were only acting as relays and were not generating any data. The sources of data were devices located on a plane (2-dimensional) which simplifies the problem compared to a 3-dimensional space. On the other hand, in "pure" aerial networks it is usually assumed the hovering or flying devices are high enough so that shadowing is non-existent and fading is reduced (compared to a city setting at ground level).

The study of a ground and aerial network is more complex. Devices can move in any of the three dimensions and radio conditions can suffer drastically from shadowing and multipath fading. However, a full cooperation between ground and aerial networks would unlock several benefits. Let us take the example of measuring air quality on a city. Because of the variations in temperature, the heterogeneous emission of gases and the difference in wind speed and direction throughout the city, it is not very precise to measure the air quality only at ground level or at some altitude (using a weather balloon). Connecting smart devices and vehicles, along with drones and other aerial vehicles could be used to map the air quality in 3-dimensions.

The classification of mobility-type explained in chapter 4 can be extended to recognize more classes of mobility. This could include aerial vehicles as a new class by additionally considering the altitude of the device.

Ensuring reliable communications in a degraded context

As discussed in chapter 5, position prediction can be useful in alleviating congestion on critical locations. However, in some dire situations the proposed solution is not enough. We take the case of a city-wide evacuation. This can be justified in the event of a natural catastrophe such as a flood, tornadoes or an earthquake hitting a city. The disaster can entirely cut the city from the power grid and severely damage the infrastructure such that the cellular network is completely inoperable. Evacuating the city in such circumstances is a complex process because people must be directed through safe paths. They should be split to minimize the impact of jams so they can be evacuated as fast as possible. However, several factors impede the evacuation effort: people can panic and lack of connectivity to the network makes them unaware of safe routes that are not overcrowded. Furthermore, preparing in advance an evacuation procedure has limitations, as it is not possible to predict which roads will be blocked, which bridges will be unusable and so on.

A potential solution to this is the usage of a programmable objective function as detailed in chapter 3. By including more metrics such as which wireless technologies are available on the devices present in the critical area, the needs of people and the localization of hazardous zones, it could be possible to ensure connectivity to people to allow them to evacuate the city swiftly. The needs of people can include local map

downloading (with GNSS positioning) to travel to a safe location or phone service to call for help. Furthermore, deploying the objective function on a software defined network-like architecture increases flexibility by allowing, for instance, to dedicate more resources to locations with a higher density of users. The usage of drones can be very helpful because they do not impede the evacuation (they fly/hover) and can reach flooded areas without problem.

Network resources management based on forecasting strategy

Better understanding the needs of users and predicting future conditions on the whole network would unlock what can be seen as "network (weather) forecast". That is, to anticipate future needs and specific locations to optimize network resources, such as predictive slicing.

To unlock this, we would need a more thorough examination on the relevance of metrics. In classification (chapter 4) or prediction (chapter 5), the use of supervised learning implies giving inputs that are labeled with the correct (expected) output in order to train the algorithm. However, it is possible that using some metrics yield better increase in performance compared to using others. Perhaps in some cases system metrics (residual energy, amount of available memory, processing power, etc.) are more relevant than network metrics (delay, throughput usage, degree of connectivity, etc.). Perhaps a specific combination of all of them yields good all-around performance. It could also be possible to determine specific use cases (low/high density of users, majority of users consumes specific kind of application, etc.) in which specific metrics are better than others. One must also keep in mind that some metrics can directly affect others. For instance, a low throughput can cause delay to increase, though it is not necessarily the case.

Furthermore, although in this work we used several wireless technologies, cellular communication was assumed to be occurring between the relays and the cellular antenna, and shorter range communication was achieved through WiFi. It could be interesting to test the contributions on heterogeneous wireless technologies. For instance, a new metric could be the link quality between two devices using WiFi or LTE [145]. They could hence form a "long link" with a long range technology and other devices would be connected with a shorter range technology. Given some technologies focus on energy saving with low throughput (such as Bluetooth) and others offer more throughput (e.g. WiFi), some parts of a topology could rely on one whereas a different part would rely on another one.

The concept of metrics to select the best link could be extended at the system level. For instance, using the concept of caching, one device considered as the best locally (perhaps it possesses the highest available memory and processing power) could cache data for a given amount of time that has a high probability of being requested by nearby users.

The forecasting strategy would allow the interoperability of solutions. Let us suppose many devices are predicted to be located in some area and that they are equipped with only one specific wireless technology (they all have the same one). On this area, this specific wireless technology does not allow Internet access. An accurate forecasting of available resources could be used to deploy (or activate) a device at this location that would serve as a relay: it could use the same wireless technology as the other devices and also be equipped with another technology that is able to connect to a nearby antenna, unlocking Internet access for the devices.

Slicing of network and other resources or caching can then be used to improve the devices' quality of service through the consideration of the aforementioned relevant metrics. This is interesting because from the devices' perspective, they have a transparent access to Internet no matter what wireless technology they use (or are using).

Finally, this strategy can even be deployed in the degraded context to predict the future position of people and monitor the evolution of available resources to help deciding if more drones should be deployed and where to allocate the resources first.

Appendix A

Appendix

Figure A.1 shows the UML class diagram of the packets that are used in the simulator. "Probe" is a signaling packet sent at regular intervals if no traffic was received from the parent or children. It is to make sure they are still alive and connected to the current node.

The flowchart represented on figure A.2 shows the managing of received packets. Each node - whether a sink, relay or standard user - follows the same flowchart.

To work in more realistic scenarios, real-world maps from OpenStreetMap[1] were used. Figure A.3 shows an example of the location of the University Gustave Eiffel to the east of Paris. A portion of map from OpenStreetMap has the *.osm* extension, which is a XML type of file. The file is then imported in the Simulator of Urban MObility (SUMO) [146]. SUMO contains many tools to modify maps and generate mobility (fig. A.4). Once a map has been adapted for the simulation, random trips are created and run in SUMO. Figures A.5 and A.6 show an example of a simulation with users traveling on the map. Simulations were run with three different types of users: pedestrians, bicycles and cars (as depicted in fig. A.6). Once a run is completed in SUMO, users' location through time are recorded in another XML file. This data is finally imported in OMNeT++ and used to update the position of users during simulations.

[1] https://www.openstreetmap.org/

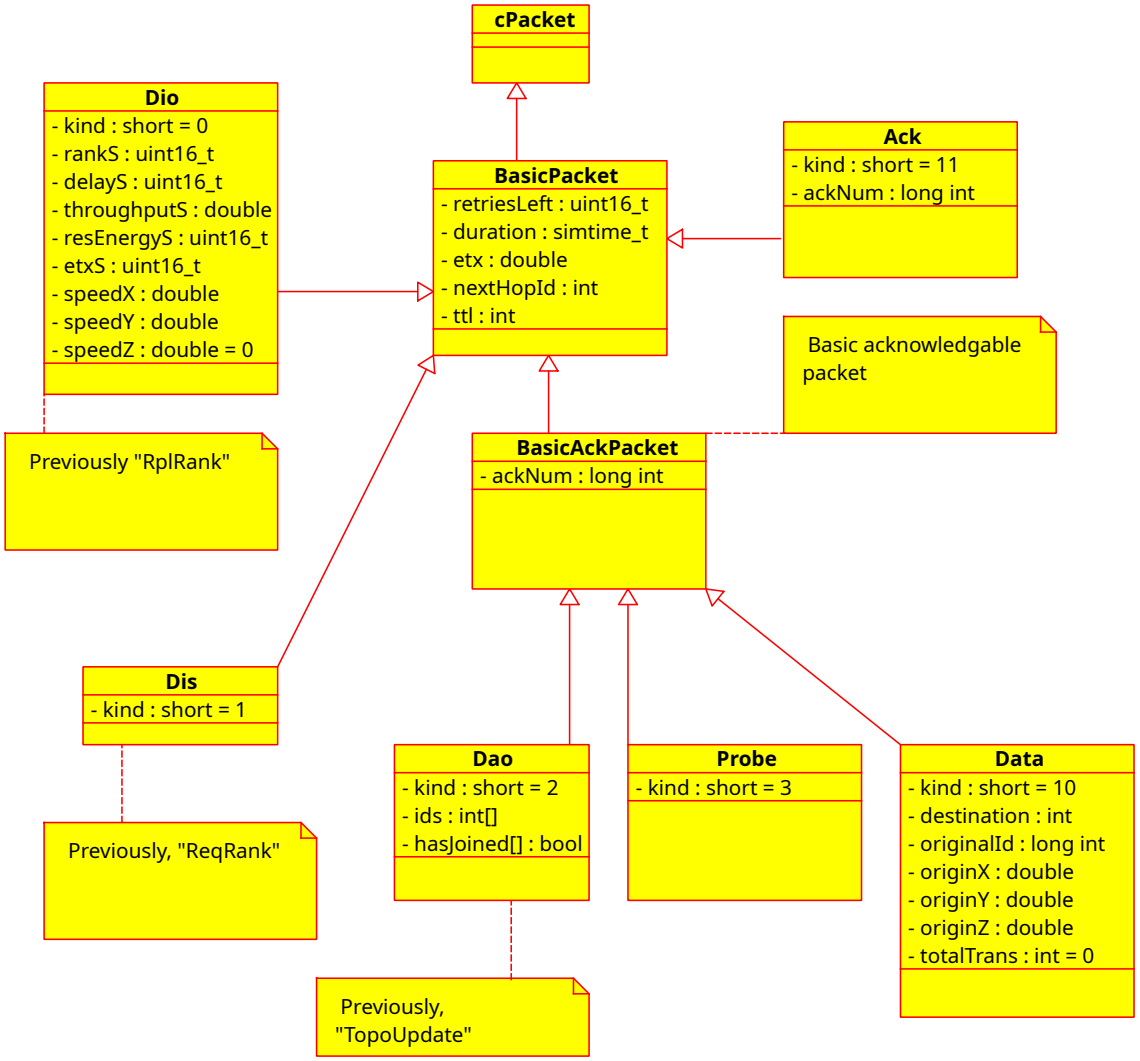


Figure A.1: The UML class diagram of the packets.

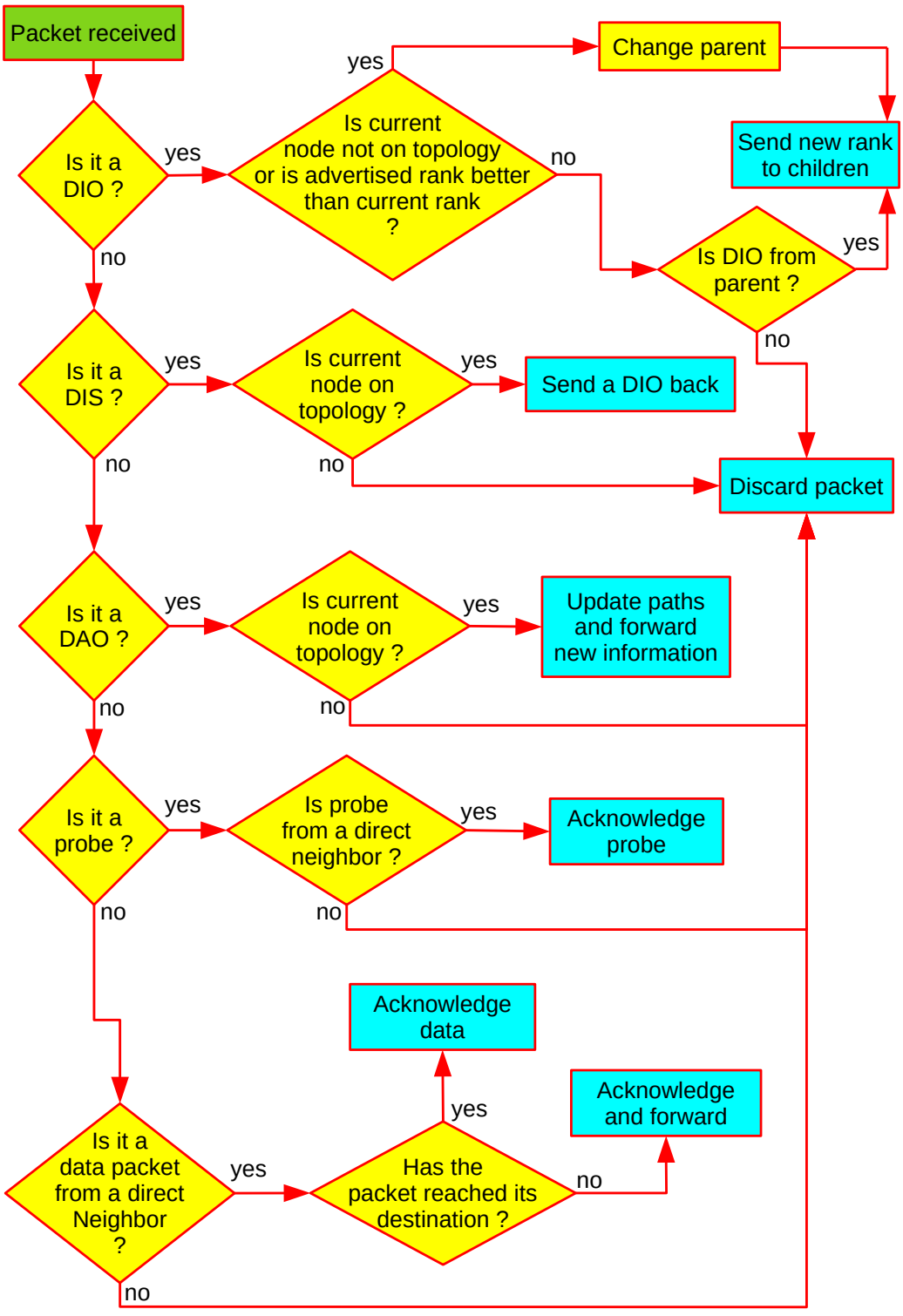


Figure A.2: The flowchart representing the behavior of the implemented version of RPL when a packet is received. The green box (top left) is the entry point and light blue boxes (without outgoing arrows) are final states.

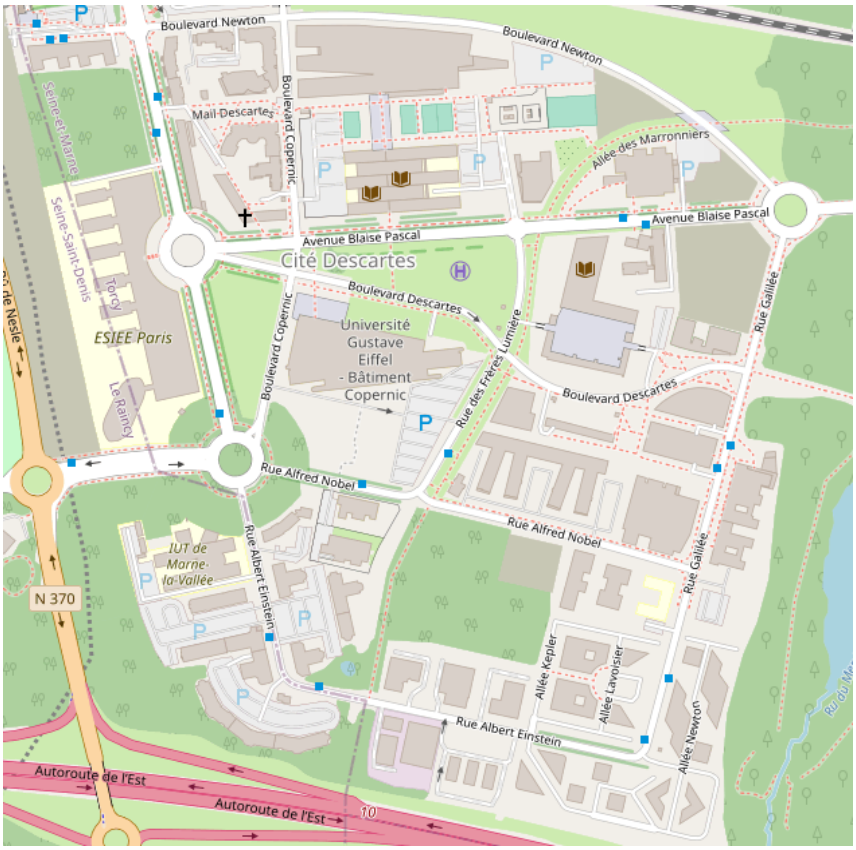


Figure A.3: Cité Descartes as seen on OpenStreetMap. The size of the area is about 1 km · 1 km.

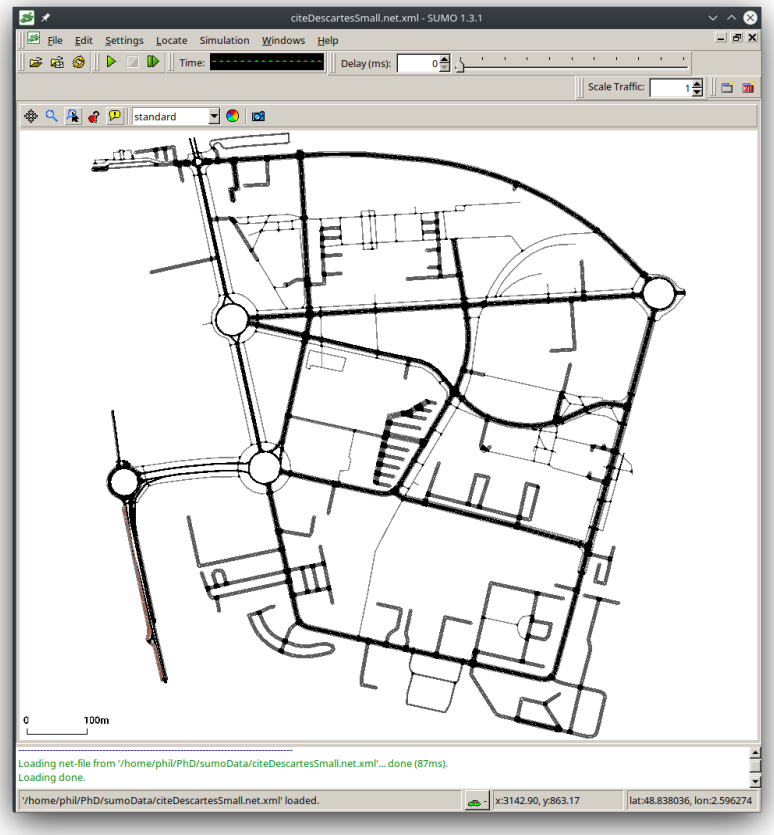


Figure A.4: The map imported in SUMO has been slightly modified.

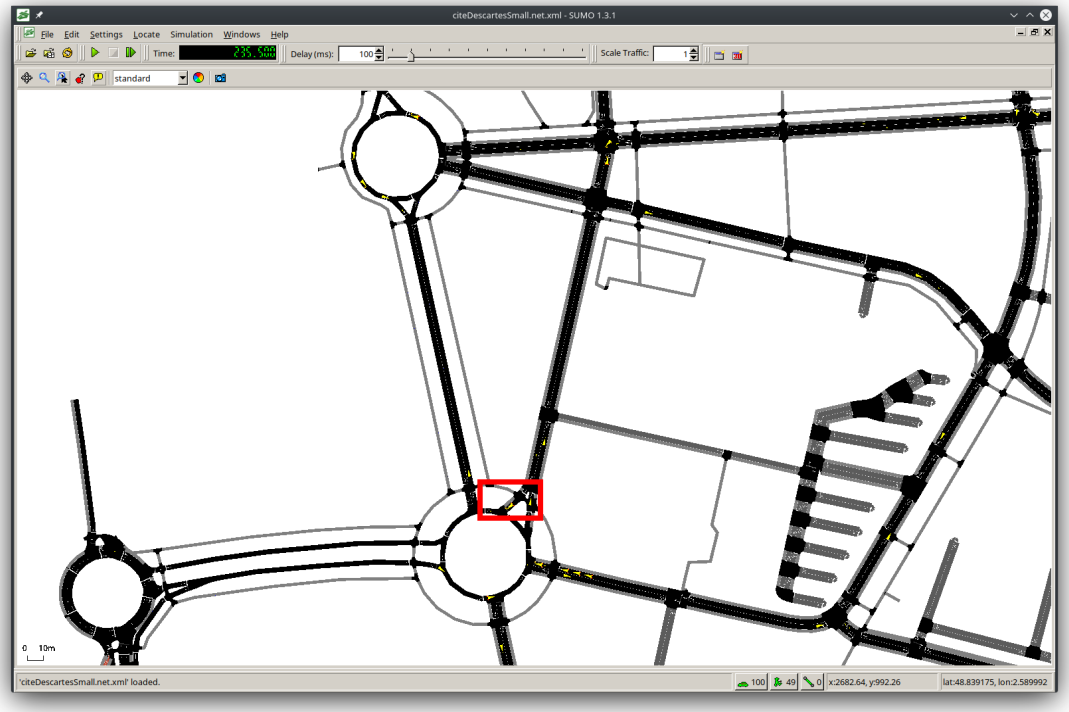


Figure A.5: Screenshot taken after 235.5 seconds. The yellow triangles represent cars. A zoom of the red box is shown on figure A.6.

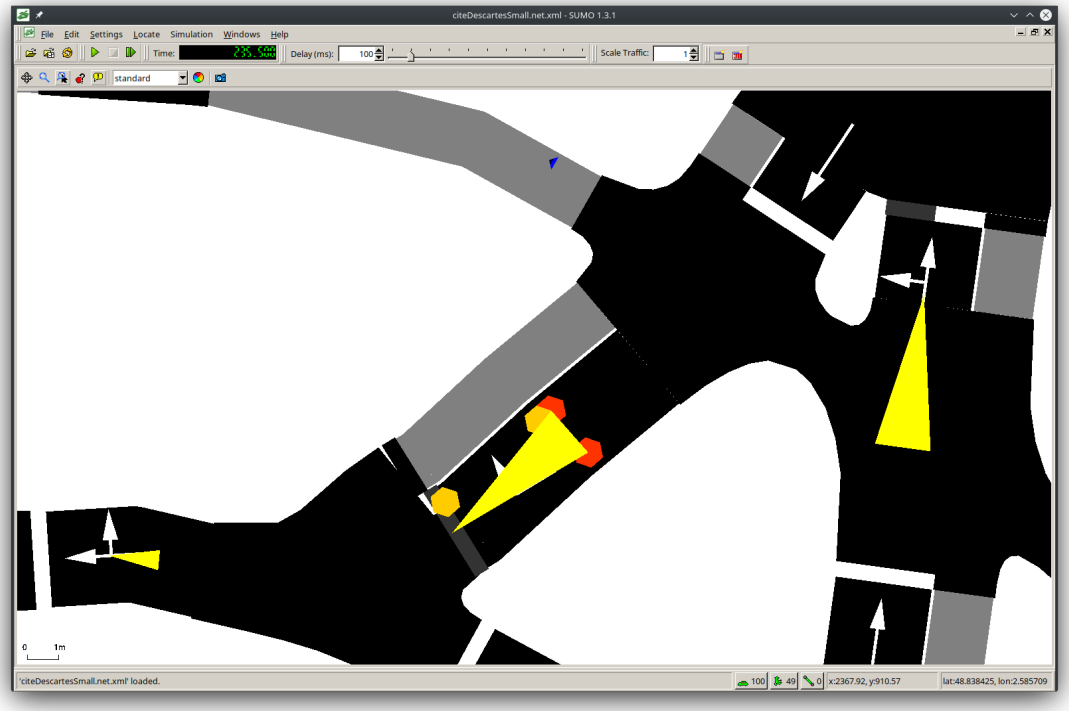


Figure A.6: A zoomed view of the roundabout (red box from fig. A.5). Yellow triangles are vehicles: cars are the large ones and bicycles are the small ones. Small blue triangles (middle top of the image) are pedestrians. Note that tail lights and blinkers are also visible on cars while in use.

Bibliography

[1] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, November 2018. [Online]. Available: https://elib.dlr.de/124092/

[2] I. Union, "Imt traffic estimates for the years 2020 to 2030," *Report ITU*, pp. 2370–0, 2015.

[3] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[4] A. C. Müller, S. Guido *et al.*, *Introduction to machine learning with Python: a guide for data scientists*. " O'Reilly Media, Inc.", 2016.

[5] D. Bendouda, A. Rachedi, and H. Haffaf, "Programmable architecture based on software defined network for internet of things: Connected dominated sets approach," *Future Generation Computer Systems*, vol. 80, pp. 188–197, 2018.

[6] F. Hillebrand, *GSM and UMTS: the creation of global mobile communication*. John Wiley & Sons, Inc., 2002.

[7] ——, "The creation of standards for global mobile communication: Gsm and umts standardization from 1982 to 2000," *IEEE Wireless Communications*, vol. 20, no. 5, pp. 24–33, 2013.

[8] W. Stallings, *Data and Computer Communications 9th Edition*. 1 Lake Street, Upper Saddle River, New Jersey, 07458: Prentice Hall, 2011.

[9] C. Bisdikian, "An overview of the bluetooth wireless technology," *IEEE Communications magazine*, vol. 39, no. 12, pp. 86–94, 2001.

[10] J. Zheng and M. J. Lee, "A comprehensive performance study of ieee 802.15. 4," *Sensor network operations*, vol. 4, pp. 218–237, 2006.

[11] R. Alexander, A. Brandt, J. Vasseur, J. Hui, K. Pister, P. Thubert, P. Levis, R. Struik, R. Kelsey, and T. Winter, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550, Mar. 2012. [Online]. Available: https://rfc-editor.org/rfc/rfc6550.txt

[12] N. A. Somani and Y. Patel, "Zigbee: A low power wireless technology for industrial applications," *International Journal of Control Theory and Computer Modelling (IJCTCM)*, vol. 2, no. 3, pp. 27–33, 2012.

[13] A. J. Wixted, P. Kinnaird, H. Larijani, A. Tait, A. Ahmadinia, and N. Strachan, "Evaluation of lora and lorawan for wireless sensor networks," in *2016 IEEE SENSORS*. IEEE, 2016, pp. 1–3.

[14] M. Lauridsen, B. Vejlgaard, I. Z. Kovács, H. Nguyen, and P. Mogensen, "Interference measurements in the european 868 mhz ism band with focus on lora and sigfox," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2017, pp. 1–6.

[15] P. Layer, "Ieee standard for ethernet," 2018.

[16] G. R. Hiertz, D. Denteneer, L. Stibor, Y. Zang, X. P. Costa, and B. Walke, "The ieee 802.11 universe," *IEEE Communications Magazine*, vol. 48, no. 1, pp. 62–70, 2010.

[17] "Internet Protocol," RFC 791, Sep. 1981. [Online]. Available: https://rfc-editor.org/rfc/rfc791.txt

[18] D. S. E. Deering and B. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 8200, Jul. 2017. [Online]. Available: https://rfc-editor.org/rfc/rfc8200.txt

[19] "Transmission Control Protocol," RFC 793, Sep. 1981. [Online]. Available: https://rfc-editor.org/rfc/rfc793.txt

[20] "User Datagram Protocol," RFC 768, Aug. 1980. [Online]. Available: https://rfc-editor.org/rfc/rfc768.txt

[21] S. Dang, O. Amin, B. Shihada, and M.-S. Alouini, "What should 6g be?" *Nature Electronics*, vol. 3, no. 1, pp. 20–29, 2020.

[22] P. Yang, Y. Xiao, M. Xiao, and S. Li, "6g wireless communications: Vision and potential techniques," *IEEE Network*, vol. 33, no. 4, pp. 70–75, 2019.

[23] T. Huang, W. Yang, J. Wu, J. Ma, X. Zhang, and D. Zhang, "A survey on green 6g network: Architecture and technologies," *IEEE Access*, vol. 7, pp. 175 758–175 768, 2019.

[24] T. Bhandare, "Lte and wimax comparison," *Santa Clara University*, vol. 12, pp. 19–20, 2008.

[25] K. Fazel and S. Kaiser, *Multi-carrier and spread spectrum systems: from OFDM and MC-CDMA to LTE and WiMAX*. John Wiley & Sons, 2008.

[26] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5g wireless networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1617–1655, 2016.

[27] T. Mshvidobadze, "Evolution mobile wireless communication and lte networks," in *2012 6th international conference on Application of information and communication technologies (AICT)*. IEEE, 2012, pp. 1–7.

[28] F. Sakiz and S. Sen, "A survey of attacks and detection mechanisms on intelligent transportation systems: Vanets and iov," *Ad Hoc Networks*, vol. 61, pp. 33–50, 2017.

[29] B. Ji, X. Zhang, S. Mumtaz, C. Han, C. Li, H. Wen, and D. Wang, "Survey on the internet of vehicles: Network architectures and applications," *IEEE Communications Standards Magazine*, vol. 4, no. 1, pp. 34–41, 2020.

[30] M. Stoyanova, Y. Nikoloudakis, S. Panagiotakis, E. Pallis, and E. K. Markakis, "A survey on the internet of things (iot) forensics: Challenges, approaches and open issues," *IEEE Communications Surveys & Tutorials*, 2020.

[31] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," *Business & information systems engineering*, vol. 6, no. 4, pp. 239–242, 2014.

[32] S. Vaidya, P. Ambad, and S. Bhosle, "Industry 4.0–a glimpse," *Procedia Manufacturing*, vol. 20, pp. 233–238, 2018.

[33] S. Nahavandi, "Industry 5.0—a human-centric solution," *Sustainability*, vol. 11, no. 16, p. 4371, 2019.

[34] M. F. Othman and K. Shazali, "Wireless sensor network applications: A study in environment monitoring system," *Procedia Engineering*, vol. 41, pp. 1204–1210, 2012.

[35] X. Wang, Z. Ning, X. Hu, E. C.-H. Ngai, L. Wang, B. Hu, and R. Y. Kwok, "A city-wide real-time traffic management system: Enabling crowdsensing in social internet of vehicles," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 19–25, 2018.

[36] T. Ruohomäki, E. Airaksinen, P. Huuska, O. Kesäniemi, M. Martikka, and J. Suomisto, "Smart city platform enabling digital twin," in *2018 International Conference on Intelligent Systems (IS)*. IEEE, 2018, pp. 155–161.

[37] X. Wang, Z. Ning, and L. Wang, "Offloading in internet of vehicles: A fog-enabled real-time traffic management system," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4568–4578, 2018.

[38] W.-J. Chang, L.-B. Chen, and K.-Y. Su, "Deepcrash: A deep learning-based internet of vehicles system for head-on and single-vehicle accident detection with emergency notification," *IEEE Access*, vol. 7, pp. 148 163–148 175, 2019.

[39] V. Davydov and S. Bezzateev, "Accident detection in internet of vehicles using blockchain technology," in *2020 International Conference on Information Networking (ICOIN)*. IEEE, 2020, pp. 766–771.

[40] S. Pallavi and S. R. Sarangi, "Internet of things: architectures protocols and applications," *Journal of Electrical and Computer Engineering*, vol. 2017, pp. 1–0147, 2017.

[41] B. Dorsemaine, J.-P. Gaulier, J.-P. Wary, N. Kheir, and P. Urien, "Internet of things: a definition & taxonomy," in *2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies*. IEEE, 2015, pp. 72–77.

[42] I. Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran, and M. Guizani, "Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges," *IEEE wireless communications*, vol. 24, no. 3, pp. 10–16, 2017.

[43] M. Chen, Y. Ma, J. Song, C.-F. Lai, and B. Hu, "Smart clothing: Connecting human with clouds and big data for sustainable health monitoring," *Mobile Networks and Applications*, vol. 21, no. 5, pp. 825–845, 2016.

[44] S. W. Loke, "The internet of flying-things: Opportunities and challenges with airborne fog computing and mobile cloud in the clouds," *arXiv preprint arXiv:1507.04492*, 2015.

[45] C. Zhan, Y. Zeng, and R. Zhang, "Energy-efficient data collection in uav enabled wireless sensor network," *IEEE Wireless Communications Letters*, vol. 7, no. 3, pp. 328–331, 2017.

[46] W. Ayoub, A. E. Samhat, F. Nouvel, M. Mroue, and J.-C. Prévotet, "Internet of mobile things: Overview of lorawan, dash7, and nb-iot in lpwans standards and supported mobility," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1561–1581, 2018.

[47] M. Gerla, E.-K. Lee, G. Pau, and U. Lee, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds," in *2014 IEEE world forum on internet of things (WF-IoT)*. IEEE, 2014, pp. 241–246.

[48] J. Loo, J. L. Mauri, and J. H. Ortiz, *Mobile ad hoc networks: current status and future trends*. CRC Press, 2016.

[49] A. R. Ragab, "A new classification for ad-hoc network," *iJIM*, vol. 14, no. 14, p. 215, 2020.

[50] O. S. Oubbati, N. Chaib, A. Lakas, P. Lorenz, and A. Rachedi, "Uav-assisted supporting services connectivity in urban vanets," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3944–3951, 2019.

[51] D. Liu, J. Wang, Y. Xu, Y. Xu, Y. Yang, and Q. Wu, "Opportunistic mobility utilization in flying ad-hoc networks: A dynamic matching approach," *IEEE Communications Letters*, vol. 23, no. 4, pp. 728–731, 2019.

[52] A. Chriki, H. Touati, H. Snoussi, and F. Kamoun, "Fanet: Communication, mobility models and security issues," *Computer Networks*, vol. 163, p. 106877, 2019.

[53] J. Harri, F. Filali, and C. Bonnet, "Mobility models for vehicular ad hoc networks: a survey and taxonomy," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 4, pp. 19–41, 2009.

[54] D. S. J. De Couto, "High-throughput routing for multi-hop wireless networks," Ph.D. dissertation, Massachusetts Institute of Technology, 2004.

[55] M. Z. Chowdhury, M. Shahjalal, S. Ahmed, and Y. M. Jang, "6g wireless communication systems: Applications, requirements, technologies, challenges, and research directions," *IEEE Open Journal of the Communications Society*, 2020.

[56] S. Chen, Y.-C. Liang, S. Sun, S. Kang, W. Cheng, and M. Peng, "Vision, requirements, and technology trend of 6g: how to tackle the challenges of system coverage, capacity, user data-rate and movement speed," *IEEE Wireless Communications*, vol. 27, no. 2, pp. 218–228, 2020.

[57] F. Tang, Y. Kawamoto, N. Kato, and J. Liu, "Future intelligent and secure vehicular network toward 6g: Machine-learning approaches," *Proceedings of the IEEE*, vol. 108, no. 2, pp. 292–307, 2019.

[58] M. E. Morocho-Cayamcela, H. Lee, and W. Lim, "Machine learning for 5g/b5g mobile and wireless communications: Potential, limitations, and future directions," *IEEE Access*, vol. 7, pp. 137 184–137 206, 2019.

[59] 3rd Generation Partnership Project. (2020) 3gpp home. Accessed: 2020-10-08. [Online]. Available: https://www.3gpp.org/about-3gpp/about-3gpp

[60] A. Gupta and R. K. Jha, "A survey of 5g network: Architecture and emerging technologies," *IEEE access*, vol. 3, pp. 1206–1232, 2015.

[61] C.-X. Wang, F. Haider, X. Gao, X.-H. You, Y. Yang, D. Yuan, H. M. Aggoune, H. Haas, S. Fletcher, and E. Hepsaydir, "Cellular architecture and key technologies for 5g wireless communication networks," *IEEE communications magazine*, vol. 52, no. 2, pp. 122–130, 2014.

[62] J. de Carvalho Silva, J. J. Rodrigues, A. M. Alberti, P. Solic, and A. L. Aquino, "Lorawan—a low power wan protocol for internet of things: A review and opportunities," in *2017 2nd International Multidisciplinary Conference on Computer and Energy Science (SpliTech)*. IEEE, 2017, pp. 1–6.

[63] J.-S. Lee, Y.-W. Su, and C.-C. Shen, "A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi," in *IECON 2007-33rd Annual Conference of the IEEE Industrial Electronics Society*. Ieee, 2007, pp. 46–51.

[64] S. C. Ergen, "Zigbee/ieee 802.15. 4 summary," *UC Berkeley, September*, vol. 10, no. 17, p. 11, 2004.

[65] A. Gopalasingham, D.-G. Herculea, C. S. Chen, and R. Laurent, "Virtualization of Radio Access Network by Virtual Machine and Docker: Practice and Performance Analysis," in *IM 2017 - IFIP/IEEE International Symposium on Integrated Network Management*. Lisbon, Portugal: IEEE, May 2017, pp. 680–685. [Online]. Available: https://hal.inria.fr/hal-01427979

[66] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An updated performance comparison of virtual machines and linux containers," in *2015 IEEE international symposium on performance analysis of systems and software (ISPASS)*. IEEE, 2015, pp. 171–172.

[67] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *2008 grid computing environments workshop*. Ieee, 2008, pp. 1–10.

[68] T. Velte, A. Velte, and R. Elsenpeter, *Cloud computing, a practical approach*. McGraw-Hill, Inc., 2009.

[69] T. Verbelen, P. Simoens, F. De Turck, and B. Dhoedt, "Cloudlets: Bringing the cloud to the mobile user," in *Proceedings of the third ACM workshop on Mobile cloud computing and services*, 2012, pp. 29–36.

[70] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[71] A. Ceselli, M. Premoli, and S. Secci, "Mobile Edge Cloud Network Design Optimization," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1818–1831, Jun. 2017. [Online]. Available: https://hal.sorbonne-universite.fr/hal-01432579

[72] C. Systems, "Fog computing and the internet of things: Extend the cloud to where the things are," 2015.

[73] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90–97, 2015.

[74] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: A taxonomy, survey and future directions," in *Internet of everything*. Springer, 2018, pp. 103–130.

[75] C. Pritchard, Y. Beheshti, and m. sepahi, "Mobile Edge Computing: Architecture, Use-cases, Applications," May 2020, working paper or preprint. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02612631

[76] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5g," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[77] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications surveys & tutorials*, vol. 18, no. 1, pp. 236–262, 2015.

[78] F. Ben Jemaa, G. Pujolle, and M. Pariente, "Cloudlet- and NFV-based carrier Wi-Fi architecture for a wider range of services," *Annals of Telecommunications - annales des télécommunications*, vol. 71, no. 11, pp. 617–624, 2016. [Online]. Available: https://hal.sorbonne-universite.fr/hal-01292428

[79] Y. Li and M. Chen, "Software-defined network function virtualization: A survey," *IEEE Access*, vol. 3, pp. 2542–2553, 2015.

[80] F. Hu, Q. Hao, and K. Bao, "A survey on software-defined network and openflow: From concept to implementation," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2181–2206, 2014.

[81] O. Flauzac, C. Gonzalez, and F. Nolot, "SDN Based Architecture for Clustered WSN," in *2015 9th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*. Santa Cantarina, Brazil: IEEE, Jul. 2015, pp. 342–347. [Online]. Available: https://hal.univ-reims.fr/hal-02514867

[82] T. Abar, A. Rachedi, A. ben Letaifa, P. Fabian, and S. El Asmi, "Fellowme cache: Fog computing approach to enhance (qoe) in internet of vehicles," *Future Generation Computer Systems*, 2020.

[83] A. Rachedi and H. Badis, "Badzak: An hybrid architecture based on virtual backbone and software defined network for internet of vehicles," in *2018 IEEE International Conference on Communications (ICC)*. Kansas City, MO, USA: IEEE, 2018, pp. 1–7.

[84] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5g: Survey and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[85] N. Alliance, "Description of network slicing concept," *NGMN 5G P*, vol. 1, no. 1, 2016.

[86] F. Montori, L. Bedogni, M. Di Felice, and L. Bononi, "Machine-to-machine wireless communication technologies for the internet of things: Taxonomy, comparison and open issues," *Pervasive and Mobile Computing*, vol. 50, pp. 56–81, 2018.

[87] M. Z. Chowdhury, M. T. Hossan, A. Islam, and Y. M. Jang, "A comparative survey of optical wireless technologies: Architectures and applications," *IEEE Access*, vol. 6, pp. 9819–9840, 2018.

[88] S. R. Das, C. E. Perkins, and E. M. Belding-Royer, "Ad hoc On-Demand Distance Vector (AODV) Routing," RFC 3561, Jul. 2003. [Online]. Available: https://rfc-editor.org/rfc/rfc3561.txt

[89] M. Di Francesco, G. Anastasi, M. Conti, S. K. Das, and V. Neri, "An adaptive algorithm for dynamic tuning of mac parameters in ieee 802.15. 4/zigbee sensor networks," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*. IEEE, 2010, pp. 400–405.

[90] P. Thubert, "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)," RFC 6552, Mar. 2012. [Online]. Available: https://rfc-editor.org/rfc/rfc6552.txt

[91] O. Gnawali and P. Levis, "The Minimum Rank with Hysteresis Objective Function," RFC 6719, Sep. 2012. [Online]. Available: https://rfc-editor.org/rfc/rfc6719.txt

[92] D. Barthel, J. Vasseur, K. Pister, M. Kim, and N. Dejean, "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks," RFC 6551, Mar. 2012. [Online]. Available: https://rfc-editor.org/rfc/rfc6551.txt

[93] N. Accettura, L. A. Grieco, G. Boggia, and P. Camarda, "Performance analysis of the rpl routing protocol," in *2011 IEEE International Conference on Mechatronics*. IEEE, 2011, pp. 767–772.

[94] E. Ancillotti, R. Bruno, and M. Conti, "The role of the rpl routing protocol for smart grid communications," *IEEE Communications Magazine*, vol. 51, no. 1, pp. 75–83, 2013.

[95] L. B. Saad, C. Chauvenet, and B. Tourancheau, "Simulation of the rpl routing protocol for ipv6 sensor networks: two cases studies," 2011.

[96] B. Pavković, F. Theoleyre, and A. Duda, "Multipath opportunistic rpl routing over ieee 802.15. 4," in *Proceedings of the 14th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, 2011, pp. 179–186.

[97] E. Baccelli, M. Philipp, and M. Goyal, "The p2p-rpl routing protocol for ipv6 sensor networks: Testbed experiments," in *SoftCOM 2011, 19th International Conference on Software, Telecommunications and Computer Networks*. IEEE, 2011, pp. 1–6.

[98] P. Gonizzi, R. Monica, and G. Ferrari, "Design and evaluation of a delay-efficient rpl routing metric," in *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2013, pp. 1573–1577.

[99] Y. He, W. Xu, and X. Lin, "A stable routing protocol for highway mobility over vehicular ad-hoc networks," in *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*. IEEE, 2015, pp. 1–5.

[100] M. Bouaziz, A. Rachedi, and A. Belghith, "Ekf-mrpl: Advanced mobility support routing protocol for internet of mobile things: Movement prediction approach," *Future Generation Computer Systems*, vol. 93, pp. 822–832, 2019.

[101] G. Costantino, R. R. Maiti, F. Martinelli, and P. Santi, "Losero: a locality sensitive routing protocol in opportunistic networks," in *Proceedings of the 31st annual ACM symposium on applied computing*, 2016, pp. 644–650.

[102] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi, "Machine learning for predictive maintenance: A multiple classifier approach," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 812–820, 2014.

[103] J. C. Cheng, W. Chen, K. Chen, and Q. Wang, "Data-driven predictive maintenance planning framework for mep components based on bim and iot using machine learning algorithms," *Automation in Construction*, vol. 112, p. 103087, 2020.

[104] F. Li, A. Shinde, Y. Shi, J. Ye, X.-Y. Li, and W. Song, "System statistics learning-based iot security: Feasibility and suitability," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6396–6403, 2019.

[105] E. B. Karbab and M. Debbabi, "Maldy: Portable, data-driven malware detection using natural language processing and machine learning techniques on behavioral analysis reports," *Digital Investigation*, vol. 28, pp. S77–S87, 2019.

[106] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Communications Surveys & Tutorials*, 2019.

[107] M. Barcelo, A. Correa, J. L. Vicario, A. Morell, and X. Vilajosana, "Addressing mobility in rpl with position assisted metrics," *IEEE Sensors Journal*, vol. 16, no. 7, pp. 2151–2161, 2015.

[108] M. Bouaziz, A. Rachedi, A. Belghith, M. Berbineau, and S. Al-Ahmadi, "Ema-rpl: Energy and mobility aware routing for the internet of mobile things," *Future Generation Computer Systems*, vol. 97, pp. 247–258, 2019.

[109] Y. Chang, X. Yuan, B. Li, D. Niyato, and N. Al-Dhahir, "Machine-learning-based parallel genetic algorithms for multi-objective optimization in ultra-reliable low-latency wsns," *IEEE Access*, vol. 7, pp. 4913–4926, 2018.

[110] Y. Tahir, S. Yang, and J. McCann, "Brpl: Backpressure rpl for high-throughput and mobile iots," *IEEE Transactions on Mobile Computing*, vol. 17, no. 1, pp. 29–43, 2017.

[111] W. Khallef, M. Molnar, A. Benslimane, and S. Durand, "Multiple constrained qos routing with rpl," in *2017 IEEE International Conference on Communications (ICC)*. Paris, France: IEEE, 2017, pp. 1–6.

[112] M. O. Farooq, C. J. Sreenan, K. N. Brown, and T. Kunz, "Design and analysis of rpl objective functions for multi-gateway ad-hoc low-power and lossy networks," *Ad Hoc Networks*, vol. 65, pp. 78–90, 2017.

[113] H.-S. Kim, H. Kim, J. Paek, and S. Bahk, "Load balancing under heavy traffic in rpl routing protocol for low power and lossy networks," *IEEE Transactions on Mobile Computing*, vol. 16, no. 4, pp. 964–979, 2017.

[114] C. Cobarzan, J. Montavont, and T. Noel, "Analysis and performance evaluation of rpl under mobility," in *2014 IEEE symposium on computers and communications (ISCC)*. Funchal, Portugal: IEEE, 2014, pp. 1–6.

[115] I. El Korbi, M. B. Brahim, C. Adjih, and L. A. Saidane, "Mobility enhanced rpl for wireless sensor networks," in *2012 Third international conference on the network of the future (NOF)*. Gammarth, Tunisia: IEEE, 2012, pp. 1–8.

[116] R. Pradhan, S. Rakshit, and T. De, "Performance evaluation of rpl under mobility for vanets," in *2018 5th International Conference on Signal Processing and Integrated Networks (SPIN)*. Noida, India: IEEE, 2018, pp. 739–744.

[117] A. Serhani, N. Naja, and A. Jamali, "Aq-routing: mobility-, stability-aware adaptive routing protocol for data routing in manet–iot systems," *Cluster Computing*, vol. 22, pp. 1–15, 2019.

[118] B. Tian, K. M. Hou, H. Shi, X. Liu, X. Diao, J. Li, Y. Chen, and J.-P. Chanet, "Application of modified rpl under vanet-wsn communication architecture," in *2013 international conference on computational and information sciences*. Shiyang, China: IEEE, 2013, pp. 1467–1470.

[119] A. Dunkels, "Contiki: Bringing ip to sensor networks," *Ercim news*, no. 76, 2009.

[120] F. Österlind, "A sensor network simulator for the contiki os," *SICS Research Report*, 2006.

[121] T. Instruments, "Cc2520 datasheet: 2.4 ghz ieee 802.15. 4/zigbee rf transceiver," 2007.

[122] ——, "Msp430f543x, msp430f541x datasheet–mixed signal microcontroller (rev. e)," 2009.

[123] S. K. Debnath, M. Saha, N. Funabiki, and W.-C. Kao, "A throughput estimation model for ieee 802.11 n mimo link in wireless local-area networks," in *2018 3rd International Conference on Computer and Communication Systems (ICCCS)*. Nagoya, Japan: IEEE, 2018, pp. 327–331.

[124] M. Giordani, A. Zanella, T. Higuchi, O. Altintas, and M. Zorzi, "On the feasibility of integrating mmwave and ieee 802.11 p for v2v communications," in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*. Chicago, IL, USA, USA: IEEE, 2018, pp. 1–7.

[125] G. Zhang and S. Fischer-Hübner, "A survey on anonymous voice over ip communication: attacks and defenses," *Electronic Commerce Research*, vol. 19, no. 3, pp. 655–687, 2019.

[126] D. Astély, E. Dahlman, A. Furuskär, Y. Jading, M. Lindström, and S. Parkvall, "Lte: the evolution of mobile broadband," *IEEE Communications magazine*, vol. 47, no. 4, pp. 44–51, 2009.

[127] J. Wannstrom, "Lte-advanced," *Third Generation Partnership Project (3GPP)*, 2013.

[128] E. Dahlman, S. Parkvall, and J. Skold, *4G: LTE/LTE-advanced for mobile broadband*. Academic press, 2013.

[129] K. Shamganth and M. J. Sibley, "A survey on relay selection in cooperative device-to-device (d2d) communication for 5g cellular networks," in *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*. IEEE, 2017, pp. 42–46.

[130] M. S. Alam, J. W. Mark, and X. S. Shen, "Relay selection and resource allocation for multi-user cooperative ofdma networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 5, pp. 2193–2205, 2013.

[131] D. Abada, A. Massaq, A. Boulouz, and M. B. Salah, "An adaptive vehicular relay and gateway selection scheme for connecting vanets to internet via 4g lte cellular network," in *Emerging Technologies for Connected Internet of Vehicles and Intelligent Transportation System Networks*. Springer, 2020, pp. 149–163.

[132] Y. Li, P. Wang, D. Niyato, and W. Zhuang, "A dynamic relay selection scheme for mobile users in wireless relay networks," in *2011 Proceedings IEEE INFOCOM*. IEEE, 2011, pp. 256–260.

[133] N.-N. Dao, M. Park, J. Kim, J. Paek, and S. Cho, "Resource-aware relay selection for inter-cell interference avoidance in 5g heterogeneous network for internet of things systems," *Future Generation Computer Systems*, vol. 93, pp. 877–887, 2019.

[134] A. Mchergui, T. Moulahi, and S. Nasri, "Relay selection based on deep learning for broadcasting in vanet," in *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*. IEEE, 2019, pp. 865–870.

[135] Y. Zhou, H. Li, C. Shi, N. Lu, and N. Cheng, "A fuzzy-rule based data delivery scheme in vanets with intelligent speed prediction and relay selection," *Wireless Communications and Mobile Computing*, vol. 2018, 2018.

[136] S. K. Dhurandher, J. Singh, M. Obaidat, I. Woungang, S. Srivastava, and J. Rodrigues, "Reinforcement learning-based routing protocol for opportunistic networks," in *2020 IEEE International Conferences on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[137] Y. Zhang, X. Tang, Y. Xu, and W. Chen, "Data forwarding at intersections in urban bus ad hoc networks," in *2020 IEEE International Conferences on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[138] N. M. Al-Kharasani, Z. A. Zukarnain, S. K. Subramaniam, and Z. M. Hanapi, "An adaptive relay selection scheme for enhancing network stability in vanets," *IEEE Access*, 2020.

[139] Y. Zhao and L. Jiang, "A scheme of d2d-based delay analysis and vehicle relay algorithm in vanet," in *IOP Conference Series: Materials Science and Engineering*, vol. 715, no. 1. IOP Publishing, 2020, p. 012030.

[140] Z. Liao, J. Liang, and C. Feng, *Mobile relay deployment in multihop relay networks*. Elsevier, 2017, vol. 112.

[141] S. Tabatabai, I. Mohammed, A. A.-F. Senior, and J. Qadir, "Opportunistic selection of vehicular data brokers as relay nodes to the cloud," in *2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2020, pp. 1–6.

[142] A. Abdelreheem, O. A. Omer, H. Esmaiel, and U. S. Mohamed, "Deep learning-based relay selection in d2d millimeter wave communications," in *2019 International Conference on Computer and Information Sciences (ICCIS)*. IEEE, 2019, pp. 1–5.

[143] J. Qiu, J. Lyu, and L. Fu, "Placement optimization of aerial base stations with deep reinforcement learning," *arXiv preprint arXiv:1911.08111*, 2019.

[144] X. Liu, Y. Liu, Y. Chen, and L. Hanzo, "Trajectory design and power control for multi-uav assisted wireless networks: A machine learning approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7957–7969, 2019.

[145] B. Chen, J. Chen, Y. Gao, and J. Zhang, "Coexistence of lte-laa and wi-fi on 5 ghz with corresponding deployment scenarios: A survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 7–32, 2016.

[146] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018. [Online]. Available: https://elib.dlr.de/124092/