Supports de communications ubiquitaires pour les réseaux à l'échelle de la ville

Pierre Brunisholz

Communauté
UNIVERSITÉ Grenoble Alpes

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTÉ UNIVERSITÉ GRENOBLE ALPES

Spécialité : **Informatique**

Arrêté ministériel : 25 mai 2016

Présentée par

Pierre Brunisholz

Thèse dirigée par **Andrzej Duda**
et coencadrée par **Franck Rousseau**

préparée au sein du
Laboratoire d'Informatique de Grenoble (LIG)
dans **l'École Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique (EDMSTII).**

Supports de communications ubiquitaires pour les réseaux à l'échelle de la ville

Thèse soutenue publiquement le **10 juillet 2019**,
devant le jury composé de :

Isabelle Guérin Lassous
Professeure, Université Claude Bernard Lyon 1, Rapporteur
Nicolas Montavont
Professeur, IMT Atlantique, Rapporteur
Gentian Jakllari
Maître de conférences, Université de Toulouse - ENSEEIHT, Examinateur
Vivien Quéma
Professeur, Université Grenoble Alpes - Grenoble INP, Président
Andrzej Duda
Professeur, Université Grenoble Alpes - Grenoble INP, Directeur de thèse
Franck Rousseau
Maître de conférences, Université Grenoble Alpes - Grenoble INP, Co-Encadrant de thèse

Résumé en français

Le Wi-Fi est omniprésent dans les villes, que ce soit par le nombre grandissant de point d'accès public, ou la déploiement massif de points d'accès privés, sous la forme de boxes d'opérateurs notamment. Si nous supposons que l'ensemble de ces points d'accès soient utilisables afin de permettre à n'importe quel appareil d'accéder à Internet, alors nous aurions potentiellement une couverture réseau sur l'ensemble de la ville. Ce postulat nous a conduit à nous demander si le Wi-Fi pouvait être utilisé comme réseau à l'échelle urbaine.

Ce réseau pourrait plus particulièrement être utilisé dans un contexte de mobilité. Or, le Wi-Fi n'a pas été conçu dans le but de gérer des utilisateurs mobiles, et les appareils doivent régulièrement changer de points d'accès lorsqu'ils n'ont plus de connectivité. Ce mécanisme, appelé handover, peut être long car les appareils doivent d'abord constater leurs pertes de connectivité avant de commencer à chercher le prochain point d'accès auquel s'associer. Il peut être particulièrement long pour des appareils comme les smartphones car ces derniers sont contraint en énergie et n'appliquent donc pas une politique de handover aggressive.

Dans ce contexte nous avons cherché à caractériser les applications possible sur le Wi-Fi lorsqu'un utilisateur est mobile, en fonction de la durée de handover de son équipement, de sa vitesse et de la densité des points d'accès présents dans la ville. Nous avons constaté que pour les utilisateurs se déplaçant à faible allure, l'impact de la durée de handover est faible au regard de la connectivité totale, ce qui leur permet d'utiliser des applications gourmandes en terme de bande passante tant que celles-ci possèdent un certain degré de tolérances aux déconnexions. Cependant lorsque la vitesse de déplacement augmente, la durée de handover dégrade progressivement la connectivité des utilisateurs, de telle sorte que ceux ayant une allure élevée ne peuvent plus espérer utiliser les différents points d'accès. En effet, les équipements passent alors plus de temps à effectuer des handovers qu'à échanger des données applicatives.

Les retransmissions jouant un rôle important dans la durée de handover, nous avons étudié finement leurs fonctionnement en 802.11. Pour cela nous avons mis en place un banc d'essai nous permettant d'observer les séquences de messages retransmis par différentes implémentations de 802.11 lorsque l'on fait subitement disparaitre le point d'accès. Nous avons comparé ces séquences avec celle décrite dans le standard, et nous avons constaté que le nombre de retransmissions maximal ainsi que l'augmentation de la fenêtre de contention n'étaient pas respectés. De plus, ces implémentations passent beaucoup de temps à tenter de retransmettre avant d'initier leurs procédures de handover. Les retransmissions sont aussi utilisées dans les algorithmes d'adaptation de débits afin de déterminer si le lien se dégrade. Or, lors de la contention, le nombres de pertes augmente avec la plus forte probabilité de collisions. Afin d'observer l'impact des retransmissions sur les algorithmes d'adaptation de débits lors de la contention, nous avons mis en place un banc d'essai composé d'une trentaine de stations identiques. Nous avons constaté que l'algorithme de contrôle de débit utilisé est sous optimal par rapport à l'utilisation d'une modulation unique.

Enfin, nous nous sommes intéressés à l'utilisation d'un tel réseau à l'échelle de la ville afin d'acheminer des données ayant une forte validité spatiale. Nous avons alors proposé un schéma d'adressage géographique exploitant l'infrastructure d'Internet. Il permet à la fois d'obtenir un découpage hiérarchique du monde, et d'avoir un préfixe hiérarchique des adresses, similaire à celui de CIDR. Nous montrons que ce schéma d'adressage peut être utilisé dans des adresses multicast pour envoyer des messages à destination de zones géographiques précises (surface minimale d'un mètre carré).

Résumé en anglais

Wi-Fi is everywhere in cities, whether through the growing number of public access points, or the massive private access points deployment, in the form of set-top boxes for the major part. If we assume that all these access points are usable in order to allow any device to access the Internet, then we would potentially have network coverage throughout the city. This assumption led us to wonder if Wi-Fi could be used as a city-wide network.

This network could, more specifically, be used in a context of mobility. However, Wi-Fi was not designed to manage mobile users, and devices have to often change their access points when they no longer have connectivity. This mechanism, called handover, can be long because devices must first detect their connectivity losses before they can start looking for the next access point to associate with. It can be particularly long for devices such as smartphones because they are energy constrained and therefore do not apply an aggressive handover policy.

In this context we tried to characterize the possible Wi-FI applications for a moving user, considering the handover duration, the user speed and the access points density in the city. We found that for slow-moving users, the impact of the handover is small compared to the their overall connectivity. This allows them to use bandwidth-intensive applications as long as they are to some extend delay-tolerant. However, when the user's speed increases, the impact of handover's duration gradually degrades the user's connectivity, so that high speed users can no longer expect to use different access points. Fast moving devices spend more time performing handovers with new access points than transmitting application data.

Retransmissions play an important role in the duration of handover. In order to study in detail the retransmissions in 802.11, we have set up a testbed allowing us to observe the sequences of retransmitted messages using different implementations of 802.11 when we suddenly make the access point disappear. We compared these sequences with the one described in the standard, and we we found that the maximum number of retransmissions as well as the growth in the contention window were not respected. In addition, these implementations spend a lot of time trying to retransmit before initiating their handover procedures. Retransmissions are also used in the rate control algorithms to determine if the link is deteriorating. However, during contention, the number of losses increases with the higher probability of collisions. In order to observe the impact of retransmissions on the rate control algorithms during contention, we have set up a testbed composed of about thirty identical stations. We found that the rate control algorithm used underperforms compared to the use of a single modulation.

Finally, we proposed a geographical addressing scheme compliant with the Internet infrastructure. It allows to obtain both a hierarchical division of the world, and a hierarchical prefix for the addresses, similar to the one used in the CIDR format. We show that this addressing scheme can be used in multicast addresses to send messages to specific geographical areas (minimum area of one square meter).

Remerciements

Je tiens tout d'abord à remercier mon directeur de thèse Andrzej Duda et mon co-encadrant Franck Rousseau pour m'avoir accompagné tout au long de cette aventure. Je vous remercie pour la confiance que vous m'avez accordée, pour les conseils que vous m'avez donnés, et le soutien que vous m'avez apporté. J'ai eu la chance d'apprendre énormément à vos côtés.

Je tiens ensuite à remercier les membres du jury. Tout d'abord Mme Isabelle Guérin Lassous et M. Nicolas Montavont pour avoir accepté d'être rapporteurs de ma thèse. Ensuite M. Gentian Jakllari et M. Vivien Quéma pour avoir accepté d'évaluer mes travaux.

J'aimerais également remercier l'ensemble des membres de l'équipe Drakkar dans laquelle j'ai eu le plaisir de travailler durant ma thèse. J'y ai rencontré des personnes très enrichissantes avec lesquelles j'ai pu échanger (souvent autour d'un café) sur des sujets aussi divers que variés, allant du déploiement d'antennes Wi-Fi directionnelles à la Bastille, aux dernières lectures philosophiques et politiques.
J'ai une pensée particulière pour Henry-Joseph avec qui j'ai partagé mon bureau pendant l'intégralité de ma thèse, dans lequel nous avons pu discuter des raffinements de Python comme de l'encadrement de TD.
Je souhaiterais également remercier Étienne avec qui j'ai eu la chance de travailler sur WalT, et dont les discussions concernant l'IOCCC étaient passionnantes.

Je tiens à remercier mes amis géomaticiens que j'ai eu la joie de rencontrer, et dont la compagnie est toujours délicieuse ; notamment Anthony, David et Jacques, à qui j'ai l'honneur de donner le tempo.
J'aimerais également remercier Élodie, Colin et Timothy pour leur présence à chaque instant, dans les moments difficiles, mais aussi et surtout dans les moments de joie.
Je remercie également mes amis de Lyon et de Valence chez qui il est toujours agréable de se réfugier.
Un grand merci à Tristan pour son astuce, et à Anatole pour ses soirées de rires et bonne humeur.

Enfin, je tiens à remercier tout particulièrement mon frère et ma maman, François et Marie-Françoise pour leur soutien inconditionnel, passé, présent et futur.
Je pense également à toi, qui n'aura pas vu l'aboutissement de ce travail, qui t'aurait sûrement rendu fier de ton fils.

Table des matières

Table des matières 9

Liste des figures 11

Liste des acronymes 13

Glossaire 17

Liste des publications 21

Introduction 23
.1 Contexte et motivations . . . 23
.2 Contributions . . . 25
.3 Structure du document . . . 26

I La gestion de la mobilité en 802.11 29

1 – Un handover sur chaque couche 31
1.1 Le handover en 802.11 . . . 32
1.2 Le roaming aux niveaux réseau, transport et application . . . 52
1.3 Discussion générale . . . 59

II Contributions 61

2 – WalT 63
2.1 Bancs d'essai pour des expériences dans les réseaux sans fil . . . 65
2.2 Architecture de WalT . . . 66
2.3 Exemples d'expériences avec WalT . . . 71
2.4 Reproductibilité des expériences en fonction du temps : Transformer la plate-forme WalT en machine à remonter le temps . . . 73

3 – 802.11 à l'échelle de la ville 79
3.1 Vue d'ensemble . . . 81
3.2 Impact de la vitesse, du delta de connexion et de la densité sur la connectivité en mobilité . . . 85

4 – Analyse du comportement des implémentations de 802.11 lors de la perte de connectivité et de la contention 95
4.1 Comportement de différentes implémentations de 802.11 lors de la perte de connectivité . . . 96
4.2 Comportement des cartes TP-Link TL-WN722N lors de la contention . . . 100

5 – IP Geocast 113
5.1 Communications géocentriques pour DataTweet . . . 114
5.2 IP Geocast . . . 116
5.3 Discussion . . . 121

Conclusion générale 123

Liste des figures

6.1 Analogie avec le cycle de l'eau appliqué aux données 23

1.1 Recherche active 34
1.2 Association à un *point d'accès* ouvert (Les messages optionnels en cas de clé partagée sont représentés en gris) 35
1.3 Séquence d'authentification en 802.11I 42
1.4 Méthodes d'accès au canal lors de l'utilisation de plusieurs canaux en LTE-LAA 44
1.5 Déploiements LWA 46
1.6 Différenciation du trafic et accès au canal en utilisant EDCA (Fonction de coordination ajoutant la gestion de la QoS — *Enhanced Distributed Channel Access*) 47
1.7 Architecture d'*Hotspot 2.0* 48
1.8 Émission et réception d'un message en utilisant *Mobile* IPV4 52
1.9 Connexion *Multipath* TCP utilisant deux « sous-connexions » 55
1.10 Établissement d'une connexion utilisant QUIC (Connexion Internet rapide utilisant UDP — *Quick* UDP *Internet Connection*) 57
1.11 Frise temporelle de la récupération de la connectivité après un *handover* 802.11 59

2.1 Architecture fonctionnelle de *WalT* 67
2.2 Architecture spécialisée pour les réseaux de capteurs sans fil 68
2.3 Capture d'écran de Cooja représentant les échanges sur *WalT* 69
2.4 Scénario de la mesure de dérive d'horloge 71
2.5 Variation du décalage d'horloge entre deux nœuds et un référentiel temporel 71
2.6 Scénario de la mesure de *handover* 72
2.7 Durées de *handover* mesurées 73
2.8 API (Interface de programmation applicative — *Application Programming Interface*) du serveur *WalT* et de ses nœuds 75

3.1 Position géographique des *points d'accès* WI-FI (Marque déposée par la *Wi-Fi Alliance* qui regroupe l'ensemble des technologies radio reposant sur la norme IEEE 802.11) dans la ville de Grenoble 81
3.2 Delta de connexion lors du déplacement 84
3.3 Frises temporelles de connectivité le long d'un chemin pour différentes classes d'utilisateurs avec le passage d'un pont sans *point d'accès* sur le jeu de données généré et celui des « Freeboxes » 86
3.4 Taux de connectivité moyen sur la durée d'un trajet sur l'ensemble des chemins pour différentes classes d'utilisateurs et différentes valeurs de Δ_C 87
3.5 Distribution des 95% des durées de connexion par *point d'accès* pour chaque classe d'utilisateurs sur les deux jeux de données pour un $\Delta_C = 5s$ et sur l'ensemble des chemins 88
3.6 Distribution de la durée de non connexion pour chaque classe d'utilisateurs sur les deux jeux de données pour un Δ_C de $5s$ dans 95% des cas 89
3.7 Augmentation du nombre de *handovers* avec l'augmentation de la densité du nombre de *points d'accès* 90
3.8 Impact de la densité des *points d'accès* sur le taux de connectivité moyen pendant un déplacement 91

4.1 Frise temporelle représentant l'émission des 2 premiers paquets par NS-3 lors de la perte de connectivité. 97
4.2 Frises temporelles des retransmissions lors de la perte de connectivité pour différentes cartes — les pics bleus correspondent à l'émission d'un paquet UDP, les rouges à des retransmissions, les verts à l'émission d'un message de contrôle tel que RTS et les flèches marquent les *Probe Requests* 98

4.3 Distribution des délais entre chaque transmission pour les différents patterns étudiés. Ces délais sont agrégés suivant les valeurs de CW 99
4.4 Détail d'une retransmission pour certains patterns de différentes cartes 100
4.5 Nombre d'octets reçus par seconde au niveau du *point d'accès* lors de la montée en charge du nombre de *stations* tentant de saturer le lien — une nouvelle *station* démarre son émission de paquets UDP (Protocole de datagramme utilisateur — *User Datagram Protocol*) toutes les 60s — pour différentes modulations fixes et l'algorithme d'adaptation de débit intégré dans le microcode des cartes testées (hwrca) 102
4.6 Représentation des différentes étapes nécessaires à l'envoi d'un paquet de données et la réception de l'acquittement associé en 802.11G 103
4.7 Répartition des IFS (Intervalle inter-trames — *Interframe Space*) en réception au niveau du *point d'accès* en fonction du nombre de *stations* en concurrence 105
4.8 Répartition des IFS en réception au niveau du *point d'accès* pour les différentes queues EDCA avec une seule *station* saturant le lien 106
4.9 Modification du microcode concernant l'inversion des files AC_BK et AC_BE 107
4.10 Modification du pilote `ath9k_htc` concernant les tailles des fenêtres de contention 107
4.11 Répartition des IFS en réception au niveau du *point d'accès* lorsque le point d'accès n'annonce pas de support pour les WME (Extensions multimedia sans fil — *Wireless Multimedia Extensions*) 108
4.12 Répartition des IFS en réception au niveau du *point d'accès* pour les différentes queues EDCA avec une seule *station* saturant le lien après avoir modifié le microcode et le pilote `ath9k_htc` 109

5.1 Analogie avec le cycle de l'eau appliqué aux données 115
5.2 Architecture de *DataTweet* 116
5.3 Architecture d'*IP Geocast* 117
5.4 Partitionnement de l'espace 2D en utilisant un *quadtree* avec des *géopréfixes* obtenus en utilisant le schéma de codage de Morton 117
5.5 Partitionnement du monde en *quadtree* avec des exemples de *géopréfixes* pour certaines régions119
5.6 Structures des adresses IPv6 (6ème version du protocole Internet — *Internet Protocol version 6*) *multicast* et *geocast* 119
5.7 *Géoprefixe* de taille n 119
5.8 « Freeboxes » (points bleus) et arrêts de tramway (points rouges) dans le centre ville de Grenoble122

Liste des acronymes

3GPP Groupement d'organismes de standardisation pour les réseaux mobiles — *3rd Generation Partnership Project*. 45, 46

AIFS Intervalle inter-trames d'arbitrage — *Arbitration Interframe Space*. 103, 107

ANONCE Nombre aléatoire utilisé une seule fois par l'entité en charge de l'authentification — *Authenticator random Number used Once*. 42, 43

ANQP Protocole permettant aux *stations* d'interroger les *points d'accès* sur leurs politiques d'accès — *Access Network Query Protocol*. 49

API Interface de programmation applicative — *Application Programming Interface*. 11, 75, 83

APNS Service d'envoi de notifications d'Apple — *Apple Push Notification service*. 57

ARP Protocole de résolution d'adresse — *Address Resolution Protocol*. 39

AS Système autonome — *Autonomous System*. 114, 116, 121, 122

BE File d'attente correspondant au trafic « normal » avec une priorité normale — *Best Effort*. 47, 101, 105–107, 109, 110

BER Taux d'erreurs binaire — *Bit Error Rate*. 36

BGP Protocole pour les routeurs de bordures — *Border Gateway Protocol*. 114, 121

BK File d'attente correspondant au trafic d'arrière plan peu prioritaire — *Background*. 47, 106, 107, 109, 110

BSS Association d'un *point d'accès* et d'une ou plusieurs *stations* — *Basic Service Set*. 47, 49

BSSID Identifiant d'un BSS — *Basic Service Set Identifier*. 33, 36, 39, 47, 50, 80

CCA Évaluation d'un canal libre — *Clear Channel Assessment*. 45, 104

CIDR Routage inter domaines sans classes — *Classless Inter-Domain Routing*. 118, 119

CSA Annonce de changement de canal — *Channel Switch Announcement*. 50, 51

CSAT Écoute du canal pour adapter la transmission — *Carrier-Sensing Adaptive Transmission*. 44

CSMA/CA Écoute d'un support à accès multiple et à évitement de collision — *Carrier Sense Multiple Access/Collision Avoidance*. 47

CTS Disponible pour l'envoi — *Clear to Send*. 38, 45, 104, 110

DHCP Protocole de configuration dynamique des hôtes — *Dynamic Host Configuration Protocol*. 52, 72

DIFS Intervalle inter-trames utilisé par la fonction de coordination distribuée — *DCF Interframe Space*. 97, 103

DNS Système de noms de domaine — *Domain Name System*. 116

DTN Réseau tolérant au retard et à la coupure — *Delay-and-Disruption-Tolerant Networking*. 58

EAP Protocole extensible d'authentification réseau — *Extensible Authentication Protocol*. 42, 43, 49

EAP-AKA EAP utilisant l'USIM (UMTS *Subscriber Identity Module* afin d'authentifier un utilisateur — EAP *Authentication and Key Agreement*. 49

EAP-SIM EAP utilisant l'identifiant contenu dans la SIM afin d'authentifier un utilisateur — EAP *Subscriber Identity Module*. 49

EAP-TLS EAP utilisant un certificat utilisateur afin d'authentifier celui-ci — EAP *Transport Layer Security*. 49

EAP-TTLS Extension de EAP-TLS permettant de s'affranchir d'un certificat client, et d'utiliser un couple identifiant / mot de passe à la place — EAP *Tunneled Transport Layer Security*. 49

EDCA Fonction de coordination ajoutant la gestion de la QoS — *Enhanced Distributed Channel Access*. 11, 12, 47, 103, 104, 106, 108, 109

eNB Station de base utilisée en LTE — *Evolved Node B*. 44, 51

ERP-OFDM Extension de la couche physique afin de supporter des modulations OFDM — *Extended Rate Physical* OFDM. 103, 104, 107

ESS Ensemble de BSS appartenant à un même « lien logique » — *Extended Service Set*. 39, 43, 48, 52, 59

FAI Fournisseur d'accès Internet. 82, 116, 121, 122

FCM Service d'envoi de messages dans le cloud de Firebase — *Firebase Cloud Messaging*. 57

FLR Taux de pertes de trames — *Frame Loss Ratio*. 36

FMIPv6 IPv6 mobile avec *handovers* rapides — *Fast handovers for Mobile* IPv6. 53, 54

FTP Protocole de transfert de fichier — *File Transfer Protocol*. 46

GAS Service permettant le transport de trames dans un contexte non authentifié — *Generic Advertisement Service*. 49

GPS Géo-positionnement par satellite — *Global Positioning System*. 23, 26, 28, 36, 37, 43, 49, 80, 82, 114, 115, 117–120, 122

GTK Clé de groupe partagée éphémère — *Group Transient Key*. 42, 43

HESSID Ensemble de BSS reliés aux mêmes opérateurs — *Homogenous Extended Service Set*. 49

HMIPv6 IPv6 mobile hiérarchique — *Hierarchical Mobile* IPv6. 53, 54

HTTP Protocole de transfert de fichiers Hypertexte — *Hypertext Transfer Protocol*. 58

IEEE Institut des ingénieurs électriciens et électroniciens — *Institute of Electrical and Electronics Engineers*. 32, 49, 65, 72

IETF Force opérationnelle de l'ingénierie d'Internet — *Internet Engineering Task Force*. 58, 65

IFS Intervalle inter-trames — *Interframe Space*. 12, 105–109

IoT Internet des Objets — *Internet of Things*. 23, 65, 70, 114, 115, 123

IP Protocole Internet — *Internet Protocol*. 26, 28, 32, 52–58, 60, 72, 80, 81, 84, 85, 92, 114–117, 119, 121, 123

IPv4 4ème version du protocole Internet — *Internet Protocol version 4*. 3, 52

IPv6 6ème version du protocole Internet — *Internet Protocol version 6*. 3, 12, 36, 37, 49, 52–54, 114, 117, 119–122

ITS Systèmes de transports intelligents — *Intelligent Transportation Systems*. 46, 47

LBT Écoute avant transmission — *Listen Before Talk*. 44

LLDP Protocole de découverte sur la couche de liaison — *Link Layer Discovery Protocol*. 70

LTE Évolution à long terme — *Long Term Evolution*. 27, 44–46, 51, 59

LTE PDCP Protocole de convergence de paquets de données — *Packet Data Convergence Protocol*. 46

LTE-LAA LTE avec accès assisté par les bandes licenciées — LTE *License Assisted Access*. 44–46, 51

LTE-U LTE dans les bandes non licenciées — LTE *in Unlicensed spectrum*. 44–46, 51

LWA Agrégation LTE-WLAN — LTE-WLAN *Aggregation*. 46, 51

LWAAP Protocole d'adaptation pour LWA — LWA *Adaptation Protocol*. 46

M2M Machine à machine — *Machine to Machine*. 114

MAC Contrôle d'accès au support — *Medium Access Control*. 26, 32, 36, 38, 42, 46, 47, 50–53, 59, 60, 82, 103, 104

MCS Schéma de modulation et de codage — *Modulation & Coding Scheme*. 36

MIB Base d'information pour la gestion du réseau — *Management Information Base*. 47

MIC Code de vérification de l'intégrité d'un message — *Message Integrity Code*. 42

MIPv4 IPv4 mobile — *Mobile* IPv4. 52, 53, 60

MIPv6 IPv6 mobile — *Mobile* IPv6. 52, 53

MPTCP TCP multi chemins — *Multipath* TCP. 55, 56, 58, 60

MSK Clé maîtresse de session — *Master Session Key*. 42, 43

NAT Translation d'adresse réseau — *Network Address Translation*. 52

NDN Communication de données nommées — *Named Data Networking*. 120

NFS Système de fichiers sur le réseau — *Network File System*. 68, 70

NTP Protocole de temps sur le réseau — *Network Time Protocol*. 38, 71

OS Système d'exploitation — *Operating System*. 64, 66

OSI Interconnexion de systèmes ouverts — *Open Systems Interconnection*. 32, 52

PER Taux d'erreur de paquets — *Packet Error Rate*. 103

PIM SM Protocole *multicast* indépendant du protocole de routage utilisant un point de rendez-vous — *Protocol-Independent Multicast, Sparse Mode*. 121

PLCP Protocole de convergence de la couche physique — *Physical Layer Convergence Protocol*. 103

PMIPv6 IPv6 mobile utilisant un *proxy* — *Proxy Mobile* IPv6. 53, 54

PMK Clé maîtresse partagée — *Pairwise Master Key*. 42, 43

PoE Alimentation électrique par câble Ethernet — *Power over Ethernet*. 67, 68, 76

PTK Clé partagée éphémère — *Pairwise Transient Key*. 42, 43

PTP Protocole de synchronisation temporelle précise — *Precision Time Protocol*. 36, 71, 72, 74, 76

PyPI Gestionnaire de paquets *Python* — *Python Package Index*. 69

QoS Qualité de service — *Quality of Service*. 32, 33, 40, 43, 92, 103

QUIC Connexion Internet rapide utilisant UDP — *Quick* UDP *Internet Connection*. 11, 56–58, 60

RADIUS Service d'authentification d'utilisateurs à distance — *Remote Authentication Dial-In User Service*. 41

RAM Mémoire vive — *Random-Access Memory*. 68

RSA Algorithme de chiffrement asymétrique nommé par les initiales des trois inventeurs — *Rivest–Shamir–Adleman*. 54

RSSI Mesure de la puissance d'un signal reçu — *Received Signal Strength Indication*. 32, 35–38, 40, 50, 80, 81, 84, 85

RSU Équipement de bord de route — *Roadside Unit*. 47, 48

RTS Demande d'envoi — *Request to Send*. 38, 104, 110

RTT Durée d'un aller-retour — *Round-Trip Time*. 56, 57

SBC Ordinateur mono-carte utilisant un unique circuit intégré — *Single-Board Computer*. 64, 66, 68, 76

SCTP Protocole de transmission des commandes de flux — *Stream Control Transmission Protocol*. 56, 58

SIFS Intervalle inter-trames court — *Short Interframe Space*. 104, 107

SNMP Protocole simple de gestion de réseau — *Simple Network Management Protocol*. 70

SNONCE Nombre aléatoire utilisé une seule fois par l'entité en demandant à être authentifiée — *Supplicant random Number used Once*. 42, 43

SNR Rapport signal sur bruit — *Signal-to-Noise Ratio*. 32, 36, 38

SNS Service simple d'envoi de messages d'Amazon — *Simple Notification Service*. 57

SSID « Nom » d'un réseau WIFI — *Service Set Identifier*. 33, 36–39, 49, 80

TCP Protocole de contrôle de transmissions — *Transmission Control Protocol*. 52, 54, 55, 58, 80

TXOP Possibilité de transmettre pendant une fenêtre temporelle donnée — *Transmit Opportunity*. 45, 107

UDP Protocole de datagramme utilisateur — *User Datagram Protocol*. 12, 52, 54–56, 58, 72, 101–103

URI Identifiant de ressource uniforme — *Uniform Resource Identifier*. 120

VANET Réseau Ad-Hoc de véhicules — *Vehicular Ad-Hoc Network*. 116

VI File d'attente correspondant au trafic vidéo avec une priorité élevée — *Video*. 47, 101, 106, 107, 109

VLAN Réseau local virtuel — *Virtual Local Area Network*. 70

VO File d'attente correspondant au trafic transportant de la voix avec une priorité très élevée — *Voice*. 47, 101, 106, 107, 109

VOIP Voix sur IP — *Voice over IP*. 41, 46, 92, 123

W3C Consortium du World Wide Web — *World Wide Web Consortium*. 58

WAN Réseau étendu — *Wide Area Network*. 49

WEP Protocole de sécurisation des réseaux sans fil visant à fournir une protection équivalente à un réseau filaire (obsolète) — *Wired Equivalent Privacy*. 35

WI-FI Marque déposée par la *Wi-Fi Alliance* qui regroupe l'ensemble des technologies radio reposant sur la norme IEEE 802.11. 11, 24, 32, 45, 48, 51, 67, 70–72, 79–85

WLAN Réseau local sans fil — *Wireless Local Area Network*. 46, 52

WME Extensions multimedia sans fil — *Wireless Multimedia Extensions*. 12, 101, 103, 104, 108

WPA2 2ème version de WPA — WI-FI *Protected Access II*. 49

WPA-PSK Accès protégé utilisant WPA avec une clé partagée — WPA *Pre-Shared Key*. 42

WT Terminaison WLAN — WLAN *Termination*. 46

XW UP Protocole Xw sur le plan utilisateur servant à échanger des messages de contrôle concernant les flots de données utilisateurs — XW *User Plane*. 46

Glossaire

802.11 Spécifications de l'IEEE des couches MAC et PHY pour les communications sans fil dans les réseaux locaux. 9, 11, 24–27, 31–33, 35, 36, 38, 40–42, 44–46, 48, 50–52, 54–56, 58–60, 63, 65, 72, 80–82, 93, 95, 96, 101, 103, 115, 118, 123, 124

802.11AC WI-FI *5* : Amendement au standard 802.11 permettant d'augmenter les débits de transmission notamment grâce à l'utilisation de la bande des $5GHz$, de plusieurs flux spatiaux et de meilleurs modulations. 44

802.11E Amendement au standard 802.11 proposant des améliorations en regard de la QoS. 47

802.11G Amendement au standard 802.11 permettant d'augmenter le débit en introduisant OFDM dans la bande des $2.4GHz$. 12, 97, 101, 103

802.11I *Robust Security Network* : Amendement au standard 802.11 introduisant les services d'authentification de 802.1X et est communément appelé WPA2. 11, 42

802.11K *Radio Resource Management* : Amendement au standard 802.11 qui permet d'annoncer des informations concernant l'environnement radio et l'état du réseau WLAN afin d'améliorer sa gestion en cas de mobilité. 36, 39, 43, 44, 48, 50

802.11N WI-FI *4* : Amendement au standard 802.11 permettant d'augmenter les débits de transmission grâce à l'utilisation de plusieurs antennes MIMO et la possibilité d'agréger des trames au niveau MAC. 44

802.11P *Wireless Access in Vehicular Environments* : Amendement au standard qui définit des techniques afin de permettre aux véhicules de communiquer avec des équipements en bord de route. 46, 47, 51, 60

802.11R *Fast BSS Transition* : Amendement au standard 802.11 permettant à des *stations* mobiles de rester connectées au sein d'un même domaine WLAN. 43, 44, 50, 59

802.11U *Interworking with External Networks* : Amendement au standard 802.11 enrichissant les données transmises dans les *beacons* avec des informations concernant entre autres le réseau d'opérateurs, les possibilités de *roaming* et les modalités d'association. Il introduit aussi un mécanisme permettant aux *stations* d'interroger les *points d'accès* sur leurs caractéristiques. 48, 49

802.11V *Wireless Network Management* : Amendement au standard 802.11 qui introduit un ensemble de règles pour la gestion du réseau WLAN. 35, 43, 44, 50

802.15.4 Spécifications de l'IEEE des couches MAC et PHY pour les communications sans fil dans les réseaux sans fil à faible portée et contraints en énergie. 65, 70

Association Request Trame émise par une *station* suite à une authentification réussie afin d'initialiser son association auprès d'un *point d'accès*. 35, 41

Association Response Trame émise par un *point d'accès* en réponse à une *Association Request* afin de finaliser l'association d'une *station* auprès de celui-ci. 35

Authentication Challenge Trame émise afin d'initier une connexion sécurisée par le protocole WEP. 35

Authentication Challenge Response Trame émise suite à la réception d'une trame *Authentication Challenge*. 35

Authentication Request Trame émise par une *station* afin d'initialiser l'authentification auprès d'un *point d'accès*. 35

Authentication Response Trame émise par un *point d'accès* en réponse à une *Authentication Request*. 35

backoff Mécanisme consistant à rajouter une durée aléatoire avant une transmission afin de réduire le nombre de collisions potentielles. 45

beacon Trame de management envoyée périodiquement par un point d'accès IEEE 802.11 afin d'annoncer la présence d'un réseau WLAN. 33, 36, 38, 40, 80

Diameter Protocole d'authentification et d'autorisation d'utilisateurs. 41

disassociation Dé-association d'une *station* de son BSS initiée par l'émission d'une trame *Disassociation Request* par l'une des deux parties. 36

Disassociation Request Trame émise par un *point d'accès* ou une *station* afin de demander une dé-association au réseau sans fil. 35

Docker Outil de virtualisation légère basée sur l'emploi de conteneurs. 26, 64, 66–68, 70, 74, 76, 124

EtherType Champ d'une trame *Ethernet* spécifiant le protocole de la couche supérieure contenue dans le champ donnée. 46

femtocell Petite *station de base* de faible puissance en général déployée pour un usage domestique. 32

handover Procédure permettant à une *station* WI-FI de se connecter d'un point d'accès à un autre lors de la perte de connectivité. 11, 24–27, 32, 33, 35–41, 43, 44, 47, 49–52, 54–56, 59, 60, 63, 72, 73, 79, 80, 83, 84, 87–90, 92, 110, 111, 123

Hotspot 2.0 Spécification technique de la *Wi-Fi Alliance* visant à améliorer le partage d'informations concernant l'environnement radio pour des équipements n'appartenant pas au même ESS. 48–51, 59, 60, 80

MaxChannelTime Durée maximale pendant laquelle une station doit attendre la réception d'une ou plusieurs trames *Probe Responses* après avoir déjà reçu une trame *Probe Response* suite à l'émission d'une trame *Probe Request* sur le canal. 33, 37, 39

MinChannelTime Durée minimale pendant laquelle une station doit attendre la réception d'une trame *Probe Response* après l'émission d'une trame *Probe Request* sur le canal. 33, 37, 38

multicast Emission d'un paquet depuis une source unique vers un ensemble de destinations. 12, 26, 28, 117, 119–122, 124

multihoming Pratique consistant à relier un appareil à plusieurs réseaux. 41

offloading Technique consistant à utiliser un réseau secondaire pour acheminer des données afin de réduire la charge sur le réseau principal. 46

point d'accès Équipement WI-FI permettant aux *station* WI-FI de se connecter à celui-ci afin de les relier au réseau local. 11, 12, 23–27, 32, 33, 35–43, 45–52, 54–56, 59, 60, 72, 80–93, 96, 97, 100–106, 108–110, 114, 115, 123, 124

Probe Request Trame de management envoyée par une *station* afin de découvrir les points d'accès dans son environnement. 11, 33, 37–39, 48, 49, 98

Probe Response Trame de management envoyée par un *point d'accès* en réponse à la réception d'une trame *Probe Request* correspondant à ses critères. 33, 36, 37, 39, 40, 80

quadtree Structure de données hiérarchique dans laquelle chaque entité possède quatre fils. 12, 26, 28, 117–120, 122, 123

Raspberry Pi Ordinateur mono-carte basé sur une architecture ARM produit par la *Raspberry Pi Foundation*. 66, 68, 71, 72, 101, 104

Signal Extension Durée de $6\mu s$ nécessaire après l'envoi d'une trame modulée en OFDM afin de permettre le décodage de celle-ci. 103, 104

Sleep Request Trame émise par une *station* afin d'annoncer à son *point d'accès* que celle-ci va passer en économie d'énergie et qu'il doit retarder la transmission des paquets en direction de celle-ci. 38

station Équipement WI-FI se connectant à un *point d'accès* afin de se connecter au réseau local. 12, 25, 26, 32, 33, 35–43, 45, 47–52, 54, 55, 59, 60, 96, 101–107, 109, 123, 124

station de base Équipement auquel les équipements de téléphonie mobile s'associent afin de rejoindre le réseau. 44–46, 51

unicast Emission d'un paquet depuis une source unique vers une destination unique. 120, 121

WalT Plate-forme d'expérimentation reproductible. 11, 26, 27, 64–77, 101, 124

Liste des publications

Pierre BRUNISHOLZ, Etienne DUBLÉ, Franck ROUSSEAU et Andrzej DUDA : WalT : A reproducible testbed for reproducible network experiments. *In Computer Communications Workshops (INFOCOM WORKSHOPS), 2016 IEEE Conference on*, pages 146–151. IEEE, 2016.

Etienne DUBLÉ, Pierre BRUNISHOLZ, Franck ROUSSEAU et Andrzej DUDA : Time-Independent Experiment Reproducibility : Turning the WalT Platform into a Time Machine. *In Proceedings of the 13th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks*, pages 67–72. ACM, 2016.

Pierre BRUNISHOLZ, Franck ROUSSEAU et Andrzej DUDA : DataTweet for user-centric and geo-centric IoT communications. *In Proceedings of the 2nd Workshop on Experiences in the Design and Implementation of Smart Objects*, pages 29–34. ACM, 2016.

Pierre BRUNISHOLZ, Franck ROUSSEAU et Andrzej DUDA : Anatomie des retransmissions dans les implémentations de 802.11. *In Rencontres Francophones sur la Conception de Protocoles, l'Évaluation de Performance et l'Expérimentation des Réseaux de Communication*, 2018.

Introduction

Contexte et motivations

La croissance de l'IoT (Internet des Objets — *Internet of Things*) fait que de plus en plus d'objets connectés ont fait irruption dans notre quotidien. Ils sont déployés aussi bien à de petites échelles, avec les « maisons intelligentes », qu'à de plus grandes échelles, avec les « villes intelligentes ». De par leur proximité avec les utilisateurs et leurs lieux de vie, et de par leurs capacités à communiquer sur Internet, ils permettent l'émergence de nouvelles applications.

Ces nouvelles applications peuvent exploiter les caractéristiques offertes par les *smartphones* : ils ont tout d'abord modifié le rapport que les utilisateurs entretiennent avec Internet, en leur permettant un accès omniprésent à celui-ci. La majeure partie du trafic Internet émanant d'utilisateurs s'effectue d'ailleurs par le biais de ces équipements [1]. Ensuite, le nombre de capteurs que ces appareils embarquent ne cesse d'augmenter. Il n'est donc pas rare de trouver des capteurs de température, des accéléromètres, des gyroscopes ou des puces GPS (Géo-positionnement par satellite — *Global Positioning System*), pour ne citer que quelques exemples, au sein des *smartphones*. Il est alors possible d'utiliser ces capteurs afin d'effectuer de la collecte de données, données qui seront ensuite analysées dans le *cloud*.

Les *smartphones* permettent donc de consommer des données se trouvant sur Internet, tout en générant d'autres données pouvant être utilisées par d'autres acteurs se trouvant ailleurs sur Internet.

Dans l'idéal, afin de pouvoir collecter ou consommer des données, il faudrait être en mesure d'utiliser n'importe quel réseau sans fil environnant.

Cette thèse s'inscrit au sein du projet ANR *DataTweet* qui part du constat que les communications dans le cadre de l'IoT manquent d'unicité de par la multitude des technologies sans fil existantes [2]. Il est alors question de définir un protocole de communication. Celui-ci a pour principale caractéristique de ne plus définir la destination d'un message par l'adresse d'une machine en particulier, mais plutôt d'utiliser la donnée contenue dans le message afin de savoir où dans le réseau celle-ci sera d'utilité. Ceci permet alors aux différents appareils cherchant à envoyer une donnée de ne plus se soucier ni de la destination, ni du type de réseau sans fil à utiliser. La donnée aurait uniquement besoin d'être empaquetée dans un message *DataTweet* et d'être envoyée dans l'air par n'importe quelle interface sans fil présente sur l'équipement.

Un tel mécanisme permet alors de remonter des données dans le *cloud*, et également d'en diffuser auprès d'utilisateurs et d'équipements en ayant exprimé le besoin. Il est possible de se représenter cette utilisation des données comme une sorte de cycle de l'eau, illustré sur la FIGURE 6.1, dans lequel des données « brutes » sont tout d'abord générées par des capteurs et envoyées dans l'air. Celles-ci sont alors récupérées par un *point d'accès* environnant qui est en charge d'initier le relai de ces données vers des centres de calcul capables de stocker et d'analyser ces mêmes données. Ces centres sont alors aussi susceptibles de

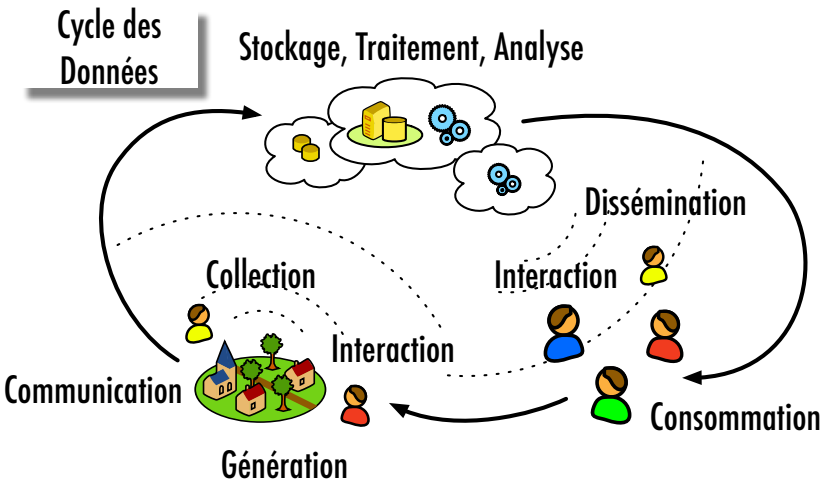


FIGURE 6.1 – Analogie avec le cycle de l'eau appliqué aux données

générer des données enrichies à partir des données brutes recueillies et peuvent alors les envoyer vers des utilisateurs ou équipements ayant un intérêt pour ces données enrichies.

Nous sommes alors parti du postulat que le WI-FI était omniprésent au sein des villes. Les *points d'accès* publics se démocratisent et leurs déploiements s'accélèrent, mais surtout de nombreux *points d'accès* privés sont déjà déployés sous la forme de boxes d'opérateurs par exemple. Supposons que l'ensemble de ces *points d'accès* soient utilisables par n'importe quel utilisateur, nous serions en présence d'une infrastructure de réseau sans fil déjà déployée, à l'échelle d'une ville, qui pourrait potentiellement offrir une couverture totale de celle-ci. Des utilisateurs pourraient alors profiter de ces *points d'accès* lors de leurs déplacements. Or, la mobilité en 802.11 n'est toujours pas assurée pour ce type de déploiement constitué de *points d'accès* « autonomes ». Contrairement à des déploiements de type infrastructure, dans lesquels une partie de la gestion du *handover* peut être déléguée aux différents *points d'accès* [3, 4], la gestion du *handover* est ici entièrement faite au niveau de l'équipement mobile. Celui-ci doit d'abord constater sa perte de connectivité avant de chercher un nouveau *point d'accès* auquel s'associer. De plus, les équipements mobiles comme les *smartphones* étant contraints en énergie, ils n'appliquent pas de politique de *handover* agressives [5], entrainant de fait des durées de déconnexion longues. Un accès sans interruptions à ce réseau semble donc compromis.

Il faut alors extraire les caractéristiques, en terme de connectivité, que permet l'utilisation de l'ensemble des *points d'accès* d'une ville. Ceci permet à la fois de mettre en évidence les problèmes inhérent à la durée de *handover*, tout en permettant de définir le type de trafic applicatif capable de fonctionner dans ces conditions. Certaines applications pourraient alors bénéficier des *points d'accès* présents en ville tout en déchargeant les autres réseaux habituellement utilisés pour leurs fonctionnements.

Dans ce cadre, nous proposons tout d'abord d'étudier la couverture 802.11 offerte dans les villes, afin de pouvoir déterminer si 802.11 est un bon candidat pour une utilisation en mobilité. Pour effectuer une telle analyse il faut prendre en compte la vitesse de déplacement d'un utilisateur, la durée de *handover* de son équipement ainsi que la quantité de *points d'accès* présents dans son entourage. Nous proposons alors un environnement de simulations basé sur des données réelles de positions de *points d'accès*, mais aussi basé sur des données enrichies permettant d'augmenter la densité du nombre de *points d'accès* tout en considérant la topologie de la ville. Ceci nous permet alors d'estimer la couverture offerte ainsi que le taux de connectivité pour différentes classes d'utilisateurs.

Il est intéressant ensuite de se pencher sur les causes de la longévité du *handover*. Pour cela nous proposons d'étudier le fonctionnement des retransmissions en 802.11 lors de la perte de connectivité. Nous avons constaté deux choses : 1. les différentes cartes 802.11 testées retransmettent pendant une période assez longue avant d'initier leur procédure de *handover* et 2. elles ne respectent pas forcément le standard en termes de nombre de retransmissions et de respect des fenêtres de contention. Lors de la contention, des retransmissions sont aussi présentes à cause du nombre croissant de collisions. Nous avons alors fait une analyse des débits observés lors d'une utilisation de 802.11 en contention, lorsque l'algorithme de contrôle de débit est activé. Nous avons constaté que lors de la contention le débit peut drastiquement chuter par rapport à une utilisation à débit fixé, ainsi qu'un défaut d'implémentation concernant les files de priorités et les fenêtres de contention sur les pilotes et microcodes ATH9K.

Finalement, nous nous sommes intéressés à l'acheminement de données avec une forte dépendance spatio-temporelle sur un réseau à l'échelle de la ville. Si nous considérons la diffusion de données vers des utilisateurs, certaines données ont une forte validité spatiale. Par exemple si l'on souhaite diffuser les horaires de passage d'un tramway, il est judicieux de cibler les *points d'accès* à proximité des arrêts. Or, il est difficile d'adresser une zone géographique particulière en exploitant la topologie d'Internet. Nous proposons alors un schéma d'adressage géographique hiérarchique permettant de résoudre ce problème.

Contributions

Le travail réalisé s'organise autour de quatre contributions principales :

- Un environnement de simulations permettant d'évaluer le taux de connectivité offert par 802.11 aux utilisateurs dans un contexte urbain.

  Pour cela nous avons défini un modèle de mobilité définissant plusieurs classes d'utilisateurs ayant chacune des vitesses différentes, allant de la vitesse d'un piéton à un déplacement plutôt rapide dans un contexte urbain. Nous avons aussi défini plusieurs chemins prenant en compte la topologie de la ville afin que ceux-ci soient le plus réaliste possible.

  Nous utilisons deux jeux de données pour les positions des *points d'accès*. Le premier est un jeu de données réel contenant les positions géographiques des « Freeboxes » dans la ville de Grenoble. Le deuxième est un jeu de données que nous générons nous-mêmes. La méthode de génération prend elle aussi en compte les données topologiques de la ville afin notamment de positionner les *points d'accès* dans des bâtiments. Ceci nous permet d'obtenir un jeu de données avec une plus forte densité de *points d'accès* pouvant correspondre à l'ensemble des *points d'accès* des différents opérateurs français.

  Enfin nous définissons un modèle de connectivité. Celui-ci prend en compte la portée des *points d'accès* ainsi que le *handover*. Concernant ce dernier nous définissons un delta de connectivité Δ_C englobant la phase de détection de la perte de connectivité, la phase de recherche, le *handover* à proprement parler ainsi que la récupération de la connectivité.

  Les résultats de ces simulations montrent que la durée nécessaire à la récupération de la connectivité influe d'autant plus qu'un utilisateur se déplace rapidement. Ainsi, avec une valeur de Δ_C réaliste, il est envisageable d'avoir un taux de connectivité sur plus de 80% du trajet d'un utilisateur piéton alors qu'il est quasi nul pour un utilisateur se déplaçant en voiture.

  Nous constatons aussi que les durées passées sur un même *point d'accès* sont de plusieurs dizaines de secondes pour les piétons, mais diminuent assez vite lorsque la vitesse de déplacement augmente.

  Enfin les durées de déconnexions durent la plupart du temps la durée Δ_C pour les piétons et augmentent avec la vitesse car ces utilisateurs n'arrivent plus à s'associer à temps avant de sortir de la portée radio du *point d'accès* voulu.

  Nous avons alors conclu que pour des utilisations de collecte de petites données, relativement tolérantes à la latence, il était possible d'utiliser un tel réseau 802.11 avec des utilisateurs se déplaçant modérément vite. Les piétons peuvent cependant même envisager des applications plus gourmandes en bande passante car ils séjournent plus longtemps sur un même *point d'accès*. Les durées de déconnexions sont tout de même trop longues pour envisager des applications nécessitant de l'interactivité.

- Une étude du comportement de différentes cartes 802.11 lors de la perte de connectivité et de la contention.

  Lors de la perte de connectivité caractérisée par le non acquittement des trames émises, les stations utilisent le mécanisme de retransmission. Ce n'est qu'après un nombre de retransmissions non acquittées donné que les stations initient la procédure de *handover*. C'est pourquoi nous nous intéressons au mécanisme de retransmission, qui sert initialement à limiter l'impact des collisions et de la dégradation du signal.

  Nous avons mis en place un banc d'essai nous permettant de mesurer les intervalles inter-trames des paquets émis par une *station* lorsque l'on arrête brutalement son *point d'accès*.

  Nous comparons alors la séquence de trames retransmises décrite dans le standard 802.11 à celle observée sur différentes implémentations de cartes 802.11 grand public lors de la perte de connectivité.

  Nous constatons ici que les motifs — ou *patterns* — de retransmissions ne suivent pas ceux décrits dans le standard en termes de nombre de retransmissions pour chaque paquet et en termes de fenêtres de contention.

  Nous constatons aussi que lors de la perte de connectivité, les cartes essaient de retransmettre un nombre de paquets différents très important pendant plusieurs secondes avant d'initier une procédure de *handover*.

Par ailleurs, les cartes 802.11 utilisent des algorithmes de contrôle de débits afin d'adapter les modulations utilisées à la qualité du lien. Or, lorsque le nombre de stations augmente, les collisions sont plus fréquentes, augmentant dès lors le nombre de retransmissions sans que celles-ci soient liées à une dégradation du bilan de liaison.

Nous avons mis en place un banc d'essai composé d'une trentaine de *stations* équipées de cartes 802.11 identiques, nous permettant de mesurer l'évolution du débit ainsi que les intervalles inter-trames au niveau d'un *point d'accès* lors de l'augmentation du nombre de *stations*.

Nous constatons que l'utilisation d'un algorithme de contrôle de débit aboutit à un débit bien plus bas que lorsque les cartes utilisent une modulation fixée.

Par ailleurs l'observation des intervalles inter-trames nous a permis de mettre en évidence des anomalies concernant les fenêtres de contention et les files de priorité de trafic dans les pilotes et micro-codes des cartes étudiées.

- Un schéma d'adressage géographique hiérarchique s'adaptant à la topologie d'Internet.

  Il est difficile d'utiliser Internet lorsque l'on souhaite envoyer des messages dans une zone géographique précise. La topologie d'Internet n'est pas liée à une topologie géographique réelle, imposant alors des méthodes compliquées afin de récupérer les adresses de machines situées dans une zone géographique donnée.

  Nous proposons alors d'effectuer un découpage géographique du monde en *quadtree*. Il s'agit d'un découpage permettant d'obtenir quatre zones de tailles équivalentes à partir d'une zone de départ englobante. Ces zones pouvant elles aussi être redécoupées, nous pouvons obtenir un nombre arbitrairement important de zones de plus en plus petites.

  Nous utilisons les codes de Morton afin d'obtenir un découpage en *quadtree* du monde se basant sur l'encodage binaire des coordonnées GPS. Il permet alors d'obtenir un préfixe binaire partagé entre différentes sous-zones, à l'instar du schéma d'adressage IP (Protocole Internet — *Internet Protocol*) et de l'utilisation de masques de sous-réseaux.

  Nous proposons d'utiliser l'encodage des coordonnées GPS dans des adresses *multicast*s permettant de définir un nombre de bits à considérer afin de définir un préfxe valide, et donc la taille de la zone géographique à adresser.

- Une plateforme permettant la reproductibilité et la répétabilité des expérimentations sur les technologies sans fil.

  Au sein de l'équipe nous avons développé *WalT*, une plateforme basée sur *Docker* permettant d'orchestrer des expérimentations avec un fort degré de reproductibilité et de répétabilité.

  Pour arriver à ce but elle utilise des composants standard et peu onéreux permettant à n'importe qui de reproduire la plateforme dans ses locaux. Seul l'environnement radio et la topologie des nœuds déployés différeraient, mais nous pensons qu'afin de valider un résultat, l'expérimentation dans des environnements différents est nécessaire, et permet d'appuyer le côté reproductible et répétable de celle-ci.

  L'utilisation de *Docker* permet par ailleurs d'assurer que l'intégralité de l'environnement logiciel d'une expérience est sauvegardé et peut être utilisé sur une autre plateforme *WalT*.

  Nous avons mis en place une plateforme *WalT* mobile, permettant de rejouer une même expérience dans un environnement différent.

Structure du document

Le manuscrit est organisé en deux parties : la première présente les travaux réalisés pour améliorer la connectivité en mobilité, notamment concernant la couche MAC (Contrôle d'accès au support — *Medium Access Control*) 802.11, mais aussi les couches supérieures, la seconde partie présentera les différentes contributions.

Partie I, Chapitre 1 — Un handover sur chaque couche Nous commencerons par présenter le fonctionnement du *handover* 802.11 au niveau MAC et les phases le caractérisant : déclenchement, découverte, sélection et (ré-)association.

Nous présenterons ensuite des améliorations cherchant à réduire la durée que prend un *handover* en respectant le standard — c'est-à-dire les solutions qui sont soit proposées dans le standard lui même, soit qui ne modifient pas lourdement celui-ci.

Nous verrons ensuite des approches plus radicales afin d'introduire une meilleure gestion de la mobilité. Ces solutions pouvant aller de propositions du standard afin de communiquer sans association à l'introduction de la LTE (Évolution à long terme — *Long Term Evolution*) dans les bandes de fréquences utilisées par 802.11.

La gestion de la mobilité devant être effectuée sur l'ensemble des couches du modèle OSI nous présenterons ensuite différents mécanismes utilisés sur les couches réseau, transport et application afin de limiter son impact.

Enfin nous discuterons de manière générale de l'ensemble de ces solutions.

Partie II, Chapitre 2 — WalT Dans ce chapitre nous présenterons *WalT*, une plateforme d'expérimentations reproductibles développée au sein du laboratoire.

Nous verrons dans un premier temps les enjeux liés à la reproductibilité et la répétabilité des expérimentations, puis nous présenterons dans le détail le fonctionnement de la plateforme ainsi que les mécanismes mis en jeu afin d'assurer cette reproductibilité et répétabilité.

Nous aborderons ensuite les problématiques liées à la pérennisation dans le temps des expérimentations. En effet comment assurer la reproductibilité et la répétabilité d'une expérimentation faite plusieurs années auparavant sur un système ayant pu évoluer depuis ? Nous présenterons alors les développements envisagés de la plateforme afin de pouvoir s'adapter à ces cas de figure.

Partie II, Chapitre 3 — 802.11 à l'échelle de la ville Nous évaluons dans ce chapitre les possibilités offertes par l'utilisation de l'ensemble des *points d'accès* 802.11 présents dans les villes pour des utilisateurs mobiles.

Pour cela nous présenterons dans un premier temps le simulateur que nous avons mis en place. Nous verrons comment nous avons mis en place une méthode de génération réaliste de chemins et de *points d'accès* en prenant en compte les données topologiques de ville. Nous détaillerons le concept de delta de connectivité, correspondant à la durée totale pendant laquelle un utilisateur est déconnecté. Enfin nous présenterons les différentes classes d'utilisateurs envisagées, ces derniers étant caractérisés par leur vitesse de déplacement.

Nous évaluerons ensuite l'impact de la vitesse, de la durée de *handover* et de la densité du nombre de *points d'accès* afin d'évaluer le taux de connectivité d'un utilisateur, son temps de connexion moyen à un *points d'accès* ainsi que sa durée de déconnexion moyenne entre deux associations. Ceci nous permettra d'établir les usages possibles sur un tel déploiement, ainsi que les paramètres à améliorer afin d'assurer une meilleure connectivité lors de la mobilité.

Partie II, Chapitre 4 — Analyse du comportement des implémentations de 802.11 lors de la perte de connectivité et de la contention Dans ce chapitre nous présentons dans un premier temps le comportement attendu lors de la perte de connectivité en terme de retransmissions. Nous verrons alors que les différentes implémentations de 802.11 des cartes que nous avons testées ne respectent pas toujours le standard : 1. en terme de nombre de retransmissions avant de passer au paquet suivant et 2. en terme d'augmentation des fenêtres de contention. De plus, nous constaterons que les cartes retransmettent pendant des périodes de temps longues avant d'entamer leur procédure de *handover*.

Dans un second temps nous nous intéresserons au comportement de cartes 802.11 lors de la contention. Nous constaterons que l'utilisation d'algorithmes de contrôle de débits aboutit à une utilisation sous optimale du réseau. En effet, les cartes étudiées réduisent leurs modulations lorsque les pertes augmentent. Or, en contention le nombre de collisions augmente aussi, mais réduire le débit n'influe pas sur les probabilités de collisions, contrairement à l'augmentation des fenêtres de contention. Lors de l'analyse des intervalles inter-trames nous verrons que nous avons constaté des anomalies concernant les files de priorités ainsi que les valeurs des fenêtres de contention au sein des pilotes et microcodes des cartes testées pouvant impacter les performances.

Partie II, Chapitre 5 — IP Geocast Dans ce dernier chapitre nous proposons un mécanisme d'adressage géographique afin de pouvoir relayer des messages en exploitant la topologie d'Internet existante.

Nous verrons comment nous utilisons les codes de Morton afin d'encoder les coordonnées GPS de manière à obtenir un découpage géographique du monde en *quadtrees*, découpage hiérarchique récursif qui permet à une zone d'être divisée en quatre « sous-zones » de taille équivalente. De plus, l'encodage utilisant les codes de Morton permet d'obtenir des adresses ayant une structure hiérarchique similaire au fonctionnement des adresses IP et des masques de sous-réseaux. L'application d'un masque permettra ici d'adresser une zone des différents *quadtrees* plus ou moins précise — de descendre plus ou moins profondément dans la hiérarchie de ce découpage.

Nous présenterons ensuite comment nous envisageons d'inclure ces adresses dans des adresses *multicast*s, et comment celles-ci seront utilisées au sein de l'infrastructure d'Internet.

Enfin une conclusion générale permettra de clôturer ce manuscrit, faisant un récapitulatif des différentes contributions et présentant les travaux futurs envisagés.

Première partie

La gestion de la mobilité en 802.11

Un handover sur chaque couche

Sommaire

1.1 Le handover en 802.11 32
1.1.1 Fonctionnement du handover 32
1.1.1.1 Déclenchement 32
1.1.1.2 Découverte 33
1.1.1.3 Sélection, authentification et (ré-)association 35
1.1.2 Améliorations en regard du standard 35
1.1.2.1 Déclenchement 35
1.1.2.2 Découverte 37
1.1.2.3 Sélection 40
1.1.2.4 Authentification et Association 41
1.1.2.5 Discussion 43
1.1.3 Propositions au-delà du standard 802.11 44
1.1.3.1 LTE sur les bandes non licenciées 44
1.1.3.2 Propositions du standard 802.11 46
1.1.3.3 Déconstruction de la notion de point d'accès 50
1.1.3.4 Discussion 51
1.2 Le roaming aux niveaux réseau, transport et application 52
1.2.1 Le roaming IP 52
1.2.1.1 Mobile IP 52
1.2.1.2 Host Identity Protocol 54
1.2.2 Le roaming au niveau transport 54
1.2.2.1 Multipath TCP 55
1.2.2.2 Stream Control Transmission Protocol 56
1.2.2.3 QUIC 56
1.2.3 Le roaming au niveau application 57
1.2.4 Discussion 58
1.3 Discussion générale 59
1.3.1 La mobilité dans les villes 59
1.3.2 Un handover au niveau MAC toujours important 59
1.3.3 La récupération de données sans association 60

Introduction

Selon une étude de Cisco [6] les *smartphones* — et *phablettes* — représentent 50% du parc des équipements mobiles, mais génèrent 88% du trafic total des équipements mobiles. Selon la même étude, 60% de ce trafic total a été *offloadé* en utilisant le WI-FI ou des *femtocells* représentant 10,7 exabytes de données chaque mois. Selon leurs prévisions, en 2020, 49% du trafic IP total sera effectué par des équipements WI-FI, 60% de ceux-ci seront des smartphones.

Cependant la gestion de la mobilité en WI-FI ne permet pas une expérience utilisateur transparente de par ses mauvaises performances. Dans une étude récente [7], plusieurs terminaux 802.11 ont été utilisés afin de mesurer le temps de déconnexion lorsqu'un utilisateur passe d'un *point d'accès* à un autre. Ils observent que lorsqu'un terminal subit une perte de connectivité, la durée avant laquelle il récupère celle-ci en s'associant à un nouveau *point d'accès* est comprise entre 5s et 15s sur les terminaux testés.

Cette gestion de la mobilité est appelée *handover*. Dans l'idéal, le *handover* devrait permettre à un équipement mobile de conserver sa connectivité sans avoir de discontinuité ressentie. Dans les faits, atteindre un tel but est relativement complexe car le *handover* peut impacter chaque couche du modèle OSI (Interconnexion de systèmes ouverts — *Open Systems Interconnection*).

Ce chapitre est organisé en deux sections présentant le *handover* appliqué aux équipements 802.11 sur l'ensemble de la pile OSI :

- La première section présente le *handover* de la couche MAC de 802.11 et des améliorations proposées afin de rendre celui-ci le plus fluide possible.
- La deuxième section présente des solutions afin d'assurer la mobilité aux niveaux réseau, transport et application.

1.1 Le handover en 802.11

Nous nous intéressons ici au *handover* de la couche MAC 802.11. De manière simplifiée, un équipement 802.11— une *station*— est connecté au réseau lorsqu'il est authentifié et associé auprès d'un *point d'accès*. Cependant ce *point d'accès* possède une portée radio limitée, et le médium radio est sensible aux interférences. Ceci peut mener à une dégradation du signal et / ou une perte de connectivité. Lorsqu'une *station* détecte un tel scénario, elle initie son processus de *handover* afin de changer de *point d'accès* et retrouver une connectivité.

1.1.1 Fonctionnement du handover

Le processus de *handover* peut être divisé en trois phases :

- Déclenchement
- Découverte
- Sélection, authentification et (ré-)association

À l'issue de ces étapes une *station* est censée avoir récupéré une connectivité auprès d'un *point d'accès*.

1.1.1.1 Déclenchement

Cette phase a pour but de décider si un *handover* est nécessaire ou non. Bien que le standard IEEE (Institut des ingénieurs électriciens et électroniciens — *Institute of Electrical and Electronics Engineers*) 802.11 décrive la procédure de *handover*, il s'agit d'une description macroscopique et sa mise en pratique est souvent liée à l'interprétation des fabricants de cartes et microcontrôleurs WI-FI. S'agissant d'un mécanisme bas niveau reposant sur différentes métriques afin de prendre la décision, le déclenchement est souvent implémenté au sein des microcodes des cartes WI-FI et / ou de leurs pilotes. Le plus souvent il s'agit d'évaluer la qualité du lien — en mesurant le RSSI (Mesure de la puissance d'un signal reçu — *Received Signal Strength Indication*) ou en calculant le SNR (Rapport signal sur bruit — *Signal-to-Noise Ratio*) [8, 9] — ou d'évaluer la QoS (Qualité de service — *Quality of Service*) globale [10, 9, 11] en se basant sur les pertes de

paquets ou le nombre de *beacons* non reçus. Lorsque le déclenchement est basé sur l'évaluation de la qualité du lien, il se produit lorsque celle-ci se dégrade jusqu'à passer en dessous d'un seuil acceptable. De la même manière, lorsque la QoS est évaluée et qu'un certain nombre de paquets envoyés sont non acquittés ou que la *station* ne reçoit plus suffisamment de *beacons* durant une certaine période, alors la *station* peut conclure qu'elle n'est plus à portée radio de son *point d'accès* et initier sa procédure de *handover*.

En pratique, le processus de déclenchement n'est pas effectué dès que les précédents seuils sont « franchement » dépassés. En effet, le médium radio est sujet à des perturbations et interférences qui dépendent de l'environnement. Ces perturbations pouvant être locales et / ou temporaires — un objet obstruant temporairement la ligne de vue entre une *station* et son *point d'accès* — il est préférable de ne pas chercher à s'associer à un nouveau *point d'accès*. De ce fait les mesures précédentes sont lissées sur un intervalle de temps afin de s'assurer que la dégradation s'instaure durablement. Si cette politique permet d'éviter des *handovers* « inutiles », elle a pour conséquence d'augmenter le temps de déclenchement de plusieurs secondes lorsque changer de *point d'accès* serait utile [10].

Enfin le choix des seuils pour ces différentes métriques est important car il se peut que la qualité de la connexion soit déjà fortement dégradée avant le déclenchement — puissance reçue faible menant à l'utilisation de modulations robustes ayant de plus faibles débits, ou taux de pertes élevé ne déclenchant pas forcément de handover mais affectant le débit [12, 13, 14].

1.1.1.2 Découverte

Dès qu'une *station* a déclenché sa procédure de *handover*, celle-ci démarre la phase de découverte lors de laquelle elle scanne son environnement à la recherche d'un *point d'accès* auquel se connecter. Le standard 802.11 [15] définit deux méthodes de recherche : 1. la recherche active et 2. la recherche passive.

Recherche active Lors de la recherche active, la *station* envoie des *Probe Requests* sur différents canaux 802.11 afin de détecter les *points d'accès* auxquels se connecter dans son voisinage, comme illustré dans la FIGURE 1.1. La *station* choisit un canal sur lequel effectuer la recherche et envoie un *Probe Request*. Ce message peut cibler un *point d'accès* particulier en recherchant son BSSID (Identifiant d'un BSS — *Basic Service Set Identifier*), un ou plusieurs SSID ("Nom " d'un réseau WIFI — *Service Set Identifier*) ou l'ensemble des *points d'accès* présents en utilisant une « wildcard BSSID », en incluant cette information dans le message de *Probe Request*. Une fois la ou les *Probe Requests* envoyées sur le canal, la *station* attend de recevoir une ou plusieurs *Probe Response* pendant une durée *MinChannelTime*. A l'expiration de cette durée, si aucune *Probe Response* n'a été reçue, la *station* passe au canal suivant et refait une recherche. À contrario si une *Probe Response* a été reçue, alors la *station* doit attendre pendant une durée *MaxChannelTime* sur le canal afin de recevoir d'éventuelles autres *Probe Responses*. A l'expiration de cette durée, la station garde les *points d'accès* éligibles et relance la procédure de recherche sur le canal suivant.

Recherche passive Lors de la recherche passive, une *station* écoute de manière périodique pendant une durée de $100ms$ sur un canal 802.11 donné à la recherche de *beacons* émis par les *points d'accès* de son voisinage. Contrairement à la recherche active, l'ensemble des canaux 802.11 ne sont pas parcourus successivement en une seule fois. Après chaque recherche passive la *station* met à jour sa liste de *points d'accès* éligibles.

La recherche passive est un mécanisme qui s'effectue le plus souvent pendant qu'une *station* est encore associée à un *point d'accès* alors que la recherche active est plus souvent utilisée lors de la rupture de connectivité.

Que la recherche soit passive ou active, il faut remarquer que lorsqu'une *station* effectue une recherche, elle ne peut pas transmettre — la *station* prévient cependant son *point d'accès* qu'elle ne pourra plus communiquer avec lui afin que celui-ci « stocke » les messages qui lui sont destinés afin de les récupérer lors de son retour sur le canal sur lequel le *point d'accès* opère. Dans le cas d'une perte de connectivité, la phase de recherche augmente le temps avant une éventuelle ré-association. Mishra et al. [8] estiment que le processus d'envoi de *Probe Requests* représenterait 90% du temps pris entre le début de la phase de recherche et l'association à un nouveau *point d'accès*.

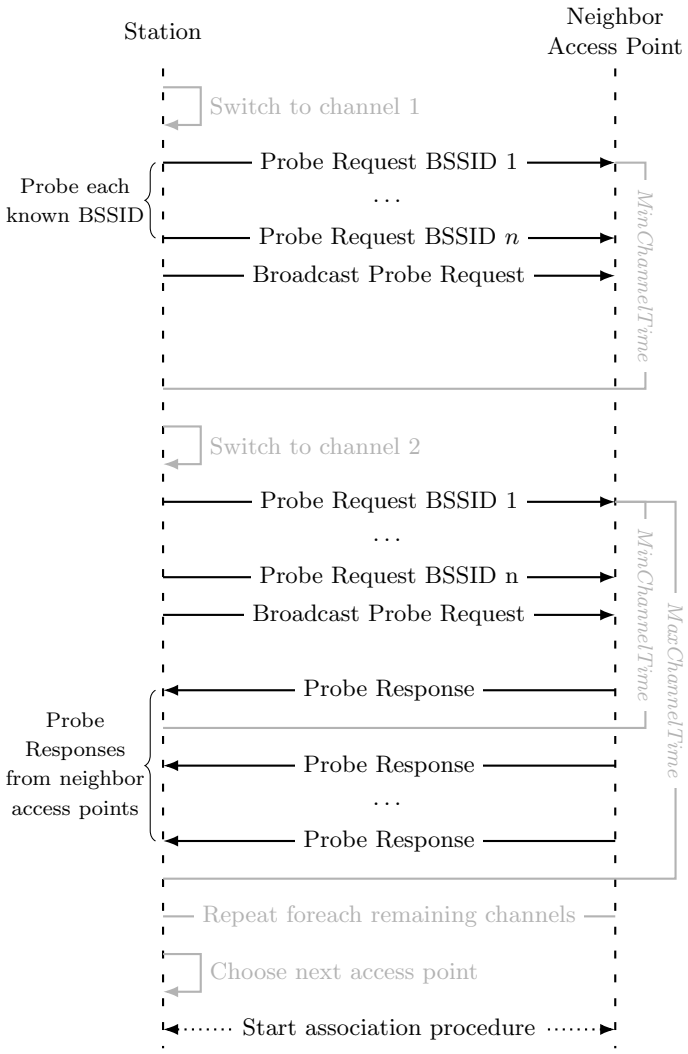


FIGURE 1.1 – Recherche active

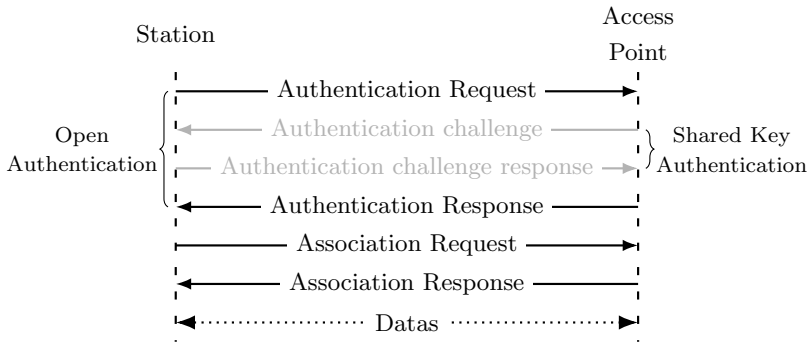


FIGURE 1.2 – Association à un *point d'accès* ouvert (Les messages optionnels en cas de clé partagée sont représentés en gris)

1.1.1.3 Sélection, authentification et (ré-)association

Lorsqu'une *station* a mis à jour la liste de ses *points d'accès* environnants auxquels elle peut potentiellement s'associer, elle entame la phase de sélection lors de laquelle elle va choisir le prochain *point d'accès* auquel elle va finalement s'associer. L'approche la plus simple est de choisir le *point d'accès* ayant le meilleur RSSI [16]. Cependant cette méthode ne prend pas en compte la charge actuelle sur le *point d'accès* ou le taux d'occupation du canal et peut donc résulter en un choix sous optimal en terme de débit [17].

Après avoir choisi le prochain *point d'accès*, la *station* initie le processus d'association avec ce *point d'accès*. Ce processus commence par l'envoi d'une trame *Disassociation Request* au *point d'accès* auquel elle est actuellement connectée. S'ensuit la procédure d'authentification et d'association auprès du nouveau *point d'accès* comme illustré sur la FIGURE 1.2 : 1. envoi d'une trame *Authentication Request*, 2. réception d'une trame *Authentication Response*, 3. envoi d'une trame *Association Request* et 4. réception d'une trame *Association Response*.

Dans le cas où il s'agirait d'une connexion à un *point d'accès* sécurisé en WEP (Protocole de sécurisation des réseaux sans fil visant à fournir une protection équivalente à un réseau filaire (obsolète) — *Wired Equivalent Privacy*), un *challenge* est initié par le *point d'accès* avec l'envoi d'une trame *Authentication Challenge* et une réponse *Authentication Challenge Response* envoyée par la *station* après la demande d'authentification.

La section 1.1.2.4 présente les évolutions concernant l'authentification et l'association du standard.

1.1.2 Améliorations en regard du standard

Nous allons voir dans les parties suivantes les différents travaux effectués afin d'améliorer le *handover* 802.11 en se concentrant sur les parties déclenchement, découverte, sélection et association.

1.1.2.1 Déclenchement

Exploitation de la qualité de service Dans leurs travaux [10], Velayos et Karlsson montrent que la phase de déclenchement est celle qui prend le plus de temps dans le processus de *handover*, à cause des nombreuses retransmissions de trames non acquittées. Ils proposent alors de réduire le nombre de retransmissions de trames non acquittées. Afin d'éviter des déclenchements intempestifs, il faut s'assurer que les retransmissions aient lieu à cause d'une déconnexion ou d'une dégradation « permanente » du signal et non à cause d'interférences temporaires ou de collisions. Ils séléctionnent un nombre de trois retransmissions avant d'initier la procédure de *handover* en se basant sur les travaux de Bianchi [12]. Ce dernier montre que même dans un réseau très dense il est très peu probable qu'un message ait à être retransmis trois fois à cause de collisions.

L'amendement 802.11v [4] *Wireless Network Management* introduit le protocole *BSS transition management for network load balancing*. Bien que celui-ci ne règle pas le problème de *handover* lorsqu'une *station* sort de la portée radio de son *point d'accès*, il propose de régler les problèmes liés à la dégradation de service lorsqu'un *point d'accès* est surchargé — cette dégradation pouvant aboutir à un *handover*. Les *points d'accès* sont alors en mesure d'envoyer une trame *BSS transition management request* à une *station*

compatible pour lui demander de s'associer à un autre *point d'accès*. Cette trame contient une liste ordonnée des *points d'accès* éligibles — *BSS Transition Candidates List Entries* — accompagnée d'une durée avant la *disassociation*. Un élément de la *BSS Transition Candidates List Entries* est composé d'un *Neighbor Report* 802.11K [3] pour chaque *point d'accès* contenant entre autres le BSSID et le canal sur lequel le *point d'accès* opère. La durée avant la *disassociation* indique à la *station* le temps dont elle dispose avant de devoir effectivement changer de *point d'accès*. À la réception d'une *BSS transition management request* une *station* peut choisir d'accepter la demande ou non. Si elle accepte alors la *station* doit se dé-associer de son *point d'accès* actuel et peut s'associer à un des *points d'accès* de sa *BSS Transition Candidates List Entries*. Si la *station* refuse, alors elle doit fournir une raison lors de l'envoi de sa *BSS transition management response*, les raisons pouvant être entre autres : nombre de *beacons* reçus insuffisants, nombre de *Probe Responses* insuffisantes ou *point d'accès* ciblé saturé. En fonction de la réponse le *point d'accès* peut envoyer une autre *BSS transition management request* avec d'autres paramètres afin que la *station* puisse faire un *handover*.

Proactive Scan [13] propose à la fois d'utiliser le débit actuel entre le *point d'accès* et la *station* pour déclencher le *handover*, et d'effectuer une recherche proactive des *points d'accès* environnants avant que le *handover* ne soit déclenché. Les auteurs défendent que le RSSI n'est pas une métrique suffisante pour déterminer si un *handover* doit être effectué, et que des métriques comme le BER (Taux d'erreurs binaire — *Bit Error Rate*), FLR (Taux de pertes de trames — *Frame Loss Ratio*) ou SNR donnent de meilleures informations concernant la qualité du lien mais peuvent être difficiles à obtenir ou calculer. Par ailleurs les auteurs observent que les cartes 802.11 adaptent leurs MCS (Schéma de modulation et de codage — *Modulation & Coding Scheme*) afin d'obtenir le meilleur débit possible dans les conditions du lien actuel. Cette adaptation permet de déduire indirectement la qualité du lien. Ils définissent alors deux seuils basés sur le taux de transmission actuel afin de déclencher la recherche proactive et le *handover*.

Exploitation de la localisation Wisniewski et al. [18] proposent de localiser les *stations* en intérieur en exploitant le delta temporel à la réception. Leurs travaux sont présentés sous la forme d'une extension de l'architecture *flex*WARE [19]. Ils déploient des cartes SMILE (*SMart integrated Localisation Extension*) servant à la localisation, ainsi qu'un *flex*WARE Controller qui est responsable de la gestion du *handover* des *stations*. Les cartes SMILE sont constituées d'un FPGA en charge de démoduler, décoder et d'horodater l'ensemble des trames 802.11 qui sont transmises dans leurs voisinages. Ce type de carte leur permet d'effectuer l'horodatage en moins d'une nanoseconde. À chaque reception d'une trame, les cartes SMILE envoient l'horodatage ainsi que l'identifiant de la *station* émettrice au *flex*WARE Controller. Pour s'assurer que les horodatage soient comparables d'une carte SMILE à une autre, celles-ci sont précisément synchronisées entre elles grâce à PTP (Protocole de synchronisation temporelle précise — *Precision Time Protocol*). Par ailleurs les cartes doivent être déployées en suivant un schéma précis — les auteurs ont choisi une grille dans leurs travaux — afin d'assurer à la fois une couverture suffisante et la possibilité de déterminer la position d'une *station*. Le *flex*WARE Controller va alors déterminer la position de chaque *station* en comparant les horodatages envoyés par les différentes cartes SMILE pour un message donné. Le *flex*WARE Controller connait aussi la position des différents *points d'accès* et est donc capable de connaitre ceux qui sont dans le voisinage de chaque *station*. Dès qu'une *station* s'éloigne de son *point d'accès* et s'approche d'un autre *point d'accès* ou qu'elle passe à côté d'un *point d'accès* sous utilisé, le *flex*WARE Controller ordonne l'envoi d'une trame *Handover Trigger Frame* contenant le canal et l'adresse MAC du *point d'accès* auquel s'associer en direction de cette *station* afin de lui faire déclencher son *handover*. La *station* va lors chercher le *point d'accès* sur le canal et démarrer son processus d'association. Les auteurs observent des temps de *handover* d'une dizaine de millisecondes lors de simulations, et des temps compris entre $10ms$ et $49ms$ sur leur matériel.

Montavont et Noel [20] proposent un *handover* assisté par GPS. Dans leur scénario, les *stations* sont équipées d'un récepteur GPS et envoient leur géolocalisation à un serveur GPS dédié pour un domaine donné. Le serveur GPS connait les coordonnées, le SSID, le canal et le préfixe IPv6 de chaque *point d'accès* de son domaine. Une *station* met à jour ses coordonnées GPS chaque seconde et les compare à ses anciennes coordonnées. Une *station* envoie ses nouvelles coordonnées au serveur GPS uniquement si sa position actuelle est séparée d'au moins un mètre de la position précédente. Le serveur GPS calcule alors la distance entre la *station* et son *point d'accès*. Si cette distance est inférieure à une certaine valeur, alors la *station* est considérée comme étant à portée radio. Au contraire si la distance est supérieure à cette valeur alors la *station* risque de sortir de la portée radio de son *point d'accès* et perdre la connectivité. Le serveur

GPS va alors chercher le *point d'accès* le plus près dans sa base de données. Si le *point d'accès* trouvé est différent de celui auquel la *station* est actuellement associée, alors le serveur GPS lui envoie un message *Handover Initiate* contenant le SSID, le canal et le préfixe IPv6 du nouveau *point d'accès*. La *station* envoie alors une trame *Probe Request* pour s'assurer que le *point d'accès* est fonctionnel et démarre la procédure de *handover*. Les auteurs obtiennent un *handover* moyen de 8.926ms lors de leurs expérimentations.

Tseng et al. [14] introduisent un serveur de localisation qui maintient une liste des *points d'accès* avec leurs paramètres d'association, leurs coordonnées GPS et une liste de *points d'accès* environnants. Les *stations* sont supposées capables de récupérer leurs coordonnées GPS mais, afin d'en limiter l'impact énergétique, celles-ci sont récupérées uniquement lorsque le RSSI entre une *station* et son *point d'accès* descend sous un certain seuil. Après avoir récupéré ces coordonnées GPS une *station* demande au serveur de localisation de lui envoyer une liste de *points d'accès* environnants. La *station* va alors déterminer la direction dans laquelle elle se déplace en utilisant l'historique de ses positions, et va choisir un *point d'accès* en adéquation afin de s'y associer.

1.1.2.2 Découverte

Les approches à base de graphes Dans leurs travaux, Shin et al. [21] proposent de réduire le temps nécessaire pour sonder le voisinage en utilisant deux graphes : un graphe de voisinage *Neighbor Graph* et un graphe de qualité de liens *Non Overlapping Graph*. Ces graphes sont locaux — c'est à dire qu'un *point d'accès* ne connait que les sous-graphes correspondant à son voisinage — et sont partagés aux *stations* lors de leurs association. Le *Neighbor Graph* correspond à l'ensemble des *points d'accès* et de leurs canaux dans le voisinage. Ainsi lorsqu'une *station* a besoin d'effectuer un handover, elle utilise ce graphe afin de trouver les canaux sur lesquels envoyer des *Probe Requests*. Le *Non Overlapping Graph* rajoute une information sur la qualité du lien en représentant l'ensemble des *points d'accès* qui ne possèdent pas de zones dans lesquelles une *station* peut communiquer simultanément avec eux. Ce graphe permet de réduire l'attente *MaxChannelTime* car une *station* n'est plus obligée d'attendre si l'ensemble des *points d'accès* à portée ont envoyé une *Probe Response*. L'utilisation du *Neighbor Graph* leur permet de réduire la phase de recherche de 80.7% et si le *Non Overlapping Graph* est aussi utilisé, la phase recherche est réduite de 83.9%.

Park et al. [22] proposent une approche similaire dans laquelle une *station* interroge un *Neighbor Graph Server* afin d'obtenir les *points d'accès* environnants. Le serveur répond en envoyant une liste de *points d'accès* et leurs canaux, permettant à la *station* de chercher un *point d'accès* uniquement sur ces canaux. Le *Neighbor Graph* peut être mis en place manuellement ou automatiquement en ajoutant l'ancien *point d'accès* dans le message d'association à un nouveau *point d'accès* lors du *handover*. Le nouveau *point d'accès* peut alors informer le *Neighbor Graph Server* qu'un lien existe entre l'ancien *point d'accès* et lui même.

DeuceScan [23] permet à une *station* de maintenir un graphe spatio-temporel des points d'accès environnants. Lorsqu'une *station* entre dans le réseau elle sonde l'intégralité de son environnement et conserve les trois canaux sur lesquels elle reçoit des messages en provenance de *points d'accès* avec le plus haut niveau de signal. Ces trois *points d'accès* forment un triangle correspondant à la zone où la *station* se trouve actuellement. Lors des prochaines recherches passives de son environnement, la *station* ne fera plus une recherche complète, mais la limitera aux trois canaux précédents auxquels un à trois canaux aléatoires seront ajoutés afin de sonder le reste de son environnement. Ces canaux supplémentaires servent à déterminer si la *station* est en train de passer d'un triangle de *points d'accès* à un autre. Si la puissance observée sur un de ces canaux est supérieure à l'un des trois formant le triangle actuel, alors le *point d'accès* du triangle ayant la puissance la plus faible est remplacé par le nouveau, et un nouveau triangle est considéré. Si le *point d'accès* auquel la *station* est connectée n'est plus dans ce triangle, alors un *handover* est effectué avec le *point d'accès* ayant le plus haut niveau de RSSI. Le *handover* peut aussi être déclenché au sein d'une même zone si le niveau de RSSI descend en dessous d'un certain seuil.

Réduction du temps passé sur les canaux *Proactive Scan* (SECTION 1.1.2.1) propose d'accélérer la découverte des *points d'accès* environnants après avoir déclenché le handover lorsque le débit estimé passe en dessous d'un certain seuil. À ce moment, la *station* va sonder différents canaux de manière périodique afin de trouver des *points d'accès* potentiels. Cette périodicité est liée au taux de transmission ainsi qu'au RSSI actuel et vaut : 100ms, 200ms ou 300ms. Afin d'avoir des temps de recherche courts, ils fixent les valeurs de *MinChannelTime* et *MaxChannelTime* à 5ms. Les cartes utilisées pour leur test mettant 2.9ms à

changer de canal, une recherche sur un canal différent prend $10.8ms$. De plus ils proposent de réduire le nombre de canaux à sonder en se limitant aux canaux utilisés par des *points d'accès* déjà rencontrés partageant le même SSID. Pour limiter la perte de message en direction de la *station* pendant les phases de recherche, celle-ci envoie une *Sleep Request* à son *point d'accès* avant d'initier une recherche proactive. Les phases de recherche sont des recherches actives car elles permettent de s'assurer que les liens avec les *points d'accès* sondés ont un certain degré de « symétrie ». Dès lors que le taux de transmission dépasse un certain seuil, un *handover* est initié, et la *station* va s'associer avec le *point d'accès* trouvé lors des précédentes recherches ayant le RSSI le plus élevé. Néanmoins si aucun *point d'accès* ne possède un RSSI supérieur à celui mesuré sur le *point d'accès* avec lequel la *station* est actuellement associée, alors le handover est abandonné. Sur leur matériel, ils observent que la méthode classique de *handover* engendre une interruption de service de $3s$ à $8s$ ainsi qu'une dégradation de la qualité du lien en amont du *handover* qui augmente le délai entre chaque paquet. Lorsque leur solution est appliquée, aucune interruption de service n'est observée, mais un délai de réception de $47ms$ est néanmoins présent lorsque la *station* change de canal et s'associe à un nouveau *point d'accès*.

SyncScan [24] introduit une corrélation entre le canal sur lequel opère un *point d'accès* et sa fréquence d'émission de *beacons* afin de réduire la durée pendant laquelle une *station* sonde un canal de manière passive. Par exemple un *point d'accès* sur le canal 1 va annoncer un *beacon* toutes les t millisecondes, un *point d'accès* sur le canal 2 va annoncer toutes les $t+d$ millisecondes, un *point d'accès* sur le canal 3 va annoncer toutes les $t+2d$ millisecondes et ainsi de suite. Les *stations* sont alors capables de synchroniser leurs recherches passives sur le schéma d'émission de *beacon* actuel. Cette méthode permet de réduire la durée de recherche sur un canal spécifique, ce qui permet d'augmenter la fréquence de recherche sans augmenter le temps pendant lequel les transmissions sont mises en pause, donnant à la *station* une meilleure connaissance générale de son environnement. Le *handover* est déclenché lorsque le RSSI ou le SNR indiquent une dégradation de la qualité du lien. Malgré le fait que leur solution nécessite une synchronisation précise des *points d'accès* (assurée par l'utilisation de NTP (Protocole de temps sur le réseau — *Network Time Protocol*)), celle-ci reste compatible avec le standard 802.11 car elle n'introduit pas de messages de contrôle supplémentaires et utilise la méthode de recherche passive classique.

Jin et al. [25] proposent d'accélérer le processus de recherche en évitant de rester trop longtemps sur les canaux inoccupés par un *point d'accès*. Le fonctionnement du standard demandant aux *stations* de rester sur le canal pendant un temps *MinChannelTime*, cela retarde la recherche sur un autre canal alors que celui-ci est peut être inoccupé. Ils proposent d'utiliser le mécanisme de RTS (Demande d'envoi — *Request to Send*) / CTS (Disponible pour l'envoi — *Clear to Send*), afin de sonder rapidement les canaux. Ce mécanisme sert à s'assurer que le canal sera libre lors de l'envoi d'un message par une *station* vers son *point d'accès*. Lorsqu'une *station* envoie un message RTS à son point d'accès, celui-ci lui répond avec un message CTS lui indiquant qu'elle a le droit de communiquer avec lui tout en assurant que les autres équipements recevant ce message n'ont pas le droit de transmettre. Ceci permet à cette *station* d'envoyer son message sans que celui-ci n'ait de collisions au niveau du *point d'accès*. Ces messages ne sont pas des messages pouvant utiliser l'adresse de destination *broadcast*, et ne peuvent donc pas être envoyés à tous les *points d'accès* présents sur le canal en même temps. Pour effectuer une recherche de l'ensemble des *points d'accès* présents sur un canal à l'aide de ces messages, les auteurs proposent d'utiliser une adresse MAC virtuelle partagée par l'ensemble des *points d'accès*. Une *station* enverra alors un message RTS à destination de cette adresse et l'ensemble des *points d'accès* répondront simultanément un CTS qui contiendra par contre leur adresse MAC propre. Si plusieurs *points d'accès* sont présents sur le même canal, les CTS sont en collision, mais la station peut tout de même détecter une occupation du canal pendant la période de réception du CTS, et ainsi considérer la présence de *points d'accès* sur ce canal. L'échange RTS / CTS étant fait rapidement, cette technique permet de déterminer les canaux opérationnels des *points d'accès* environnant dans un temps restreint. La *station* pourra alors envoyer un message de *Probe Request* sur ces canaux uniquement, en utilisant l'adresse MAC partagée.

Réduction du nombres de canaux *AdaptiveScan* [26] cherche à réduire les durées de recherche en groupant les canaux par valeurs de RSSI reçues. Lorsqu'une *station* effectue une recherche proactive pour la première fois, tous les canaux vont être sondés, et la *station* va classer ces différents canaux en trois groupes en fonction des valeurs maximales de RSSI reçues. Le premier groupe contient les trois canaux possédant les valeurs reçues les plus élevées et est appelé *Candidates*. Les trois canaux suivants sont

placés dans le groupe appelé *Reserve* et les canaux restants sont placés dans le groupe *Search*. L'idée est d'adapter sa politique de recherche en fonction de ces groupes. Une *station* sondera plus fréquemment les canaux du groupe *Candidates* (période de l'ordre de 5s) car les dernières valeurs reçues étant plus hautes cela laisse entendre que les *points d'accès* dans ces canaux sont plus proches et sont donc plus enclins à être de bon candidats pour un *handover*. Les canaux du groupe *Reserve* sont sondés un peu moins régulièrement (période de l'ordre de 10s) et les canaux du groupe *Search* sont sondés peu fréquemment à des fins exploratoires (période de l'ordre de 50s). Si, lors d'une recherche sur l'un des canaux des groupes à plus faibles fréquences, la valeur observée est supérieure à un des canaux des groupes à plus fortes fréquences, alors les groupes sont réorganisés en conséquence. Cette méthode permet d'éviter de faire une recherche sur l'ensemble des canaux à chaque fois, réduisant le temps total nécessaire à l'acquisition d'informations, surtout lorsque le nombre de canaux utilisés dans le voisinage est important.

Selective Scanning [27] est une solution proposant d'appliquer un masque sur les canaux à sonder. Lorsqu'une *station* rejoint un réseau pour la première fois, elle effectue une recherche active complète de son environnement. Pour chaque canal sur lequel la *station* a reçu un message *Probe Response*, elle active l'entrée correspondante dans son masque. La *station* se connecte ensuite au *point d'accès* ayant le meilleur signal, et enlève le canal correspondant de son masque, car la probabilité d'avoir un *point d'accès* utilisant le même canal dans un voisinage proche est faible dans le cadre d'un déploiement d'infrastructure. Lorsqu'un *handover* a besoin d'être effectué, la *station* va sonder uniquement les canaux correspondant à son masque. Si aucun *point d'accès* n'est trouvé, alors la *station* va sonder les canaux correspondant à l'inverse de son masque actuel. Si toujours aucun *point d'accès* n'est trouvé, alors la *station* va effectuer une recherche active complète. Si des *points d'accès* sont trouvés lors de ces deux dernières phases, alors la *station* met à jour son masque pour l'appliquer aux recherches suivantes. Par ailleurs *Selective Scan* utilise une politique de cache afin de stocker pour chaque *point d'accès* auquel une *station* a été associée, une liste de deux *points d'accès* voisins. Lors d'un *handover*, la *station* va d'abord essayer de s'associer aux *points d'accès* de cette liste avant d'effectuer un *Selective Scan* si jamais les associations ont échoué.

Velayos and Karlsson [10] remarquent que la recherche active prend plus de temps dans les zones avec une forte densité de *points d'accès* car il est plus probable qu'une *station* ait à attendre *MaxChannelTime* sur chaque canal. Ils pointent aussi le fait qu'il y a beaucoup de canaux à sonder — onze aux États-Unis, treize dans certains pays d'Europe — alors que seul le sous ensemble correspondant aux canaux orthogonaux est utilisé. Ils proposent donc de se limiter à ce sous ensemble pour les recherches actives.

Comme présenté précédemment, 802.11K introduit les messages *Neighbor Report* qui contiennent des informations sur le voisinage radio. En plus d'être utilisé dans les trames *BSS transition management request*, un *Neighbor Report* peut être sollicité par une *station* associée avec un *point d'accès* 802.11K. Ainsi la liste de *points d'accès* environnants avec leurs BSSID et leurs canaux peut être obtenue par une *station* en envoyant un *Neighbor Report Request* à son *point d'accès*. Par défaut les *points d'accès* retournés sont membres de l'ESS (Ensemble de BSS appartenant à un même "lien logique " — *Extended Service Set*) auquel la *station* est actuellement associée, mais celle-ci peut cibler un SSID spécifique ou aucun en utilisant une « wildcard SSID ». Une *station* 802.11K est alors capable d'envoyer une *Probe Request* vers un *point d'accès* particulier sans avoir à parcourir l'ensemble des canaux, réduisant ainsi le temps nécessaire pour la recherche avant d'effectuer son *handover*.

Utilisation d'un moyen d'écoute parallèle L'utilisation de plusieurs radios [28, 29] au niveau d'une *station* peut permettre d'améliorer les performances du *handover*. Une *station* s'associe à un *point d'accès* en utilisant une de ses deux radios et utilise l'autre pour chercher d'autres *points d'accès* dans son environnement. Dès que la seconde radio trouve un *point d'accès* sur lequel s'associer, le processus d'*handover* est déclenché sur la première radio avec les informations concernant le nouveau *point d'accès*. Cependant une interruption est tout de même présente lorsque la première radio doit changer de *point d'accès*. Une approche minimisant la durée d'interruption est également proposée : lorsque la seconde radio trouve un *point d'accès* sur lequel effectuer le *handover*, alors la seconde interface s'y associe, et les deux radios échangent leurs rôles, ce qui permet d'avoir un *handover* quasi instantané — les tables de routage et ARP (Protocole de résolution d'adresse — *Address Resolution Protocol*) devant être mises à jour.

Jin et al. [30] proposent d'utiliser un canal de rendez-vous dédié à l'échange des messages de *Probes* dans un contexte où les *points d'accès* possèdent plusieurs interfaces radio. L'ensemble de ces *points d'accès* dédieraient une de leurs interfaces à l'écoute sur un même canal afin de recevoir des *Probe Request*. Une

station ayant besoin d'effectuer un handover n'aurait alors à sonder que sur un seul canal afin de trouver un ou plusieurs *points d'accès*. Un *point d'accès* enverrait le canal sur lequel il opère réellement dans la *Probe Response*.

Waharte et al. [31] proposent d'utiliser des réseaux de capteurs sans fil afin d'observer et d'analyser la couverture offerte par les *points d'accès*. Dans leur architecture ils déploient deux réseaux : un réseau de contrôle sur la bande des $2.4Ghz$ et un réseau de données sur la bande des $5Ghz$. Le réseau de contrôle est composé de trois types de nœuds : manageurs, relais et agents. Les manageurs sont rattachés à un *point d'accès* particulier et jouent le rôle d'interface entre le réseau de contrôle et celui de données. Les agents sont rattachés à une *station* et jouent aussi le rôle d'interface entre les deux réseaux. Les relais sont déployés dans la zone de couverture de chaque *point d'accès* et sont utilisés afin de transmettre des informations entre les agents et les manageurs. Les auteurs considèrent que les *points d'accès* 802.11 ont une portée de $150m$ et que les capteurs sans fil ont une portée de $50m$. Lors de l'initialisation du réseau de contrôle, les manageurs envoient des paquets d'initialisation. Ces paquets sont relayés jusqu'à deux sauts et permettent aux nœuds relais de savoir à quels *points d'accès* ils sont rattachés. Lorsqu'une *station* observe une dégradation de sa connectivité avec son *point d'accès* actuel, elle va utiliser son nœud agent afin de diffuser une trame *AP_List Request*. Les nœuds relais environnants qui reçoivent cette trame répondent avec une trame *AP_List Response* contenant les *points d'accès* auxquels ils sont rattachés. Le *handover* est alors effectué avec une recherche active limitée aux canaux sur lesquels se trouvent les *points d'accès* précédemment récupérés.

MultiNet [32] s'affranchi de l'utilisation de plusieurs interfaces radio sur un même équipement mobile en ajoutant une couche de virtualisation au dessus de l'interface physique. Ceci permet de créer des interfaces radio virtuelles capable de se connecter sur différents *points d'accès*. Cette approche permet d'obtenir un gain significatif sur la consommation énergétique par rapport à l'utilisation de plusieurs interfaces radio physique mais sacrifie alors le débit ainsi que la latence de bout en bout car l'unique interface physique utilisée doit régulièrement changer de réseau afin de se connecter aux différents réseaux de ses interfaces virtuelles. Ce changement s'effectue selon un *round-robin* équitable ou selon la charge actuelle des différentes interfaces virtuelles. De plus, *MultiNet* nécessite des modifications assez lourdes au sein des équipements afin de fonctionner.

1.1.2.3 Sélection

La méthode de sélection la plus simple se base sur le RSSI des *points d'accès* environnants afin de s'associer à celui qui a le RSSI le plus élevé. Malheureusement le RSSI ne donne pas d'information sur la charge actuelle du *point d'accès* ou du canal. Un *point d'accès* avec le signal le plus fort n'offre pas forcément la meilleure QoS, ce qui peut amener à un *handover* non souhaité.

Pour palier à ce problème, il est possible d'estimer la bande passante disponible entre une *station* et un potentiel *point d'accès*, en utilisant le délai entre l'émission de *beacons* [16]. Une fois que l'on connait l'intervalle d'émission de *beacons* théorique pour un *point d'accès* donné — en général l'intervalle entre les *beacons* vaut $100ms$ — il est possible d'estimer le temps passé en file d'attente en calculant la différence — δ_{file} — entre l'horodatage théorique de réception du prochain *beacon* et l'horodatage de la réception dudit *beacon*. Ce calcul marche sous l'hypothèse que les *beacons* n'utilisent pas une file prioritaire. Il s'agit ensuite de choisir le *point d'accès* offrant le meilleur débit.

Dang et al. [33] utilisent de la logique floue afin d'effectuer la sélection du *point d'accès*. Ils utilisent trois métriques : la charge sur la cellule, la distance entre l'utilisateur et le *point d'accès* et la bande passante disponible. Ces paramètres sont passés dans un moteur de logique floue, c'est à dire qu'ils sont divisés en sous-groupes avec des limites « souples » — pas de seuils stricts pour passer d'un sous-groupe à un autre. Par exemple la charge est divisée en quatre sous-groupes : charge basse, charge normale, charge moyenne, charge élevée. Dans leurs simulations, leur politique de *handover* avait de meilleurs résultats en terme de débit global sur le réseau car il répartit les utilisateurs vers les *points d'accès* ayant une charge modérée et de la bande passante disponible. Cependant des *handovers* vers des *points d'accès* éloignés sont générés sans considération énergétique ou de dégradation du lien.

La logique floue est aussi employée pour effectuer un *handover* dans un réseau hétérogène [34] — 802.11/ téléphonie mobile — prenant en compte le cadre applicatif de l'utilisateur ainsi que les contraintes qu'il définit. Ainsi les auteurs dressent un ensemble de métriques à prendre en compte : le prix (dans le cas de *roaming*), la bande passante, la qualité du lien, le temps sur une cellule — durée entre deux *hando-*

vers—, la durée de *handover* et la consommation énergétique. Dans leurs tests ils considèrent deux applications : VoIP (Voix sur IP — *Voice over IP*) et le téléchargement d'un gros fichier. La VoIP possède de fortes contraintes sur la durée de *handover* et la durée entre deux *handovers* alors que le téléchargement va favoriser un débit élevé et un prix restreint. Ils proposent plusieurs algorithmes de résolution de logique floue afin de choisir le « meilleur » *point d'accès* en fonction des contraintes imposées par l'utilisateur.

Théorie des jeux Shakkottai et al. [35] proposent de se baser sur le « jeu de population » afin de maximiser le débit dans un environnement *multihoming*. L'idée pour une *station* est de diviser son trafic sur plusieurs *points d'accès* de son entourage au lieu de tout envoyer à un unique *point d'accès*. Une *station* doit payer pour l'occupation du canal, le gain étant le rapport entre ce qu'elle paie et le débit obtenu. Les *stations* sont catégorisées par type d'utilisation, et les *points d'accès* font payer des prix différents par type d'utilisation. Il s'agit alors pour une *station* de minimiser le temps d'utilisation et le prix payé.

Une extension est proposée [36] dans laquelle les *stations* doivent payer un prix supplémentaire pour chaque *point d'accès* supplémentaire utilisé pour prendre en compte la complexité induite par la gestion de différentes connexions simultanées.

Des travaux [37] se basent sur l'hypothèse que dans les réseaux 802.11 denses à l'échelle de la ville, beaucoup de *points d'accès* partagent le même fournisseur d'accès à Internet. Toujours en considérant les réseaux denses, il est très probable que des *points d'accès* voisins occupent le même canal, ce qui entraine une saturation de celui-ci et / ou peut provoquer des interférences. Les auteurs proposent alors que chaque *points d'accès* appartenant à un même fournisseur d'accès à Internet soit obligé de partager une partie de sa bande passante aux utilisateurs ayant le même fournisseur, et qu'il y ait une autorité centrale pour chaque fournisseur d'accès à Internet en charge de gérer les associations aux *points d'accès* de chaque client. Pour chaque client, en fonction de ses besoins, le contrôleur va décider si celui-ci doit s'associer à son *point d'accès*, à un autre dans son entourage, ou à plusieurs *points d'accès* à la fois. Par ailleurs, en fonction de la charge actuelle, ce contrôleur est chargé d'allumer ou d'éteindre certains *points d'accès* afin de limiter les interférences.

1.1.2.4 Authentification et Association

Une version plus robuste de *Proactive Scan* (SECTION 1.1.2.1) a été proposée par Saxena et Roy [38]. Ils proposent de réduire le temps d'association une fois qu'une *station* a récupéré un ensemble de *points d'accès* grâce à une recherche proactive en introduisant un mécanisme de pré-authentification. La *station* va envoyer un message *Proactive Handover Auth Request* à son *point d'accès* contenant le *point d'accès* auquel la *station* aimerait se connecter, ainsi que les clés d'authentification nécessaires. Le *point d'accès* est alors en charge d'envoyer une *Ressource Request* au *point d'accès* concerné qui doit lui répondre en envoyant une *Ressource Response* lui indiquant s'il accepte la connexion de la *station* ou non. S'il accepte, le *point d'accès* alloue des ressources afin de pré-authentifier la *station*. Le *point d'accès* auquel la *station* est actuellement connectée doit alors envoyer une *Proactive Handover Auth Response* contenant la réponse du *point d'accès*. À ce moment la *station* peut envoyer un message *Proactive Handover Auth Confirm* à son *point d'accès* qui enverra un message *Ressource Confirm* au *point d'accès* concerné, afin de confirmer le *handover*. Une fois ce processus effectué, la *station* procède directement à son association auprès du nouveau *point d'accès* en envoyant une *Association Request*. L'ensemble de ces modifications assurent que lorsqu'une *station* entame son *handover*, le *point d'accès* auquel elle compte s'associer est en mesure de l'accepter afin d'éviter une interruption de service.

Pack and Choi [39] proposent de réduire le temps nécessaire à l'authentification d'une *station* auprès de son nouveau *point d'accès*. Ils proposent une approche basée sur les coordonnées géographiques des *points d'accès* et des *stations*. Lors de la première association d'une *station* dans un réseau d'infrastructure, le *point d'accès* va vérifier si cette *station* est autorisée à joindre le réseau en interrogeant un server d'authentification RADIUS (Service d'authentification d'utilisateurs à distance — *Remote Authentication Dial-In User Service*) ou *Diameter*. Si le client est autorisé, alors le serveur ne va pas répondre uniquement au *point d'accès* effectuant la vérification, mais va diffuser cette autorisation auprès de l'ensemble des *points d'accès* dans son voisinage — qu'ils nomment *Frequent Handoff Region*. Ceci permet à la *station* d'être pré-authentifiée auprès des *points d'accès* de son voisinage, et permet par la suite de réduire la durée prise par la phase d'authentification lors d'un *handover*, car les *points d'accès* n'auront pas à interroger le serveur

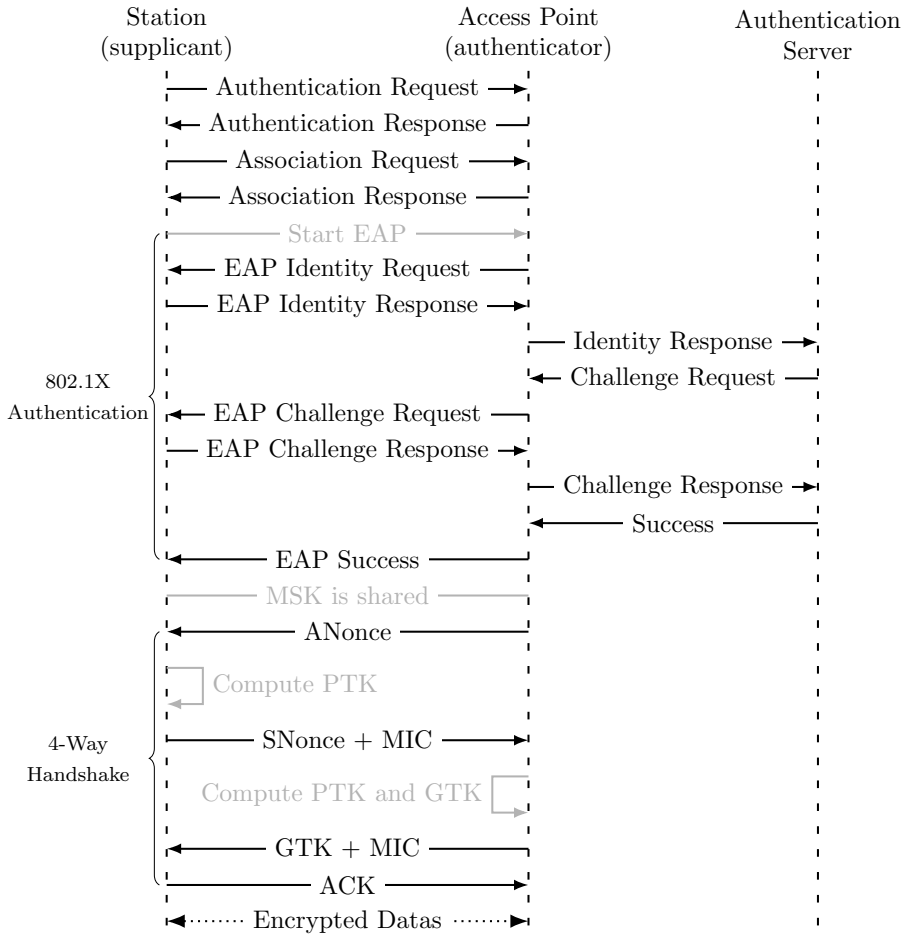


FIGURE 1.3 – Séquence d'authentification en 802.11I

d'authentification.

802.11I [40] apporte les mécanismes d'authentification de 802.1X au standard 802.11. Une *station* s'authentifie et s'associe à un *point d'accès* comme s'il s'agissait d'un *point d'accès* non sécurisé, et va ensuite démarrer une procédure d'authentification 802.1X comme illustré sur la FIGURE 1.3 — la *station* n'est pas en mesure de communiquer avec son *point d'accès* tant que cette procédure d'authentification n'est pas terminée. L'authentification repose sur EAP (Protocole extensible d'authentification réseau — *Extensible Authentication Protocol*), une *station* ayant alors le rôle d'EAP *Supplicant* peut initier une authentification EAP ou attendre que le *point d'accès* ayant le rôle d'EAP *Authenticator* envoie un message EAP *Identity Request*. Un processus de *Challenge Request/Challenge Response* démarre entre la *station* et le serveur d'authentification afin de partager une MSK (Clé maîtresse de session — *Master Session Key*). Après cet échange la *station* et le *point d'accès* initient un *4-Way Handshake*. Les deux entités dérivent une PMK (Clé maîtresse partagée — *Pairwise Master Key*) à partir de la MSK et calculent deux nombres aléatoires : ANONCE (Nombre aléatoire utilisé une seule fois par l'entité en charge de l'authentification — *Authenticator random Number used Once*) pour le *point d'accès* et SNONCE (Nombre aléatoire utilisé une seule fois par l'entité en demandant à être authentifiée — *Supplicant random Number used Once*) pour la *station. Note : Dans le cas où une* station *et un* point d'accès *possèdent une clé partagée et effectuent une association protégée par* WPA-PSK *(Accès protégé utilisant* WPA *avec une clé partagée* — WPA Pre-Shared Key*) alors cette clé partagée sera utilisée pour dériver la* PMK *et l'échange* EAP *n'est pas nécessaire.* L'*Authenticator* envoie alors le ANONCE précédemment calculé au *Supplicant* qui va alors l'utiliser pour dériver une clé PTK (Clé partagée éphémère — *Pairwise Transient Key*) en utilisant ANONCE, SNONCE, la PMK et les adresses MAC de l'*Authenticator* et du *Supplicant*. Le *Supplicant* transmet alors son SNONCE et un MIC (Code de vérification de l'intégrité d'un message — *Message Integrity Code*) à l'*Authenticator* qui va alors pouvoir dériver sa PTK et s'assurer de son intégrité en déchiffrant le MIC. L'*Authenticator* calcule alors une GTK (Clé de groupe partagée éphémère — *Group Transient Key*) et l'envoie au *Supplicant* avec un MIC. Si celui ci arrive

à déchiffrer la GTK, alors il l'installe et acquitte l'*Authenticator*, les transmissions qui suivront seront alors chiffrées en utilisant la PTK ou la GTK.

L'ensemble du processus d'authentification 802.1X prend du temps et est amplifié si le serveur d'authentification ne s'exécute pas sur le *point d'accès*, voire n'est pas sur le même réseau local. 802.11R [41] tente de réduire le temps nécessaire à l'authentification d'une *station* lorsque celle-ci se déplace au sein d'un même ESS. L'idée est d'effectuer le *4-Way Handshake* seulement lorsque la *station* rejoint un ESS. Lors de la première authentification EAP, la *station* et le serveur d'authentification créent et partagent une MSK. Ils dérivent alors deux PMK particulières : PMK-R0 depuis la MSK et PMK-R1 depuis la PMK-R0 pour chaque *point d'accès* dans l'ESS. Le serveur d'authentification envoie alors une PMK-R1 à chaque *point d'accès*. Lorsqu'une *station* a besoin de passer d'un *point d'accès* à un autre, elle va utiliser le *Fast Transition Protocol* qui consiste en l'ajout des ANONCE, SNONCE ainsi que les références aux clés PMK-R0 et PMK-R1 dans les messages d'authentification et d'association. Le *4-Way Handshake* est alors effectué lors du processus d'association.

1.1.2.5 Discussion

Les approches basées sur les graphes présentent des limitations. La complexité de construction des graphes par apprentissage ne permet pas de garantir leurs exhaustivités, alors qu'un établissement de ceux-ci a priori ne permet pas de palier aux fluctuations de la topologie. Le partage de ces graphes ajoute par ailleurs un surcoût protocolaire — qui peut cependant être négligeable.

De plus la réduction de l'espace de recherche ne respecte pas forcément le standard — qui suppose que l'ensemble des canaux doivent être sondés lors d'une recherche active. Certes, il est possible de s'affranchir de cette contrainte, et les propositions cherchant à se limiter aux canaux orthogonaux peuvent grandement accélérer le processus, mais celles-ci risquent aussi de ne pas prendre en compte des *points d'accès* utilisant des canaux non-orthogonaux qui pourraient aboutir à une meilleure QoS, surtout dans des environnements avec une forte densité de *points d'accès*.

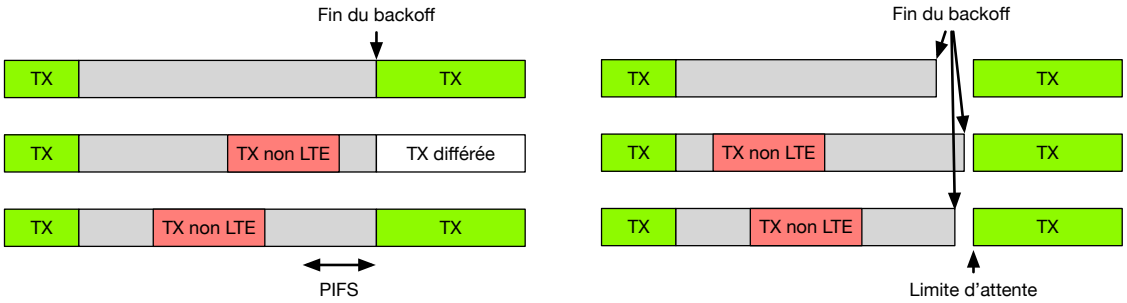
Les approches comme *SyncScan* permettent d'éviter le caractère aléatoire des phases de recherche passive lors desquelles les *stations* doivent sonder pendant une durée relativement longue afin de s'assurer de recevoir des *beacons*. Mais cette solution se base sur une forte synchronisation temporelle des *points d'accès*. Certes, cela permet d'obtenir une meilleure connaissance de l'environnement radio des *stations* à moindre frais car celles-ci savent quand une recherche passive est pertinente, mais assurer une telle coordination entre les *points d'accès* n'est pas évident.

L'utilisation de plusieurs radios permet de grandement réduire les durées de *handover* car la recherche et la nouvelle association peuvent être effectuées par la radio qui n'est actuellement pas associée. Cependant les équipements mobiles ne sont pas forcément équipés de plusieurs radios — pour des raisons de prix et de consommation énergétique — et si leur radio est capable de communiquer sur plusieurs canaux simultanément, c'est en général utilisé pour améliorer le débit. *MultiNet* ajoute un niveau de virtualisation au dessus de l'interface radio physique afin d'utiliser des interfaces radios virtuelles. Cette approche est bien plus économe en énergie que l'utilisation de plusieurs radios physiques, mais sacrifie le débit maximum et rajoute de la *latence* car la radio physique partage son temps entre les différentes interfaces virtuelles.

Les approches se basant sur un déploiement de topologie précise et / ou ajoutant du matériel spécifique peuvent être onéreuses et semblent difficiles à déployer dans des environnements non contrôlés.

Les approches utilisant la position GPS permettent d'effectuer des *handovers* rapides et/ou proactifs mais demandent l'acquisition fréquente des coordonnées GPS augmentant la consommation énergétique — ce qui peut être problématique pour des équipements mobiles, non reliés au réseau électrique.

Enfin les amendements 802.11K, 802.11R et 802.11V ont été spécifiés afin d'améliorer la connaissance de l'environnement radio d'une *station* ainsi que pour faciliter son processus de *handover*. Ces amendements ayant été officiellement ajoutés dans le standard en 2008 — 802.11K et 802.11R— et en 2011 — 802.11V— nous pourrions supposer qu'ils soient activement déployés dans les équipements grand public actuels. Malheureusement Sanchez et Boukerche [42] observent que ce n'est pas forcement le cas. Ils constatent que même si 802.11K est implémenté dans les appareils *Apple* ainsi que dans les équipements d'infrastructure, ou que 802.11R est déployé dans certains réseaux tels qu'*eduroam*, ils ne sont pratiquement pas utilisés ou proposés dans les produits destinés à un usage domestique. Les réseaux domestiques ayant des surfaces réduites et n'ayant pas d'infrastructure d'authentification et d'autorisation, l'implémen-



(a) Méthode utilisant un CCA rapide

(b) Méthode utilisant un *backoff* complet sur chaque canal

FIGURE 1.4 – Méthodes d'accès au canal lors de l'utilisation de plusieurs canaux en LTE-LAA

tation de 802.11R n'est pas forcément nécessaire. Ces trois amendements sont par ailleurs optionnels et laissent beaucoup de paramètres à la discrétion du fabricant. Finalement, le principal frein à leur déploiement vient probablement d'autres amendements ayant un impact direct sur l'utilisateur — et indirectement sur les volumes des ventes — tels que 802.11N [43] ou 802.11AC [44] qui améliorent les débits. De ce fait ces équipements utilisent le plus souvent une procédure de *handover* sans les améliorations proposées dans les amendements 802.11K, 802.11R et 802.11V.

1.1.3 Propositions au-delà du standard 802.11

1.1.3.1 LTE sur les bandes non licenciées

LTE sur les bandes non licenciées a pour objectif de porter la gestion du medium sans fil sur les bandes non licenciées, et plus particulièrement dans les bandes de fréquences $5GHz$ [45, 46, 47, 48]. L'objectif est de déployer de petites cellules sur les bandes de fréquences non licenciées sans avoir à fortement modifier le fonctionnement de LTE — gestion des ressources des ENB (Station de base utilisée en LTE — *Evolved Node B*) et adaptation du lien notamment. Il est proposé d'utiliser ces cellules comme seconde porteuse et d'utiliser l'agrégation de porteuse entre une porteuse sur bande licenciée et une porteuse sur bande non licenciée. Ceci permet d'obtenir un gain en débit lorsqu'un utilisateur est à portée d'une cellule non licenciée tout en gardant les avantages offerts par la LTE — concernant la gestion de la mobilité et les mécanismes d'authentification par exemple. Un équipement mobile est toujours connecté à une cellule opérant sur une bande licenciée et l'accès à une cellule non licenciée est géré de manière centralisée.

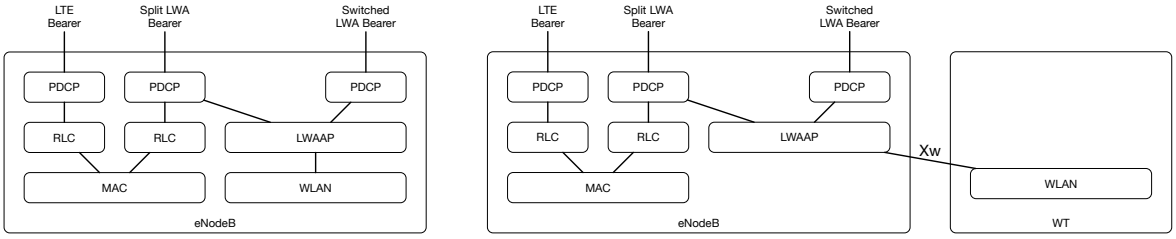
Comme cette proposition utilise la bande partagée des $5GHz$, elle doit offrir un moyen d'assurer que l'accès au médium radio est équitable vis à vis des autres technologies souhaitant l'utiliser.

LTE-U et LTE-LAA Deux techniques sont proposées afin d'apporter directement des *stations de base* sur la bande des $5GHz$: LTE-U (LTE dans les bandes non licenciées — LTE *in Unlicensed spectrum*) et LTE-LAA (LTE avec accès assisté par les bandes licenciées — LTE *License Assisted Access*). La première cible les pays où la réglementation d'accès au canal n'impose pas une politique de LBT (Écoute avant transmission — *Listen Before Talk*) tant que l'accès est effectué de manière équitable — il s'agit notamment des États-Unis, de la Corée du Sud et de l'Inde. La seconde utilise un LBT afin d'accéder au canal, et est donc applicable dans la plupart des pays européens et du Japon où un tel mécanisme est obligatoire.

Dans tout les cas, LTE-U et LTE-LAA procèdent à une mesure d'énergie sur les différents canaux $5GHz$ afin de trouver un canal non utilisé. Si un tel canal est trouvé, alors il sera occupé entièrement afin d'y opérer une petite cellule. Si au contraire aucun canal libre n'est trouvé, alors LTE-U et LTE-LAA vont avoir recours à des mécanismes différents afin d'assurer une occupation et un accès au canal équitable. LTE-U applique une approche *Duty Cycle* [49] à LTE à travers un algorithme nommé CSAT (Écoute du canal pour adapter la transmission — *Carrier-Sensing Adaptive Transmission*). Celui-ci est en charge de sonder continuellement le canal afin d'estimer son taux d'utilisation par 802.11 et d'adapter son taux de *Duty Cycle* en conséquence. Si le canal est libre, alors la *station de base* LTE-U passe en mode LTE « ON » et commence l'envoi de messages. LTE-LAA opte pour une approche de type LBT lors de laquelle une *station de base* LTE-LAA effectue

un CCA (Évaluation d'un canal libre — *Clear Channel Assessment*) avant chaque transmission [50]. La *station de base* LTE-LAA doit alors sonder le canal pendant un temps de *backoff* aléatoire de n CCA slots — un slot valant $9\mu s$, et n étant borné à la valeur de la fenêtre de contention de la *station de base*. Si de l'énergie est détectée lors du *backoff*, alors celui-ci est stoppé jusqu'à ce qu'il n'y ait plus d'énergie détectée, le *backoff* est alors repris après une période fixe — $43\mu s$: $16\mu s$ plus trois durées d'un slot de CCA — afin de ne pas perturber les acquittements 802.11. Une fois le *backoff* effectué, la transmission s'effectue depuis la *station de base*. Si le rapport de cette transmission indique un fort taux d'erreurs de décodage — 80% — alors la fenêtre de contention est doublée, sinon elle retourne à sa valeur par défaut. Enfin un *backoff* post-transmission est effectué afin d'éviter de monopoliser le médium radio. Dans le cas où plusieurs canaux $5GHz$ seraient utilisés, la version 13 du 3GPP (Groupement d'organismes de standardisation pour les réseaux mobiles — 3^{rd} *Generation Partnership Project*) propose deux mécanismes de *backoff* comme illustré sur la FIGURE 1.4. Le premier consiste à effectuer un *backoff* complet sur seulement l'un des canaux et faire un CCA rapide — vérifier que le canal est libre depuis un temps PIFS — sur les autres, afin d'utiliser tout les canaux disponibles simultanément (FIGURE 1.4a). Le deuxième effectue un *backoff* complet sur l'ensemble des canaux utilisés et transmet uniquement sur ceux ayant fini leur *backoff* à l'issue d'un temps d'attente commun (FIGURE 1.4b). Une transmission LTE-LAA repose sur un mécanisme de TXOP (Possibilité de transmettre pendant une fenêtre temporelle donnée — *Transmit Opportunity*) pouvant monopoliser le canal jusqu'à $10ms$ en fonction du type de trafic. Lors des premières itérations de LTE-U, les périodes pendant lesquelles la *station de base* n'était pas activement en train de transmettre devaient servir à transmettre des *Almost Blank Subframes* [51] contenant des messages de contrôle. Ces messages pouvant tout de même altérer les transmissions 802.11, il a été décidé d'utiliser le mécanisme d'activation/désactivation d'agrégation de porteuses [52] afin d'arrêter la *station de base* opérant sur les bandes non licenciées pendant les périodes où celle-ci ne doit pas communiquer.

Dans la pratique, la coexistence entre les technologies LTE et WI-FI n'est pas entièrement assurée. Chai et al. [53] ont cherché à mesurer l'impact que pouvait avoir la version 13 de LTE-U sur le WI-FI. Ils considèrent dans leurs travaux que la coexistence est « équitable » si l'impact d'une *station de base* est similaire à celle d'un *point d'accès*. Ils ont tout d'abord cherché à vérifier si le mécanisme de CCA des équipements 802.11 était suffisant pour détecter les périodes de transmission d'une *station de base*. Pour cela ils configurent une *station de base* afin qu'elle émette en continu et de telle sorte que les équipements 802.11 reçoivent une énergie en dessous de leur seuil de CCA fixé à $-62dB$. Les équipements ne sont pas alors en mesure d'observer que le canal est occupé et vont démarrer leurs transmissions qui seront sujettes à des interférences et collisions réduisant drastiquement leurs débits. Ils considèrent ici qu'il ne s'agit pas d'un partage équitable car dans un environnement entièrement 802.11 et dans les mêmes conditions, les équipements 802.11 vont détecter leurs collisions sur une trame et démarrer une procédure de *backoff* afin de réduire la probabilité de collision lors de leur prochaine émission. Ici de par la nature du fonctionnement de LTE-U, seulement les équipements WI-FI effectuent un *backoff* pendant la période où la *station de base* émet. Les auteurs pointent par ailleurs du doigt que dans leurs scénarios de tests, lorsqu'uniquement des équipements 802.11 communiquaient, plus de la moitié des transmissions étaient effectuées en dessous du seuil de CCA. Ils critiquent par ailleurs la politique de *Duty Cycle* adoptée. Dans les mêmes conditions les auteurs comparent plusieurs politiques de *Duty Cycle* : deux slots d'émission / deux slots d'extinction, quatre slots, dix slots et cinquante slots. Ils constatent que plus les durée sont courtes, plus les débits observés en 802.11 sont faibles. En effet, pendant les périodes où la *station de base* émet, les équipements 802.11 vont entrer en collision et vont alors augmenter leurs fenêtres de contention respectives. Lorsque le canal redevient disponible, ils doivent attendre pendant une période de *backoff* relativement longue comparativement à la durée pendant laquelle la *station de base* n'accède pas au canal. Au contraire, lorsque les durées de basculement sont longues, les équipements 802.11 vont toujours entrer en collision, mais lorsque le canal redeviendra libre ils auront plus de temps pour émettre, une fois leurs *backoff* effectué, augmentant alors le débit mais aussi la gigue. Afin de palier à ce problème, les auteurs proposent ULTRON (*Unlicensed* LTE *RadiO Node*). Une *station de base* indique aux équipements 802.11 voisins qu'elle va commencer à transmettre en émettant une trame CTS à elle même. Dans leurs mesures ils observent que les équipements 802.11 reçoivent cette trame de manière fiable — 95% de taux de réception pour des équipements recevant une puissance de $-85dB$. Ceci permet d'obtenir un meilleur débit global car les *stations* qui ne peuvent pas effectuer un CCA vont quand même arrêter de transmettre et donc éviter des collisions augmentant leurs fenêtres de contention.



(a) Déploiement intégré

(b) Déploiement délocalisé

FIGURE 1.5 – Déploiements LWA

Dans une autre étude [54] évaluant la coexistence de ces deux technologies, les auteurs ont testé l'impact de la largeur de la bande utilisée par une *station de base*— $5MHz$, $10MHz$, $15MHz$ et $20MHz$ — et constatent que la variation de celle-ci n'affecte pas le débit observé en 802.11 mais uniquement celui observé en LTE-U. Comme pour l'étude de Chai et al. [53] ils ont testé plusieurs modèles de *Duty Cycle* assurant à chaque fois une occupation temporelle de 60% en faveur de la *station de base* et font le même constat que leurs pairs à savoir que plus le nombre de slots consécutifs pendant lesquels un équipement 802.11 peut communiquer est grand, plus son débit sera élevé. Ils ont par ailleurs cherché à voir l'impact des transmissions 802.11 sur le trafic observé en LTE-U. Une dégradation du débit reçu est notée uniquement lorsque les appareils 802.11 émettent avec de fortes puissances, et que la *station de base* utilise une modulation peu robuste aux interférences.

Mukherjee et al. [50] se sont intéressés à l'évaluation de la coexistence entre 802.11 et LTE-LAA. Dans leurs simulations ils s'intéressent à un déploiement en intérieur constitué de *points d'accès* et de plusieurs *stations de base* sur lesquels une vingtaine d'utilisateurs vont générer du trafic en FTP (Protocole de transfert de fichier — *File Transfer Protocol*) et VOIP. Leurs résultats montrent qu'une coexistence entre ces deux technologies est possible, et que LTE-LAA n'a pas un impact significatif sur les communications 802.11.

D'autres simulations effectuées par Chen et al. [55] dans un scénario constitué d'un *point d'accès* et d'une *station de base* montrent que la probabilité de transmissions réussies est relativement équitable lorsque le trafic n'est pas trop important, mais est largement en faveur de la *station de base* LTE-LAA lorsque la charge augmente. Ils observent que ce phénomène est amplifié lorsque le nombre de *points d'accès* et *stations de base* concurrentes augmente.

LTE-WLAN LWA (Agrégation LTE-WLAN — LTE-WLAN *Aggregation*) [56, 57] est une autre proposition du 3GPP qui contrairement à la LTE-U et la LTE-LAA propose d'encapsuler des trames LTE à l'intérieur des paquets 802.11. De ce fait LWA est entièrement compatible avec la couche MAC 802.11— le 3GPP a d'ailleurs fait enregistrer un *EtherType* spécifique. LWA spécifie un mécanisme particulier afin de passer du protocole LTE PDCP (Protocole de convergence de paquets de données — *Packet Data Convergence Protocol*) [58] à un paquet 802.11 et vice versa : LWAAP (Protocole d'adaptation pour LWA — LWA *Adaptation Protocol*) [59]. L'*offloading* en 802.11 est décidé par l'opérateur en envoyant un message à l'utilisateur depuis la *station de base* à laquelle il est actuellement associé. Lors de cette activation la *station de base* envoie une liste d'identifiants de réseaux sans fil — WLAN *(Réseau local sans fil* — Wireless Local Area Network*) Mobility Set* — que l'équipement peut utiliser sans avoir à faire de demande explicite d'association. Il existe deux types de déploiement comme illustré sur la FIGURE 1.5 : « intégré » dans lequel la *station de base*, le WT (Terminaison WLAN — WLAN *Termination*) et l'entité en charge du contrôle d'accès sont présents au sein du même équipement (FIGURE 1.5a) — idéal pour des petites cellules — ou « délocalisé » dans lequel la *station de base* et les WT / contrôleurs d'accès sont reliés entre eux par une interface XW UP (Protocole Xw sur le plan utilisateur servant à échanger des messages de contrôle concernant les flots de données utilisateurs — XW *User Plane*) [60] (FIGURE 1.5b) — ce qui permet d'utiliser l'infrastructure 802.11 existante.

1.1.3.2 Propositions du standard 802.11

802.11P Dans le contexte des ITS (Systèmes de transports intelligents — *Intelligent Transportation Systems*) des besoins de communications à faible latence particuliers sont apparus. En effet, les véhicules se déplaçant à grande vitesse, il faut pouvoir réagir dans les plus brefs délais. 802.11P [61, 62, 63] introduit la

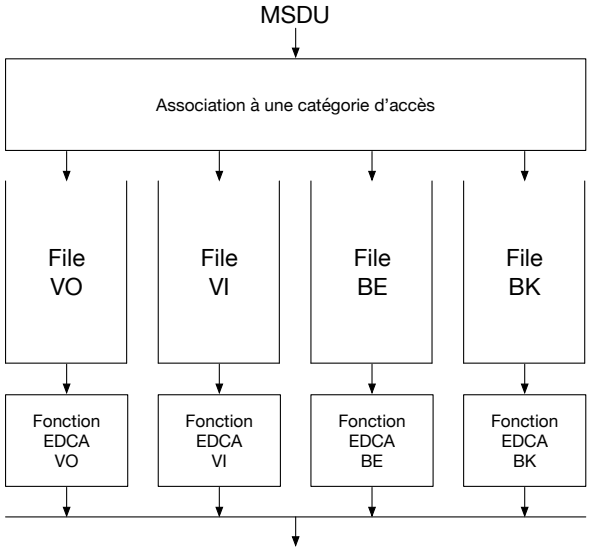


FIGURE 1.6 – Différenciation du trafic et accès au canal en utilisant EDCA

possibilité pour une *station* de communiquer en dehors d'un BSS (Association d'un *point d'accès* et d'une ou plusieurs *stations* — *Basic Service Set*). Cela signifie qu'aucune authentification ou association n'est nécessaire afin de transmettre un message depuis une *station* vers un *point d'accès*. Cet ajout permet en effet de s'affranchir de mécanismes de *handover*, qui est un processus lent, et permet aussi d'être réactif en envoyant un message dans l'air dès que son émission est nécessaire. Ceci se caractérise dans le standard par l'ajout d'une nouvelle variable — *dot11OCBActivated* — à la MIB (Base d'information pour la gestion du réseau — *Management Information Base*). Dans ce scénario, une *station* 802.11P peut émettre et recevoir des trames utilisant le « wildcard » BSSID, permettant à des véhicules opérant sur un même canal de communiquer immédiatement. Cela signifie par contre que la gestion de l'authentification n'est plus gérée au niveau MAC, mais par des protocoles de plus haut niveau définis par les acteurs des ITS. Afin de limiter les interférences, 802.11P opère sur une bande licenciée à $5.9Ghz$. Des modifications ont par ailleurs été apportées au standard afin d'utiliser des canaux ayant une largeur de bande de $10MHz$. Les véhicules n'ont pas besoin de scanner leur entourage car les différents canaux sont utilisés pour différentes applications, un canal étant réservé au contrôle, deux aux applications relatives à la sécurité routière, les autres pour la transmission de données pour différents services. Pour l'accès au canal, celui-ci est effectué par CSMA/CA (Écoute d'un support à accès multiple et à évitement de collision — *Carrier Sense Multiple Access/Collision Avoidance*) et EDCA. Le mécanisme d'EDCA est similaire à celui introduit par 802.11E [64] avec une mise en concurrence de l'accès au canal par les différentes *stations*, mais aussi avec une mise en concurrence interne en divisant le trafic selon quatre classes de priorités illustrées sur la FIGURE 1.6 : BK (File d'attente correspondant au trafic d'arrière plan peu prioritaire — *Background*), BE (File d'attente correspondant au trafic "normal " avec une priorité normale — *Best Effort*), VI (File d'attente correspondant au trafic vidéo avec une priorité élevée — *Video*) et VO (File d'attente correspondant au trafic transportant de la voix avec une priorité très élevée — *Voice*). La mise en concurrence interne est effectuée par une fenêtre de contention variant selon les priorités comme illustré sur le TABLEAU 1.1. Cependant, afin de ne pas mettre en concurrence du trafic lié à la sécurité routière avec du trafic moins sensible, EDCA est effectuée à la fois sur les canaux de contrôle et sur les canaux de services.

AC	CWMIN	CWMAX
AC_BK	$aCWmin$	$aCWmax$
AC_BE	$aCWmin$	$aCWmax$
AC_VI	$(aCWmin+1)/2-1$	$aCWmin$
AC_VO	$(aCWmin+1)/4-1$	$(aCWmin+1)/2-1$

TABLEAU 1.1 – Valeurs des fenêtres de contention pour les différentes files d'attente utilisant EDCA

Choi et Lee [65] s'intéressent au *handover* en 802.11P. Dans leurs simulations, ils considèrent des RSU

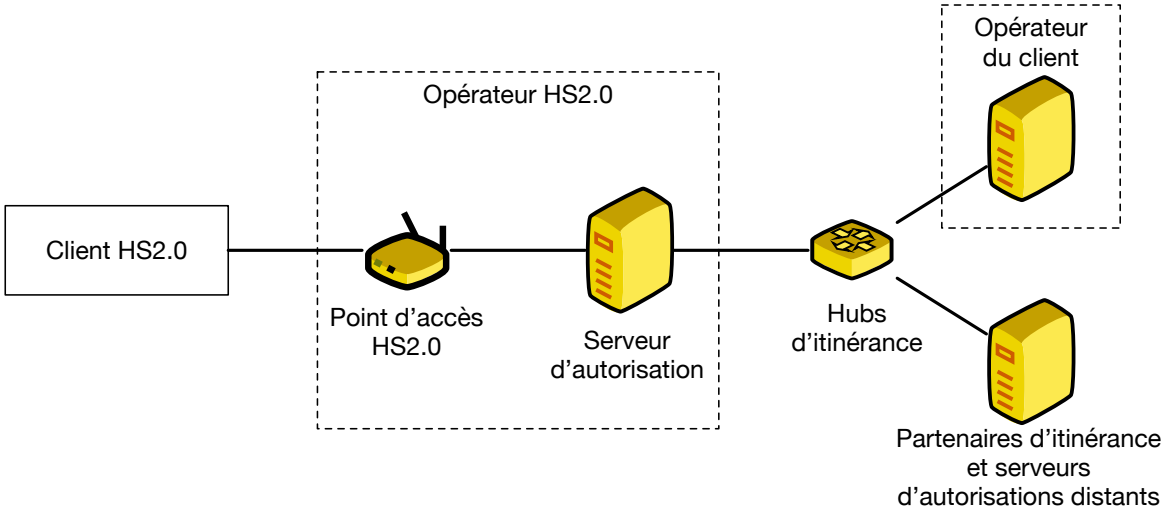


FIGURE 1.7 – Architecture d'*Hotspot 2.0*

(Équipement de bord de route — *Roadside Unit*) séparés de plusieurs centaines de mètres et s'intéressent entre autres à la durée de réception lorsqu'un véhicule passe d'un RSU à un autre. Ils remarquent que si la distance séparant les RSU augmente, la durée de non connectivité augmente également et est comprise entre $0.15s$ et $0.5s$. Ils font l'hypothèse que dans un déploiement de RSU, les identifiants de ceux-ci sont successifs, ou que du moins il est possible de déterminer leur schéma d'adressage. Ceci leurs permet d'envoyer un message de « disassociation » au RSU lorsque celui-ci va sortir de portée. Le RSU va alors stopper ses transmissions en cours, et va les faire suivre au RSU suivant afin que celui-ci puisse les envoyer au véhicule. Ceci leur permet d'avoir des temps de déconnexion compris entre $0.15s$ et $0.4s$.

WI-FI Direct Le WI-FI *Direct* [66] est une spécification technique de la *Wi-Fi Alliance* appelée aussi WI-FI *P2P*. Dans ce mode les différents équipements 802.11 se connectent les uns aux autres afin de former un groupe. Les différents équipements WI-FI *Direct* négocient leurs rôles lors de leur connexion : un *point d'accès* appelé *P2P Group Owner* et des clients appelés *P2P Clients*. Une mécanique intéressante de WI-FI *Direct* est l'inclusion de la réduction de l'espace de recherche au sein de la spécification. En effet lors de la phase de recherche, les équipements vont effectuer des recherches actives et passives sur des *Social Channels*. Ces canaux ne sont autres que les canaux orthogonaux du standard dans la bande des $2.4GHz$: canal 1, canal 6 et canal 11. Une fois trouvés, la négociation des rôles démarre et l'équipement ayant le rôle de *P2P Group Owner* choisit un canal 802.11 sur lequel le groupe va opérer.

Dans leurs travaux [67], Camps-Mur et al. évaluent le temps de découverte et de mise en place d'un tel réseau. Ils observent que la découverte peut prendre une dizaine de secondes malgré le fait que les *Social Channels* soient bien connus. En effet lorsque deux périphériques veulent se connecter ensemble, ils alternent leurs états de recherche active et passive indépendamment l'un de l'autre. Il faut donc que l'un des périphériques soit en train d'envoyer des *Probe Requests* pendant que l'autre écoute pour que la découverte ait lieu. De plus, le délai avant la découverte est retardé de $3s$ sur leur matériel car les périphériques effectuent d'abord une recherche complète sur les différents canaux 802.11 afin de voir si un groupe correspondant ne serait pas existant sur un canal.

Hotspot 2.0 La *Wi-Fi Alliance* a initié une spécification technique appelée *Hotspot 2.0* basée sur 802.11U [68] ainsi que son programme de certification nommé *Passpoint*. Le but de *Hotspot 2.0* [69, 70, 71, 72] est de mieux informer les terminaux 802.11 de leurs réseaux 802.11 environnants — à l'instar de 802.11K sans se limiter à un même ESS — afin d'améliorer la sélection de *points d'accès*, le *roaming*, l'*offloading* ou encore d'améliorer l'intégration avec les opérateurs de réseaux. Les ajouts majeurs de 802.11U sont la possibilité pour une *station* d'interroger un *point d'accès* lors de sa phase de recherche afin d'obtenir plus d'informations sur celui-ci avant de s'associer, ainsi que l'enrichissement des *beacons* envoyés par les *points d'accès*. Les *points d'accès* 802.11U incluent un *Interworking Element* dans leurs *beacons*. Cet élément contient trois groupes d'informations : *Access Network Option* contenant des informa-

tions sur l'accès au réseau, *Venue Info* contenant des informations sur « les raisons d'existence » du *point d'accès* et HESSID (Ensemble de BSS reliés aux mêmes opérateurs — *Homogenous Extended Service Set*) identifiant un groupe de BSS donnant accès aux mêmes réseaux extérieurs. L'*Access Network Option* définit notamment si le réseau est privé, privé avec des invités, public payant, public gratuit, s'il s'agit d'un réseau d'équipements personnels, si le réseau permet un accès à Internet ou s'il s'agit d'un réseau pour les urgences uniquement. Enfin s'il s'agit d'un réseau ouvert, il indique si des étapes supplémentaires pour l'accès sont nécessaires. Il indique par exemple les conditions d'utilisation ainsi que l'URL à utiliser pour s'authentifier. *Venue Info* donne des groupes génériques sur l'appartenance au *point d'accès* tel que *Éducation, Industrie, Extérieur, Commerce*, ainsi que des groupes plus précis comme *Bibliothèque, Café, Stade* afin de permettre à un utilisateur d'affiner la sélection de ses *points d'accès*. Un autre élément est introduit : le *Roaming Consortium Element*. Celui-ci sert à grouper les opérateurs ayant une politique de *roaming* commune afin qu'un utilisateur puisse s'authentifier auprès des infrastructures de ces différents opérateurs. La FIGURE 1.7 présente un client connecté à un *point d'accès Hotspot 2.0* lui permettant de s'authentifier auprès de son opérateur. Un *point d'accès* va donc inclure dans ses *beacons* les consortiums et/ou opérateurs accessibles par son biais. Les *Roaming Consortiums* sont identifiés par un identifiant d'organisation assigné par l'IEEE. Lors de sa phase de recherche un équipement va recevoir une liste d'identifiants et va pouvoir choisir un point d'accès en fonction de sa politique de *roaming* et des opérateurs accessibles. Les *beacons* ne pouvant pas contenir l'ensemble des informations, un protocole d'interrogation des *point d'accès* est proposé par 802.11U. Les *stations* peuvent utiliser des GAS (Service permettant le transport de trames dans un contexte non authentifié — *Generic Advertisement Service*) *Action Frames* afin d'envoyer des messages ANQP (Protocole permettant aux *stations* d'interroger les *points d'accès* sur leurs politiques d'accès — *Access Network Query Protocol*) à un *point d'accès* en particulier. L'avantage de l'utilisation de messages GAS est qu'ils ne nécessitent pas qu'une *station* soit associée à un *point d'accès*. En plus d'interroger un *point d'accès* sur les éléments vus précédemment, les ANQP *Elements* permettent à une *station* de demander des informations supplémentaires telles que la présence d'adresses IPV6 et/ou IPV4, la position GPS du point d'accès, son adresse postale, etc. *Hotspot 2.0* enrichit cette liste d'interrogations avec notamment des informations concernant la qualité du lien WAN (Réseau étendu — *Wide Area Network*) ou les restrictions de connexion telles que les ports ouverts et protocoles autorisés. Une fois la sélection effectuée par une *station*, celle-ci doit s'authentifier et s'associer à un *point d'accès* de manière classique. Cependant *Passpoint* exige que cette phase soit sécurisée par WPA2 (2ème version de WPA — WI-FI *Protected Access II*) et spécifie quatre types d'authentifications EAP : EAP-SIM (EAP utilisant l'identifiant contenu dans la SIM afin d'authentifier un utilisateur — EAP *Subscriber Identity Module*), EAP-AKA (EAP utilisant l'USIM (UMTS *Subscriber Identity Module* afin d'authentifier un utilisateur — EAP *Authentication and Key Agreement*), EAP-TLS (EAP utilisant un certificat utilisateur afin d'authentifier celui-ci — EAP *Transport Layer Security*) et EAP-TTLS (Extension de EAP-TLS permettant de s'affranchir d'un certificat client, et d'utiliser un couple identifiant / mot de passe à la place — EAP *Tunneled Transport Layer Security*). Les opérateurs de téléphonie mobile seront enclin à utiliser EAP-SIM ou EAP-AKA car ils utilisent déjà les infrastructures d'authentification. Les autres acteurs pourront utiliser l'une des deux autres méthodes. EAP-TLS est intéressant car il permet d'authentifier un équipement avec un seul certificat. Cependant la génération et la gestion desdits certificats peuvent être compliquées. EAP-TTLS reposant sur un couple identifiant / mot de passe est l'option par défaut.

Les performances concernant le *handover* horizontal d'*Hotspot 2.0* n'ont pas beaucoup été évaluées. Cependant le coût supplémentaire dû à l'augmentation de la taille des *beacons* et *Probe Requests* a été estimé [73]. L'occupation du canal augmente avec le nombre de SSID annoncés par un *point d'accès*. De plus, en augmentant le nombre de *points d'accès* présents, l'occupation du canal augmente d'autant plus. Avec trois *point d'accès* annonçant chacun trois réseaux, l'occupation du canal augmente de 3%. L'auteur conclut en donnant des recommandations sur la diminution de la fréquence d'émission des *beacons* ou la réduction de la taille de ceux-ci afin de ne pas occuper le canal trop souvent car, dans des scénarios avec une forte densité de *points d'accès*, l'impact sur le débit global peut ne pas être négligeable.

Quelques évaluations concernant les possibilités d'utiliser *Hotspot 2.0* afin de décharger les réseaux mobiles ont été faites. Des expérimentations sur un usage en mobilité pédestre et véhiculaire [74] semblent indiquer que le coût supplémentaire des messages de contrôle introduits n'ont pas d'impact significatif sur les possibilités offertes en terme d'*offloading*. Cependant ces tests ont été effectués dans un scénario avec un seul *point d'accès*. Les auteurs informent par ailleurs qu'ils n'ont pas évalué les possibilités de *handover*

d'un *point d'accès Hotspot 2.0* à un autre. Il s'agit plutôt d'une comparaison pour un accès opportuniste. L'utilisation de *points d'accès Hotspot 2.0* peut être envisagée pour augmenter la capacité du réseau [75].

1.1.3.3 Déconstruction de la notion de point d'accès

Ubiquiti a développé une technique afin de limiter l'impact de la durée de *handover* avant que les amendements 802.11R, 802.11K et 802.11V se généralisent : *UniFi Zero-Handoff* [76]. Cette technique n'est actuellement plus utilisées dans les équipements *Ubiquiti* récents. *UniFi Zero-Handoff* permettait de faire croire aux *stations* que l'ensemble des *points d'accès* déployés n'étaient en fait qu'un seul et unique *point d'accès*. Pour cela les *points d'accès* devaient opérer sur le même canal 802.11 et partageaient la même adresse MAC. L'avantage d'une telle solution est que le processus de *handover* n'est plus nécessaire. En contrepartie l'ensemble des *stations* se font concurrence pour l'accès au canal, pouvant entrainer une baisse de performance lors d'un déploiement dense. De plus, il faut un contrôleur central capable de gérer le fait que plusieurs *points d'accès* peuvent entendre une même *station*.

Une autre approche est de passer du paradigme où une *station* se déplace d'un *point d'accès* à un autre à celui où le *point d'accès* suit la *station* [77]. Pour chaque *station* associée dans le réseau, celui-ci crée un *point d'accès* virtuel qui la suivra d'un *point d'accès* physique à un autre. Les auteurs proposent le framework PACMAC afin de pouvoir passer les paramètres d'un *point d'accès* à un autre. Les *points d'accès* opèrent sur le même canal et maintiennent une liste de *stations* gérées ainsi qu'une liste de *stations* observées. Lorsqu'une *station* communique avec le *point d'accès* qui est actuellement en train de la gérer, celui-ci enregistre le RSSI reçu et l'ajoute dans ses *beacons*. Les *points d'accès* environnants sont alors capables d'entendre ces *beacons* et de savoir avec quelle intensité le *point d'accès* reçoit les messages de la *station*. Si un *point d'accès* environnant entend une communication de la *station* avec une intensité supérieure — à partir d'un certain seuil — à celle du *point d'accès* qui gère cette *station*, alors il va la rajouter dans sa liste de *stations* gérées et enverra le niveau d'énergie reçue par cette *station* dans ses *beacons*. Les autres *points d'accès* vont alors entendre ces *beacons* et le *point d'accès* qui était précédemment en charge de cette *station* va la retirer de sa liste de *stations* gérées. Cette solution permet de ne plus avoir à faire de *handover* et leurs résultats indiquent qu'il n'y a plus de perte de connectivité lorsqu'une *station* change de *point d'accès*.

Une extension [78] de ces travaux est proposée afin de pouvoir utiliser des *points d'accès* sur différents canaux. Comme ces *points d'accès* opèrent sur des canaux différents ils ne peuvent plus profiter de l'écoute opportuniste pour savoir qu'une *station* entre dans leur portée radio, cependant ils sont reliés entre eux par une interface ethernet comme dans le cas d'un réseau d'infrastructure. Une autre approche est donc proposée. Lorsque le *point d'accès* auquel la *station* est connectée observe une diminution du signal reçu de celle-ci, alors le *point d'accès* envoie un message de demande de recherche aux autres *points d'accès* par le biais de son interface ethernet. Ceux-ci vont alors changer de canal pour se mettre sur celui du *point d'accès* émetteur afin d'écouter s'ils entendent la *station*. Si un *point d'accès* entend la *station* alors il répond à la demande de recherche avec le niveau de signal observé. L'émetteur choisit alors le *point d'accès* avec le meilleur niveau de signal et lui envoie un message l'informant que la *station* va lui être « envoyée ». A la réception de ce message, le *point d'accès* va commencer à émettre des *beacons* avec les bons paramètres pour faire croire à la *station* qu'il s'agit de son ancien *point d'accès*. Le *point d'accès* émetteur va envoyer un message de type CSA (Annonce de changement de canal — *Channel Switch Announcement*) afin de forcer la *station* à changer de canal vers celui de son nouveau *point d'accès*. La *station* change alors de *point d'accès* physique mais reste connectée au même *point d'accès* virtuel.

BIGAP [79] reprend l'idée d'utiliser des messages CSA afin de faire passer une *station* d'un point d'accès à un autre, mais n'utilise pas la notion de *points d'accès* virtuels. À la place ils utilisent un mécanisme similaire à *Ubiquiti Zero-Handoff* et n'utilisent qu'un seul BSSID pour l'ensemble de leurs *points d'accès*. Pour la détection de la mobilité des *stations*, les *points d'accès* utilisent deux interfaces radios. L'une d'entre elles sert à faire fonctionner le *point d'accès* alors que l'autre est utilisée pour sonder les différents canaux de manière périodique afin de détecter si une *station* entre à portée radio. Ces informations sont remontées à un contrôleur central qui s'occupe de déclencher le *handover* lorsqu'une *station* s'éloigne de son *point d'accès* et s'approche d'un nouveau. Le nouveau *point d'accès* va alors effectuer une « fausse » association de la *station* avec les paramètres utilisés par l'ancien *point d'accès* afin de l'avoir dans sa liste de *stations* gérées avant que celle-ci change de canal. Le contrôleur va ensuite demander à l'ancien *point d'accès* d'émettre un CSA à la *station* concernée afin que celle-ci change effectivement de canal.

Une approche similaire [80] propose d'utiliser des *points d'accès* uniquement sur les canaux orthogonaux — 1, 6 et 11 — de 802.11, et d'équiper les *points d'accès* de trois interfaces radios, chacune d'elles opérant sur l'un de ces canaux, une servant à faire fonctionner le *point d'accès* et les deux autres à sonder de manière continue l'environnement. Le mécanisme mis en jeu afin d'effectuer le *handover* est similaire à celui de BIGAP, sauf que les *points d'accès* ne partagent pas la même adresse MAC. Il faut donc s'assurer que la *station* change l'adresse de son *point d'accès* au moment où le *handover* est déclenché.

1.1.3.4 Discussion

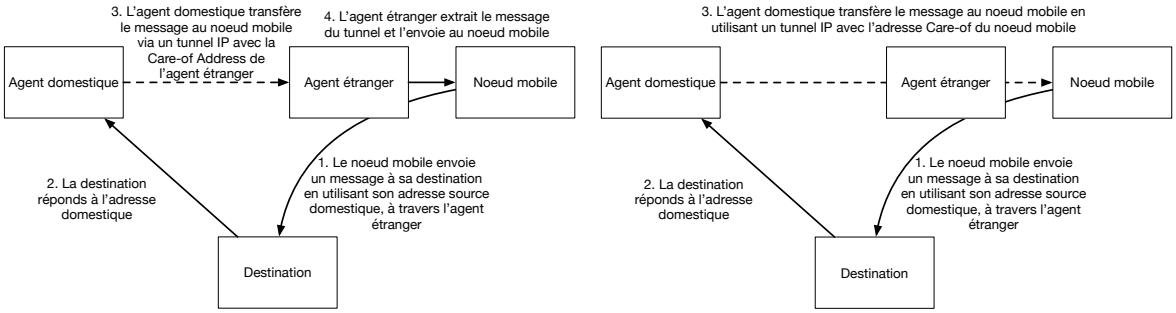
L'apport de la LTE sur les bandes non licenciées permet d'ajouter les avantages de la gestion des réseaux cellulaires en matière de *handover* sur ces bandes. Que ce soit en LTE-U, LTE-LAA ou LWA, ces techniques reposent toutes sur de l'agrégation de porteuse. Ceci implique que le périphérique est toujours connecté à un ENB et va donc être moins enclin à subir une rupture de connexion avec une *station de base* sur la bande licenciée. De plus cette connexion permanente permet de profiter pleinement de la gestion du *handover* dans le réseau. En effet c'est sur ce lien que les informations concernant les *points d'accès* environnant pour LWA d'une part, et les demandes de changement de *station de base* opérant sur les bandes non licenciées pour LTE-U et LTE-LAA d'autre part, circuleront. Cependant le déploiement sur ces bandes n'est pas forcément aisé. LTE-U et LTE-LAA visent un déploiement de micro / femto cellules pour augmenter la capacité globale du réseau demandant alors de l'investissement dans le matériel adéquat. Le respect de l'équité avec les autres protocoles opérant sur ces bandes de fréquences n'est par ailleurs pas garanti. LWA ne demande pas le déploiement de matériel dédié car l'utilisation de 802.11 pour encapsuler des trames LTE pourrait s'appuyer sur le parc 802.11 existant. De plus l'utilisation de 802.11 assure une équité avec celui-ci concernant l'accès au canal.

802.11P est intéressant car il propose un mécanisme de communication sans association. Lorsqu'un véhicule a besoin de transmettre de l'information, il se met sur le canal adéquat et après s'être assuré que celui-ci est disponible, il envoie son message. Ceci permet d'éviter d'avoir à rechercher les *points d'accès* disponibles dans le voisinage du véhicule tout en évitant la procédure d'association, ce qui permet de ne plus avoir de *handover*. L'authentification — et le temps nécessaire à celle-ci — n'est plus gérée au niveau MAC. Cependant 802.11P est fortement lié au monde des communications véhiculaires. Des contraintes sont mises concernant les modulations et largeurs de bandes afin de réduire les interférences survenant lors de déplacements à hautes vitesses rendant son utilisation au sein des équipements grand public délicate. Il faut par ailleurs équiper les routes afin de permettre aux véhicules de communiquer.

Hotspot 2.0 introduit la gestion des accords de *roaming* entre opérateurs au sein des *points d'accès*. Ceci permet de profiter de la couverture radio offerte par des *points d'accès* auxquels on ne pourrait pas se connecter normalement. Si le déploiement se généralisait on pourrait supposer que l'ensemble des *points d'accès* privés pourrait être utilisé à partir du moment où un accord de *roaming* existe. *Hotspot 2.0* essaye par ailleurs de permettre aux différents équipements d'améliorer la connaissance de leur environnement et de contextualiser la présence de *points d'accès* afin d'améliorer la prise de décision d'association à un *point d'accès*.

Le WI-FI *Direct* ne sert pas à résoudre les problématiques de mobilité, mais cherche à résoudre les difficultés à trouver et à s'associer à un équipement 802.11. Pour cela l'espace de recherche est réduit à l'utilisation des canaux orthogonaux. Cependant comme il s'agit d'un mécanisme servant à connecter des appareils en dehors de l'existence d'un *point d'accès*, les périphériques cherchant à se connecter oscillent entre la recherche active et passive pouvant augmenter le temps de découverte.

Enfin les techniques telles que *UniFi Zero-Handoff*, BIGAP ou les *points d'accès* virtuels cherchent à supprimer la notion de *point d'accès* en faisant croire aux *stations* que celles-ci sont connectées à un unique *point d'accès*, évitant ainsi d'avoir à effectuer un *handover*. Cependant des problèmes de passage à l'échelle peuvent survenir avec des déploiements reposant sur *UniFi Zero-Handoff* car un seul canal est utilisé. De plus, pour l'ensemble de ces solutions, de l'intelligence doit être portée dans le réseau afin de savoir quel *point d'accès* « physique » est actuellement en charge d'une *station*, ou pour gérer la politique de *handover* lors de l'utilisation de CSA par exemple.



(a) *Mobile* IPv4 utilisant la *Care-of Address* de l'agent étranger

(b) *Mobile* IPv4 utilisant la *Care-of Address* du nœud mobile

FIGURE 1.8 – Émission et réception d'un message en utilisant *Mobile* IPv4

1.2 Le roaming aux niveaux réseau, transport et application

Bien que les travaux présentés dans cette thèse se focalisent sur le *handover* au niveau MAC, il nous semble pertinent de présenter de manière synthétique la gestion du *handover* sur les couches supérieurs du modèle OSI afin de mieux appréhender la complexité du problème de la mobilité dans son ensemble.

En effet, lorsqu'une *station* a effectué son *handover* au niveau MAC, celle-ci n'a pas encore récupéré sa connectivité. Elle va devoir récupérer une adresse IP, sans avoir la garantie de pouvoir utiliser son adresse précédente. Si des sessions TCP (Protocole de contrôle de transmissions — *Transmission Control Protocol*), UDP ou de n'importe quel autre protocole de transport étaient en cours, elles vont devoir être récupérées / réinitialisées afin de permettre aux applications les utilisant de continuer leur fonctionnement.

1.2.1 Le roaming IP

La problématique du *handover* IP d'un équipement 802.11 en mobilité se présente lors du changement d'ESS. Dans un déploiement d'infrastructure ce n'est pas forcement observable lors du changement d'un *point d'accès* à un autre car ils peuvent appartenir au même ESS. Dans un déploiement à l'échelle d'une ville les *points d'accès* disponibles n'appartiennent pas forcément à un même ESS et le fait d'effectuer un *handover* résulte donc en un changement d'adresse IP pour l'utilisateur.

Lors de ce *handover*, le changement d'adresse IP peut intervenir à deux niveaux. Premièrement, lorsque l'équipement mobile fait une demande d'adresse IP via DHCP (Protocole de configuration dynamique des hôtes — *Dynamic Host Configuration Protocol*) : il lui sera alors alloué une nouvelle adresse IP. Deuxièmement, si cette adresse IP est privée — non routable sur Internet, ce qui est fréquent dans les déploiements de réseaux WLAN domestiques — il utilisera l'adresse IP publique de son routeur par défaut par le biais de l'utilisation d'un NAT (Translation d'adresse réseau — *Network Address Translation*), ce qui change son adresse vue depuis Internet.

Dans les deux cas, le changement d'adresse IP risque de perturber les communications en cours car les protocoles de transport comme TCP et UDP récupèrent mal d'un changement d'adresse IP — arrêt de la session en cours ou envoi de messages vers un hôte inexistant.

Par ailleurs il faut noter que la négociation d'adresse IP par DHCP n'est pas instantanée. Même si celle-ci peut être accélérée lorsque l'on rejoint un réseau connu, le temps nécessaire à celle-ci s'ajoute à la durée globale du *handover*.

1.2.1.1 Mobile IP

Mobile IP est un protocole de gestion du *handover* au niveau IP, pour les versions IPv4 (4$^{\text{ème}}$ version du protocole Internet — *Internet Protocol version 4*) et IPv6. Nous présentons le fonctionnement de Mobile IP dans ces deux versions : MIPv4 (IPv4 mobile — *Mobile* IPv4) et MIPv6 (IPv6 mobile — *Mobile* IPv6).

MIPv4 [81] a pour but de permettre à un équipement mobile d'utiliser toujours la même adresse IP, même quand celui-ci passe d'un sous-réseau IP à un autre. Pour cela MIPv4 introduit trois entités : un nœud mobile — *Mobile Node*, un agent domestique — *Home Agent* et un agent étranger — *Foreign Agent*. Le nœud mobile est un équipement qui change de sous-réseau lui permettant de se rattacher à Internet lors de son déplacement. Cependant, même s'il change de sous-réseau, il garde une adresse IP qui lui est propre et qui appartient à son réseau domestique. L'agent domestique est un routeur opérant sur le réseau domestique du nœud mobile — son réseau de « base » — et s'occupe de transférer les paquets de données au nœud mobile via un tunnel lorsque celui-ci est en dehors de son réseau domestique. Il maintient par ailleurs la localisation du nœud mobile sous la forme d'une *Care-of Address* qui représente l'adresse IP du réseau auquel le nœud mobile est actuellement rattaché. L'agent étranger est un routeur appartenant à un réseau autre que le réseau domestique sur lequel le nœud mobile est rattaché. L'agent étranger peut servir de routeur par défaut pour les messages émis par le nœud mobile. Le fonctionnement de MIPv4 lorsqu'un nœud mobile rejoint un réseau étranger peut être décrit comme suit. Tout d'abord le nœud mobile va solliciter le réseau afin de savoir si un agent est présent en envoyant des messages *Agent Solicitation*. Les agents peuvent par ailleurs diffuser régulièrement des messages *Agent Advertisement* ou simplement répondre aux sollicitations avec ces mêmes messages. Lorsque le nœud mobile reçoit une réponse d'un agent, il détermine s'il se trouve sur un réseau domestique ou étranger. S'il se trouve sur un réseau étranger, le nœud mobile reçoit l'adresse IP *Care-of Address* de l'agent étranger dans le message *Agent Advertisement* ou peut se « créer » une *Care-of Address* en utilisant l'adresse IP qui lui aura été donnée lors de l'association au réseau. Le nœud mobile envoie ensuite sa *Care-of Address* à son agent domestique par le biais de messages *Registration Request* / *Registration Reply*. Les messages envoyés depuis Internet vers le nœud mobile passent par l'agent domestique qui se charge de les transférer vers le réseau étranger via un tunnel utilisant l'adresse *Care-of Address*. Les messages sortent du tunnel, soit au niveau de l'agent étranger si son adresse est utilisée comme *Care-of Address* qui les transférera ensuite au nœud (FIGURE 1.8a), soit directement sur le nœud mobile si celui-ci utilise une adresse *Care-of Address* auto affectée (FIGURE 1.8b). Pour les messages émanant du nœud mobile en direction d'Internet, ils ne passent pas forcément par un tunnel vers le réseau domestique si le réseau étranger autorise l'émission de messages IP avec une adresse IP source différente de celles correspondant à son sous-réseau, sinon l'emploi d'un *Reverse Tunnel* [82] est nécessaire. Il faut noter ici que dans le cas où l'on n'utilise pas l'adresse de l'agent étranger comme *Care-of Address* il faut que celle donnée au nœud mobile soit une adresse routable sur Internet.

MIPv6 [83] permet d'utiliser un mécanisme de tunnels similaire à MIPv4 — en se séparant des agents étrangers —, mais permet aussi d'utiliser un autre mode appelé *Route Optimization* avec les correspondants compatibles. Dans ce cas, le nœud mobile enregistre son adresse de type *Care-of Address* auprès de son correspondant sur Internet. Les paquets échangés après utiliseront l'option IPv6 permettant d'ajouter une entête de routage — *Routing Header*. Lorsque le correspondant enverra un message au nœud mobile, il utilisera l'adresse *Care-of Address* comme adresse de destination et ajoutera une entête de routage contenant l'adresse du nœud sur son réseau domestique. De la même manière, lorsque le nœud mobile communique avec son correspondant, il utilise son adresse *Care-of Address* comme adresse source et ajoute une entête contenant une option de destination avec son adresse domestique. Ce mécanisme permet d'utiliser des routes « directes » entre le nœud mobile et son correspondant, sans avoir à passer par un agent domestique, en exploitant les extensions du protocole IPv6.

HMIPv6 (IPv6 mobile hiérarchique — *Hierarchical Mobile* IPv6) [84, 85], FMIPv6 (IPv6 mobile avec *handovers* rapides — *Fast handovers for Mobile* IPv6) [86] et PMIPv6 (IPv6 mobile utilisant un *proxy* — *Proxy Mobile* IPv6) [87] sont des améliorations proposées.

HMIPv6 part du constat que beaucoup de trafic de contrôle est envoyé sur Internet lorsque le nœud mobile change de *Care-of Address*. Cependant une partie de ce trafic pourrait être évité lorsque le nœud mobile ne fait que changer de point d'attachement au niveau MAC sans changer de sous-réseau IP — micro mobilité vs. macro mobilité. Pour cela ils introduisent une nouvelle entité — *Mobility Anchor Point* — résidant dans le réseau étranger. Lorsqu'un nœud mobile rejoint un réseau compatible, il reçoit l'adresse d'un ou plusieurs *Mobility Anchor Points* et utilisera cette adresse comme *Regional Care-of Address*. Il aura juste à informer son agent domestique ou son correspondant de son adresse *Regional Care-of Address* comme s'il s'agissait de son adresse *Care-of Address* et enregistrer son adresse locale sur le réseau auprès du *Mobility Anchor Point*. Celui-ci jouera alors le rôle d'un agent domestique sur le réseau étranger en recevant les messages de son correspondant ou de son agent domestique et en les transférant au nœud mobile. Lorsque

le nœud mobile bouge au sein d'un même réseau étranger, il a juste à informer le *Mobility anchor Point* de sa nouvelle adresse locale et n'a plus à envoyer du trafic sur Internet afin d'informer son correspondant ou son agent domestique car il n'a pas changé de *Regional Care-of Address*.

FMIPV6 est une approche *make before break* afin de réduire le temps d'établissement d'un nouveau tunnel lorsque le nœud mobile change de point d'attache au réseau. Lorsqu'un nœud mobile trouve un nouveau point d'attache — avec une recherche active ou passive en 802.11 par exemple — il demande quelle sera sa prochaine *Care-of Address* lorsqu'il rejoindra ce point d'attache, afin de pouvoir directement l'utiliser lorsque le changement sera effectué. Avant de rejoindre le nouveau point d'attache, il demande à son point d'attache actuel d'établir un tunnel entre sa *Care-of Address* actuelle et sa prochaine *Care-of Address* le temps que celui-ci informe son correspondant du changement d'adresse après s'être attaché au nouveau point d'attache.

PMIPV6 cherche à retirer la gestion de la mobilité au niveau du nœud mobile. Celui-ci a uniquement besoin d'utiliser une pile IP standard. La notion d'adresse domestique change aussi, car celle-ci n'aura de validité que dans un domaine PMIPV6, c'est-à-dire que lorsqu'un nœud mobile change de domaine, il change aussi d'adresse domestique. Cela dit, PMIPV6 vise des déploiements d'assez grande échelle, donc le renouvellement d'adresse ne devrait pas arriver aussi fréquemment que dans les scénarios établis dans *Mobile IPv6*. Deux entités sont donc définies, la *Mobile Access Gateway* qui est en charge de la gestion de la mobilité du nœud mobile et réside au niveau de son point d'attache au réseau, et la *Local Mobility Anchor* qui joue un rôle similaire à l'agent domestique sauf qu'elle réside sur le domaine PMIPV6. Cette entité est en charge d'assigner l'adresse domestique de chaque nœud mobile du domaine et gère une table de routage entre ces adresses et les *Mobile Access Gateway* auxquels ils sont rattachés. Lorsqu'un nœud mobile change de *Mobile Access Gateway*, celui-ci informe la *Local Mobility Anchor* de la présence de ce nouveau nœud afin de mettre à jour sa table de routage.

Des études [88, 89, 90] montrent que chacune de ces améliorations permettent de réduire le temps nécessaire au *handover* par rapport à la proposition standard de *Mobile* IPV6. HMIPV6 et PMIPV6 semblent permettre d'avoir le plus de gain sur la durée de *handover*, PMIPV6 ne permettant cependant pas de garder une adresse unique lorsque l'on change de domaine.

1.2.1.2 Host Identity Protocol

Host Identity Protocol [91, 92] vise à casser le fait qu'IP serve à la fois à localiser et identifier une machine sur le réseau. *Host Identity Protocol* se situe entre les couches réseau et transport et propose de reléguer IP à son rôle de localisation d'une machine sur le réseau — il ne s'attaque pas à la gestion des problématiques de routage — et introduit un identifiant cryptographique permettant d'identifier une machine qui serait utilisé par les couches supérieures à la place de l'adresse IP. L'identifiant se base sur un couple de clés privée / publique pouvant être générées par n'importe quel algorithme de cryptographie asymétrique — RSA (Algorithme de chiffrement asymétrique nommé par les initiales des trois inventeurs — *Rivest–Shamir–Adleman*) doit être supporté afin d'assurer une compatibilité minimale. La clé publique est considérée comme l'identifiant d'hôte — *Host Identifier* — mais un *hash* cryptographique de cette clé — *Host Identity Tag* — est utilisé pour représenter cet identifiant. C'est ce *hash* qui est utilisé par les protocoles des couches supérieures et il doit avoir trois caractéristiques : faire 128 bits afin de pouvoir être utilisé à la place d'une adresse IPV6, être robuste afin d'empêcher de retrouver facilement l'identifiant d'hôte à partir du *hash* et minimiser les probabilités de collisions entre deux *hash* représentant deux hôtes distincts. La taille fixe de ce *hash* permet de facilement l'intégrer aux protocoles des couches supérieures tout en leur permettant de ne pas avoir connaissance de l'algorithme de cryptographie sous-jacent.

Host Identity Protocol semble plus performant que les solutions *Mobile IPs* [93], notamment car il limite le nombre de messages de contrôle.

1.2.2 Le roaming au niveau transport

Dans le modèle « classique » de l'utilisation de TCP / IP ou UDP / IP, une connexion au niveau transport est identifiée par les couples d'adresses IP et de ports utilisés par les deux hôtes. Si un de ces éléments change alors une autre connexion est identifiée. Or si aucune solution de roaming IP — tel que présenté dans la section précédente — n'est utilisé, lors d'un changement de *point d'accès*, une *station* change

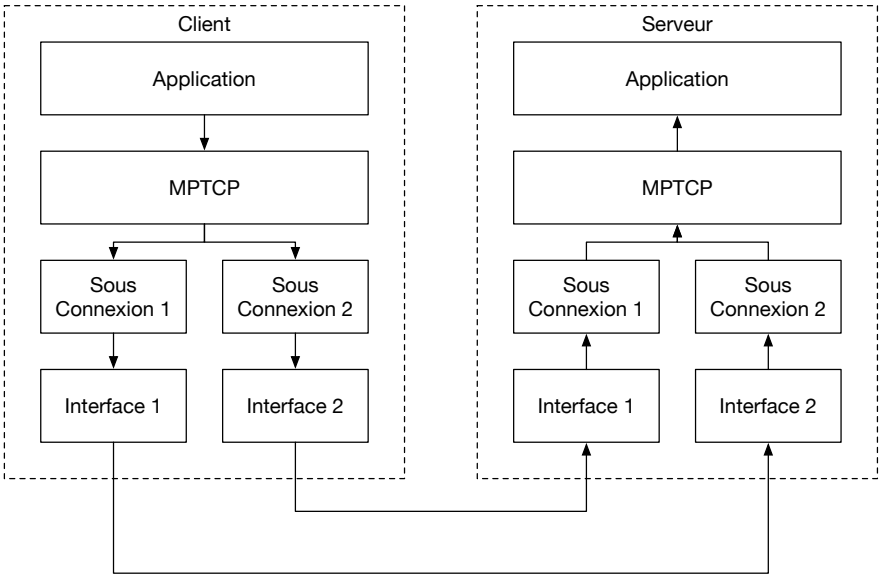


FIGURE 1.9 – Connexion *Multipath* TCP utilisant deux « sous-connexions »

d'adresse IP. Dans ce cas, l'ensemble des sessions TCP en cours ne sont plus utilisables et si une *station* veut continuer à communiquer avec sa destination elle doit réinitier la session précédente. UDP n'ayant pas d'état connecté, un changement d'adresse IP ne demande pas de réinitialisation de la session. Dans tous les cas, la gestion d'un tel changement doit être assurée au niveau applicatif afin de ne pas perturber l'expérience utilisateur.

Cette partie présente différents protocoles de transport permettant de gérer la mobilité au niveau IP sans affecter les couches applicatives.

1.2.2.1 Multipath TCP

Les équipements modernes ont le plus souvent plusieurs interfaces qui leurs permettent d'accéder à Internet— interface 802.11 et cellulaire sur les smartphones ou Ethernet et 802.11 sur les ordinateurs portables par exemple — et il serait intéressant de pouvoir profiter de ces interfaces simultanément afin d'atteindre une destination. MPTCP (TCP multi chemins — *Multipath* TCP) [94, 95] propose d'exploiter cette caractéristique au niveau de TCP afin d'établir une connexion empruntant plusieurs chemins différents. L'établissement d'une connexion MPTCP s'effectue de la même manière qu'une connexion TCP standard. Si les hôtes sont capables d'effectuer une connexion MPTCP, à savoir qu'ils possèdent plusieurs interfaces avec des adresses IP différentes, alors une autre « sous-connexion » TCP est démarrée sur ces interfaces comme illustré sur la FIGURE 1.9. Ces différentes « sous-connexions » sont agrégées par MPTCP afin d'apparaitre comme une unique connexion TCP au niveau de la couche applicative. Afin d'être sûre de pouvoir ré-assembler les différents segments TCP utilisant des chemins différents avec des latences différentes, MPTCP rajoute son propre numéro de séquence. Lors de l'établissement de la connexion MPTCP les deux correspondants s'envoient chacun une clé de soixante quatre bits. Des *hashs* cryptographiques de ces clés seront alors utilisés afin d'identifier cette connexion.

Le fait de pouvoir identifier une connexion MPTCP par un identifiant unique qui ne soit pas basé sur une adresse IP tout en étant indépendant de l'interface utilisée permet de faciliter le *handover* au niveau transport. MPTCP peut fonctionner suivant plusieurs modes [96] : MPTCP total où l'ensemble des chemins exploitables sont utilisés afin d'obtenir le meilleur débit possible, MPTCP en mode sécurité où une « sous-connexion » est ouverte sur l'ensemble des interfaces mais seulement un sous ensemble de ces « sous-connexions » est utilisé, et MPTCP en mode chemin unique où une seule « sous-connexion » est ouverte avec la possibilité de remplacer cette « sous-connexion » par une nouvelle sur une autre interface si le lien se dégrade. Dans leur étude Paasch et al. [96] étudient le *handover* vertical — d'une interface 802.11 vers une interface cellulaire — en le comparant suivant ces différents modes. Lorsqu'une application utilise MPTCP en mode total ou sécurité, la latence introduite est uniquement liée à la détection de la perte de signal lorsque l'équipement n'est plus à portée du *point d'accès* 802.11. Lorsque MPTCP est utilisé en mode

chemin unique il faut rajouter à cette latence le temps d'ouverture d'une nouvelle « sous-connexion ».

Le *handover* vertical MPTCP entre un lien 802.11 et un lien cellulaire sur des smartphones a été mesuré [97], et on constate que celui-ci peut prendre plusieurs dizaines de secondes. L'utilisation de MPTCP sur les smartphones [98] montre qu'une vaste majorité de ces équipements n'ouvrent qu'une seule « sous-connexion » et qu'un peu moins d'un tiers en ouvrent une deuxième. Les « sous-connexions » sont en majorité ouvertes moins de $200ms$ après avoir récupéré une adresse IP sur une interface. Il est par ailleurs observé que la plupart de ces « sous-connexions » ne sont pas utilisées pour envoyer du trafic. De plus la décision d'utiliser une « sous-connexion » par rapport à une autre est prise au niveau de l'émetteur et se base sur le RTT (Durée d'un aller-retour — *Round-Trip Time*) de celle-ci. Ceci revient à dire que cette décision est le plus souvent prise au niveau du serveur plutôt qu'au niveau du smartphone et n'est donc pas optimale dans un cadre de mobilité. De ces différents constats les auteurs proposent une amélioration de MPTCP appelée *MultiMob* [99]. Il s'agit un ordonnanceur ciblé pour l'usage mobile, prenant en compte la dernière « sous-connexion » utilisée par un smartphone. Si, par exemple, le dernier message reçu par le serveur provient du lien utilisant l'interface cellulaire, c'est probablement car l'interface 802.11 n'était pas en mesure d'envoyer le message. Il est donc préférable d'utiliser ce lien afin de transmettre des messages au smartphone même si le RTT a été mesuré comme moins bon précédemment. Ils indiquent aussi que l'utilisation d'une « sous-connexion » de sécurité n'est pas optimale en terme d'énergie car cela implique de maintenir une connexion active sur le réseau cellulaire. Ils proposent donc d'établir cette connexion uniquement lorsque le lien utilisant le 802.11 commence à défaillir. Pour cela, ils partent du postulat que, lorsque le lien se dégrade, ils vont observer des retransmissions au niveau du serveur. Ils intègrent donc un oracle se basant sur ces observations afin d'établir une « sous-connexion ».

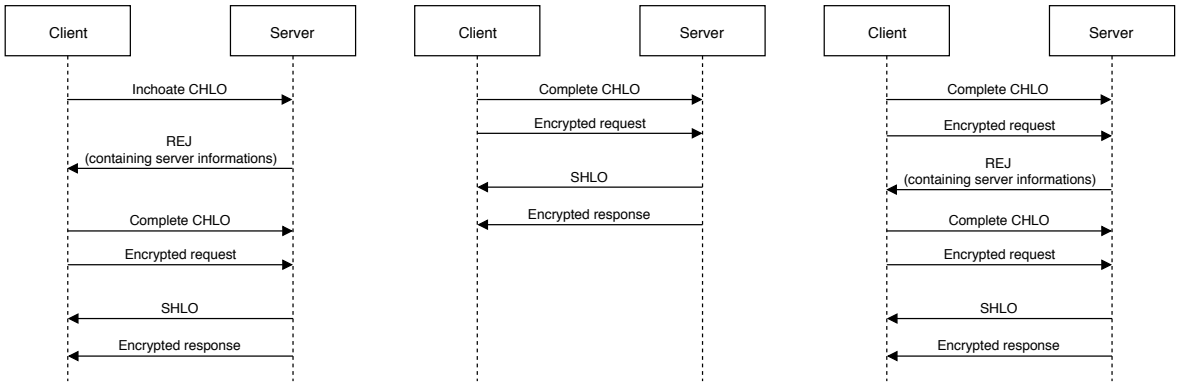
1.2.2.2 Stream Control Transmission Protocol

SCTP (Protocole de transmission des commandes de flux — *Stream Control Transmission Protocol*) [100, 101] est un protocole de transport orienté messages. C'est un protocole *unicast* permettant la communication entre deux hôtes, cependant ces hôtes peuvent avoir plusieurs adresses IP. Cette possibilité permet d'ajouter de la redondance et de mieux s'adapter lors de la perte d'un lien. Une de ces adresses est considérée comme primaire et les autres comme secondaires. Lorsque un message a besoin d'être retransmis, il est envoyé sur une des interfaces ayant une adresse secondaire. Lors du processus d'association entre deux hôtes, ceux-ci échangent une liste d'adresses utilisables. Les hôtes gardent une liste des adresses actives en envoyant régulièrement des messages sur l'ensemble des interfaces de leur correspondant. Si ces messages ne reçoivent pas suffisamment d'acquittements alors les interfaces correspondantes sont considérées comme inactives. L'interface principale peut devenir inactive si le nombre de retransmissions excède un certain seuil. Dans ces cas là une autre adresse de destination est utilisée.

La possibilité d'avoir plusieurs adresses IP correspondant à une même session SCTP permet comme pour MPTCP d'utiliser plusieurs interfaces différentes afin d'éviter une coupure franche de la connectivité lorsqu'un utilisateur est mobile. La possibilité de changer dynamiquement les adresses des hôtes [102] permet de s'adapter aux changements d'adresses lorsque l'équipement mobile change de réseau. Cette propriété a été utilisée dans MSCTP — *Mobile* SCTP [103] afin de permettre un handover pour des utilisateurs mobiles et reprise afin d'évaluer les possibilités de *handover* vertical entre un réseau 802.11 et un réseau cellulaire [104]. Un handover SCTP entre deux interfaces 802.11 basé sur le RTT afin de décongestionner les *points d'accès* a aussi été proposé [105].

1.2.2.3 QUIC

QUIC [106, 107] est un protocole de transport basé sur UDP. Lorsqu'un client se connecte pour la première fois à un serveur, il envoie un message *Inchoate Client Hello* auquel le serveur répond avec un message contenant l'ensemble des données cryptographiques nécessaires afin d'effectuer une connexion sécurisée. S'ensuit alors un *handshake* classique permettant d'initier la connexion lors duquel un client envoie un message *Hello* contenant ses informations cryptographiques et le message chiffré qu'il veut transmettre au serveur. Celui-ci répond avec un *Hello* et la réponse chiffrée. On a donc besoin d'un échange afin de récupérer les informations cryptographiques et un échange afin d'initier une connexion pour la première fois en utilisant QUIC. Cette opération est illustrée sur la FIGURE 1.10a et est appelée 1-RTT *handshake*.



(a) 1-RTT *handshake* initial

(b) 0-RTT *handshake*

(c) 1-RTT *handshake* lors de l'expiration des informations cryptographiques

FIGURE 1.10 – Établissement d'une connexion utilisant QUIC

Si le client a besoin d'effectuer une connexion avec le même serveur à l'avenir, le premier échange n'est plus nécessaire car il connait déjà les informations relatives au serveur. Le *handshake* cryptographique peut donc directement s'initier, permettant une connexion avec un seul aller-retour comme illustré sur la FIGURE 1.10b et est appelée 0-RTT *handshake*. Si les informations cryptographiques changent au niveau du serveur, lorsqu'un client tente de se connecter avec les anciennes informations, le serveur rejette cette connexion en lui envoyant les nouvelles informations comme illustré sur la FIGURE 1.10c. Une connexion QUIC est identifiée par un identifiant de connexion échangé lors de l'établissement de la connexion entre deux hôtes. Cet identifiant de connexion permet de s'assurer que celle-ci reste active lorsque les adresses IP des hôtes changent.

1.2.3 Le roaming au niveau application

Le *roaming* au niveau applicatif est surtout centré sur la récupération d'erreurs et l'intégration du fait que la connectivité ne peut pas être considérée comme constante et parfaite dans un contexte de mobilité [108].

L'adresse IP ne pouvant pas être utilisée comme un identifiant, les applications peuvent utiliser d'autres méthodes d'authentification afin de transmettre et de pouvoir recevoir des données en mobilité. Par exemple le système d'envoi de notifications d'Apple APNS (Service d'envoi de notifications d'Apple — *Apple Push Notification service*) [109] se base sur la génération d'un jeton unique par application et utilisateur. Lorsqu'une application s'enregistre auprès du serveur de notifications d'Apple, un jeton unique est généré. Ce jeton sera aussi partagé avec le service générant des données afin qu'il l'utilise pour identifier l'utilisateur pour lequel ces données sont générées. Lorsque des données sont générées par le service, il les envoie avec le jeton au serveur de notifications. Si l'application identifiée par ce jeton est connectée alors il transfère le message, sinon il le garde jusqu'à une prochaine connexion de celle-ci. Lorsque l'application se connecte aux serveurs de notifications d'Apple, elle fournit son jeton et si des messages en attente lui sont destinés, ils lui sont transférés.

Google, Amazon ou Microsoft utilisent un système similaire à base de jetons avec respectivement FCM (Service d'envoi de messages dans le cloud de Firebase — *Firebase Cloud Messaging*) [110], SNS (Service simple d'envoi de messages d'Amazon — *Simple Notification Service*) [111] et *Azure Notification Hubs* [112].

Les services de *streaming* tels que Spotify, Netflix, Youtube utilisent du pré-téléchargement afin de palier aux problèmes de connectivité. Lorsque l'équipement mobile accède au réseau, il va télécharger le contenu dont il a besoin dans l'immédiat mais aussi du contenu dont il aura besoin dans le futur. Pour une musique ou une vidéo cela revient à dire que plusieurs secondes d'avance sont téléchargées par rapport à la position dans la lecture actuelle. Cette technique permet de réduire l'impact d'une déconnexion franche en laissant du temps pour se reconnecter, ou d'une diminution du débit en laissant le temps d'adapter la qualité d'encodage. Spotify adapte sa politique de pré-téléchargement en fonction de la qualité de la connexion et du niveau de batterie [113], mais aussi en fonction de l'utilisation faite par l'utilisateur [114]. Si celui-ci a ten-

dance à ne pas écouter les morceaux jusqu'à la fin, ou à souvent passer au morceau suivant en début de chanson, l'application ne téléchargera pas forcément les morceaux entiers, mais plutôt les débuts des prochains morceaux. Netflix pré-télécharge les flux audio et vidéo ayant une forte probabilité d'être démarrés ensuite afin de ne pas avoir de temps d'interruption [115].

Les communications avec l'espace ont permis l'émergence d'un nouveau type de réseau : les DTN (Réseau tolérant au retard et à la coupure — *Delay-and-Disruption-Tolerant Networking*) [116, 117]. Les problématiques liées à de telles communications ne fonctionnent pas avec les suppositions faites par IP [118] à savoir :

- Un chemin entre deux destinations existe pendant la durée totale d'une transmission
- Les retransmissions sont un moyen efficace de palier aux erreurs
- Les pertes sur un chemin sont faibles
- Les routeurs et hôtes supportent TCP/IP
- Les applications n'ont pas besoin de se préoccuper des performances du réseau
- La sécurité aux extrémités de la communications est suffisante
- Une route unique est suffisante pour avoir des performances acceptables

Les DTN sont construits de manière à ne pas avoir à reposer sur ces suppositions. Le protocole *Bundle* [119] répond aux besoins des DTN et fonctionne sur la couche applicative, en jouant le rôle d'un *overlay network* généralement au dessus de TCP/IP mais sans être limité à cette pile protocolaire. Il inclue notamment les concepts de *store-and-forward* permettant de stocker un paquet dans le réseau tant que les chemins vers un hôte sont rompus. Les DTN ont été évalués pour des usages en mobilité [120, 121, 122, 123].

1.2.4 Discussion

Les techniques de mobilité au niveau d'IP n'ont pas été largement adoptées [124, 125, 126]. *Mobile* IP comme *Host Identity Protocol* sont complexes et modifient de nombreux éléments dans la pile protocolaire TCP/IP. Ils demandent aussi le déploiement d'infrastructures au coeur du réseau afin d'assurer la mobilité.

Les techniques au niveau transport semblent plus applicables. MPTCP comme QUIC ne demandent pas autant de déploiements spécifiques que les solutions au niveau réseau. Ils fonctionnent comme une couche intermédiaire entre le transport et l'application et utilisent respectivement TCP et UDP qui sont des protocoles de transport largement déployés. Ils ne nécessitent que des modifications mineures de la pile protocolaire afin de permettre aux applications de les utiliser et afin de rajouter la couche d'interface avec les protocoles de transports existants. MPTCP est utilisé par Apple pour son assistant vocal *Siri* [127, 128] ou par Korea Telecom afin de faire de l'agrégation entre le 802.11 et le cellulaire pour augmenter les débits au travers d'un proxy SOCKS [129], mais l'adoption au sein des applications reste limitée. QUIC est utilisé dans beaucoup de services de Google. Il est notamment intégré au navigateur Chrome qui l'utilise pour accéder au moteur de recherche et Youtube, et est aussi déployé dans les applications mobiles de ces services. SCTP est utilisé par *WebRTC* qui est une API javascript du W3C (Consortium du World Wide Web — *World Wide Web Consortium*) et de l'IETF (Force opérationnelle de l'ingénierie d'Internet — *Internet Engineering Task Force*). *WebRTC* est donc déployé dans la plupart des navigateurs et a pour but de faciliter le développement d'applications ayant besoin de communications en temps réel.

Enfin au niveau application, en mobilité le changement d'adresse est un problème connu et est pris en compte depuis longtemps. Des mécanisme d'authentification au niveau applicatif sont utilisés lorsque l'on a besoin d'authentifier un utilisateur sans avoir à se baser sur son adresse IP. Par ailleurs des efforts sont faits pour minimiser les temps de reconnexions et / ou réduire la quantité de données transférées. HTTP/2 [130] utilise par exemple de la compression sur l'ensemble des données et s'appuie un système de multiplexage des requêtes afin de paralléliser les flux au sein d'une même connexion TCP. Google fournit une API HTTP (Protocole de transfert de fichiers Hypertexte — *Hypertext Transfer Protocol*) faisant usage de cache afin d'éviter d'avoir à re-télécharger des données lorsque celles-ci sont suffisamment récentes : *Volley* [131].

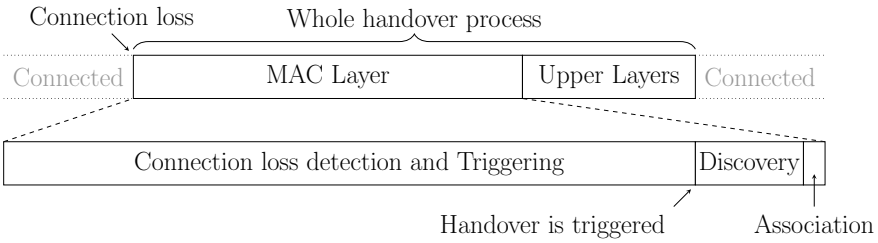


FIGURE 1.11 – Frise temporelle de la récupération de la connectivité après un *handover* 802.11

1.3 Discussion générale

1.3.1 La mobilité dans les villes

Lorsque l'on parle de mobilité à l'échelle d'une ville, nous n'avons pas les mêmes problématiques que lors de la mobilité au sein d'un bâtiment. En admettant que l'on puisse utiliser l'ensemble des *points d'accès* publics et privés disponibles dans une ville, il est possible d'obtenir une couverture radio importante mais il n'est pas possible de profiter des avantages offerts par un réseau 802.11 *managé*.

Les solutions reposant sur des déploiements spécifiques pour assurer une bonne répartition des canaux 802.11 ou ceux partageant des graphes correspondant à la topologie physique, sont difficilement applicables dans ce cas d'utilisation notamment en raison de la difficulté de construction des graphes et du passage à l'échelle.

Les *handovers* rapides proposés par 802.11R supposent que les *points d'accès* appartiennent à un unique ESS. Ils sont par ailleurs reliés à un même serveur d'autorisation qui leur permet de pré-authentifier une *station* lorsque celle-ci passe d'un *point d'accès* à un autre.

Les solutions basées sur le partage d'une unique adresse MAC pour les différents *points d'accès* passent difficilement à l'échelle. *UniFi Zero-Handoff* repose sur l'utilisation d'un unique canal mais même lorsque des solutions multicanal sont proposées, il faut la présence d'un contrôleur central qui gère les différents *points d'accès*.

Des approches basées sur l'apport de la LTE sur les bandes de fréquences utilisées en 802.11 semblent plus prometteuses car les réseaux cellulaires sont construits autour de la prise de décision du *handover* par le réseau et non par l'équipement connecté. Cependant les opérateurs de téléphonie mobile utilisent des bandes licenciées sur lesquelles ils ont un contrôle absolu. L'apport au niveau des bandes non licenciées impose une utilisation équitable avec les autres protocoles opérant sur celles-ci. Il doivent alors jouer avec les accès non garantis et assouplir leurs contraintes temporelles. De plus, la collaboration des différents protocoles sur ces bandes de fréquences n'est pas garantie.

Hotspot 2.0 est une proposition cherchant à améliorer l'intégration des opérateurs au sein des réseaux 802.11. Les accords de *roaming* et l'enrichissement des *beacons* émis par les *points d'accès* afin de faciliter la sélection et la réassociation ont pour objectif de fluidifier la connectivité entre différents *points d'accès*. Cependant *Hotspot 2.0* ne permet pas une gestion aussi fine du *handover* que celle proposée en téléphonie cellulaire.

Il serait cependant intéressant d'aborder le problème de la mobilité à l'envers. Au lieu de chercher uniquement à obtenir une durée de *handover* toujours plus courte afin de garder une connectivité quasi permanente pour pouvoir assurer tout type d'applications et de trafics lorsqu'un utilisateur se déplace, il serait judicieux d'estimer quels types de trafics et d'applications sont susceptibles de fonctionner sur un réseau 802.11 actuel dans un contexte urbain. D'autant plus que si nous nous intéressons aux smartphones, ceux-ci peuvent assurer une connectivité cellulaire pour les applications ne pouvant pas tolérer les pertes et pourraient décharger une partie de leur trafic sur le réseau 802.11 lorsque cela est possible.

1.3.2 Un handover au niveau MAC toujours important

La démocratisation des smartphones et des usages en mobilité associés ont conduit à l'amélioration de la gestion de la rupture de connectivité au niveau des applications. Celles-ci prennent de plus en plus

en compte le fait qu'un utilisateur peux changer régulièrement d'adresse IP ou que la connection peut se dégrader pendant une session active.

QUIC et MPTCP sont des propositions encore jeunes et peu démocratisées, mais permettent d'apporter plus de souplesse concernant l'utilisation de l'adresse IP comme identifiant de connexion. Comme ils se basent sur des identifiants cryptographiques, l'adresse IP est uniquement utilisée pour répondre aux problématiques de routage et de positionnement d'un équipement au sein de la topologie du réseau. Cette souplesse au niveau de la couche de transport permettrait de s'affranchir des solutions proposées au niveau de la couche réseau comme MIPv4, ces solutions étant difficilement déployables à grande échelle.

S'il est vrai que QUIC et MPTCP permettent de réduire la durée de *handover* liée à la couche transport, la durée de *handover* totale est majoritairement dûe à la difficulté d'effectuer un *handover* rapide au niveau MAC. La FIGURE 1.11 représente schématiquement le temps que met un équipement 802.11 à récupérer sa connectivité à partir du moment où il perd celle-ci. Lors du *handover* au niveau MAC, la phase la plus longue est la detection de la perte de connectivité permettant le déclenchement de la recherche et de la réassociation.

La durée de la phase de détection est accentuée, lorsque l'on parle d'équipement mobile n'ayant pas forcément de politique aggressive en terme d'évaluation de la qualité du signal, et dans un contexte de déploiement urbain non managé où les *points d'accès* ne s'échangent pas d'informations concernant la qualité de leur environnement radio entre eux, ne possèdent pas de contrôleur central et ne peuvent pas indiquer à leurs *stations* de changer de *point d'accès*— *Hotspot 2.0* tente de résoudre ces problèmes mais n'est pas encore déployé.

La phase de recherche est aussi une phase chronophage mais celle-ci peut « facilement » être diminuée en réduisant l'espace de recherche aux canaux orthogonaux au prix d'une non exhaustivité des *points d'accès* découverts. Cependant la proportion de *points d'accès* qui ne serait pas découverte resterait marginale car les canaux orthogonaux sont les canaux les plus utilisés [132].

L'association est quant à elle un processus assez rapide — l'authentification auprès d'un serveur distant pouvant cependant être assez longue.

La proportion du temps passé dans le mécanisme de détection est importante [10, 133]. Dans leurs travaux, Montavont et Noel [134] constatent que la détection de la mobilité est importante lors du processus de *handover* au niveau IP, car elle permet de déterminer quand le déclencher et influe sur les latences observées. De la même manière nous pensons qu'il faut améliorer la phase de *handover* au niveau de la couche MAC 802.11 afin d'améliorer l'expérience des utilisateurs en déplacement.

1.3.3 La récupération de données sans association

À titre d'exemple, si nous considérons des utilisateurs ayant besoin de récupérer les horaires de passage des transports en commun à proximité, actuellement ils sont émetteur de la demande de cette information auprès de la société de transport de leur agglomération. En opposition nous pourrions imaginer utiliser les *points d'accès* 802.11 présents dans l'agglomération afin de diffuser cette information dans l'air, celle-ci étant récupérée de manière opportuniste par les équipements mobiles.

Dans ce genre de scénario nous aimerions nous affranchir d'une association — et potentiellement d'une authentification — afin de pouvoir transmettre cette information. 802.11p est actuellement réservé aux trafic véhiculaire. Cependant un des mécanismes intéressants de cet amendement est la possibilité pour une *station* de communiquer avec un *point d'accès* sans avoir à effectuer les mécanismes d'authentification et d'association. Il n'est pas inimaginable que cet affranchissement soit intégré au sein des équipements grand public un jour, leur permettant de dépasser certaines limitations liées au *handover*.

Vient alors le problème de la validité spatio-temporelle d'une donnée. Pour reprendre l'exemple des transports en commun, il est souhaitable que les *points d'accès* ne diffusent que les informations de trafic concernant les arrêts de bus ou stations de métro avoisinant. De la même manière, il faut que ceux-ci ne diffusent que des informations concernant les prochains passages et non ceux déjà passés ou arrivant dans un temps trop lointain — pour lequel il pourrait y avoir des réajustements.

En dehors d'un accès sans association au canal, il faut aussi s'intéresser aux mécanismes permettant d'envoyer des informations dans une zone géographique précise : capacité du réseau, routage géographique et pertinence des données à diffuser.

Deuxième partie

Contributions

Sommaire

2.1 Bancs d'essai pour des expériences dans les réseaux sans fil 65
2.1.1 Expérimentations dans les réseaux sans fil 65
2.1.2 Répétabilité et reproductibilité 66
2.2 Architecture de WalT 66
2.2.1 Vue d'ensemble 66
2.2.2 Images WalT 67
2.2.3 Nœuds WalT 68
2.2.4 WalT Client 69
2.2.5 Expériences WalT 70
2.2.6 Serveur WalT 70
2.2.7 Plateformes WalT au sein du laboratoire LIG 70
2.3 Exemples d'expériences avec WalT 71
2.3.1 Comparaison de la synchronisation temporelle NTP/PTP 71
2.3.2 Mesure de la durée de *handover* en 802.11 72
2.4 Reproductibilité des expériences en fonction du temps : Transformer la plate-forme WalT en machine à remonter le temps 73
2.4.1 Expériences reproductibles indépendamment du temps 73
2.4.1.1 Paramètres de l'expérience 74
2.4.1.2 Description de la plate-forme 74
2.4.1.3 Discussion 74
2.4.2 Application de ces mécanismes dans WalT 74
2.4.2.1 Versionnement des composants de la plate-forme 75
2.4.2.2 Mise à niveau et rétrogradation des versions 75
2.4.2.3 Gestion des expériences WalT 76
2.4.2.4 Discussion 76

Introduction

La validation de nouveaux protocoles pour les réseaux sans fil est une tâche difficile pour laquelle nous pouvons principalement utiliser des simulations ou procéder à des expérimentations. Les simulations sont assez limitées et ne peuvent nous donner qu'un premier aperçu du comportement d'un protocole choisi sous des hypothèses simplifiées [135, 136]. Habituellement, les simulations ne tiennent pas suffisamment compte de l'environnement de propagation réel — bilans de liaison peu variables, prise en compte des trajets multiples, implémentations sur du matériel réel, etc. —, donnant des résultats intéressants au premier abord, mais pouvant s'avérer différents d'un déploiement réel.

La réalisation d'expériences réelles est considérée comme la condition préalable à une véritable validation d'une proposition de protocole et exige à la fois répétabilité et reproductibilité [137]. La répétition exige que la même expérience soit menée plusieurs fois dans les mêmes conditions par l'investigateur, produisant des résultats similaires, et il est nécessaire de garantir que les résultats expérimentaux sont bons. La reproductibilité est garantie si une expérience peut être reproduite dans des conditions différentes, tout en fournissant des résultats suffisamment similaires. Ceci est essentiel pour prouver que la proposition scientifique est robuste dans divers environnements et pas seulement dans un contexte particulier. Il est également crucial, pour permettre aux chercheurs de reproduire des expériences, de s'appuyer sur leurs résultats et de les comparer avec les travaux antérieurs.

Les expériences impliquant des réseaux sans fil sont difficiles à mener. Nous avons le choix entre l'utilisation de plates-formes spécialisées (comme ORBIT [138], w-iLab.t [139], IoT-LAB [140], et autres) ou la mise en place de bancs de test *ad-hoc*. Les plates-formes spécialisées facilitent la tâche d'expérimentation et offrent un bon support pour la répétabilité de l'expérience. Cependant, elles n'offrent que des conditions d'exploitation qui non sont pas toujours similaires à celles d'un déploiement réel [141] : les nœuds peuvent être répartis sur une grille régulière dans une grande pièce, et la reproductibilité dans des conditions environnementales variables est très limitée. Comme la performance des réseaux sans fil dépend fortement de leur déploiement, de leur environnement et de leurs conditions d'exploitation, la validation des protocoles dans des conditions artificielles peut ne pas aboutir à des résultats significatifs, et la reproductibilité est de facto limitée. De plus, les chercheurs n'ont pas d'accès physique à l'équipement, de sorte que le débogage est fastidieux et que certaines mesures, comme la consommation d'énergie à grains fins, peuvent être impossibles. Les bancs d'essai *ad-hoc* peuvent aider à valider un protocole donné, mais ils n'évoluent pas et il est difficile de reproduire les résultats car la plupart du temps, il manque une spécification précise.

Ce chapitre présente *WalT*, une plate-forme reproductible pour réaliser des expériences reproductibles. L'objectif est triple : 1. mettre en place facilement une plate-forme pour développer, déboguer et effectuer une validation préliminaire, 2. permettre le déploiement et le contrôle d'expériences à plus grande échelle, 3. permettre à d'autres de déployer exactement la même expérience dans un environnement différent, remettre en question la reproductibilité et garantir également un point de départ approprié et une répétabilité adéquate lors de la comparaison avec d'autres propositions. Elle suit une approche qui se situe quelque part entre des plates-formes spécialisées, à topologie physique rigide et limitée, et des bancs d'essai *ad-hoc*. Les nœuds *WalT* sont des SBC (Ordinateur mono-carte utilisant un unique circuit intégré — *Single-Board Computer*) sur lesquels les utilisateurs peuvent déployer leur système d'exploitation (système de fichiers, noyau) sous la forme d'une image *Docker* pour une personnalisation et un partage faciles.

Une plate-forme *WalT* fournit :

- un contrôle total à distance des nœuds : redémarrage, sessions `shell` distantes, déploiement d'images d'OS (Système d'exploitation — *Operating System*),
- une gestion des images des nœuds de l'OS : clonage à partir du *Docker Hub*, modification locale et publication des images,
- une gestion des *logs* : des moyens de collecter, stocker et interroger les *logs* d'expérience et les marqueurs d'événements sont fournis,
- une découverte automatisée de la topologie logique de la plate-forme.

WalT présente les avantages suivants :

- il peut être facilement reproduit sur les lieux de déploiement requis,
- il est économique, construit à partir de composants peu coûteux utilisant des logiciels libres [142],

- il est léger, il peut servir pour une démonstration portable ou en tant que plate-forme de développement dans un bureau sans avoir les difficultés liées à un déploiement dans un bâtiment entier,
- il est adaptable, il est possible de l'utiliser pour mener des expériences sur des réseaux de capteurs sans fil aussi bien que des expériences sur 802.11,
- il est facile à installer et à utiliser.

En plus de soutenir les expériences reproductibles, *WalT* permet également l'émergence de plates-formes reproductibles — les chercheurs peuvent mettre en place leurs plates-formes *WalT* pour valider les résultats d'autres chercheurs à différents endroits et dans d'autres environnements.

La plate-forme est actuellement développée au sein du Laboratoire d'Informatique de Grenoble, et est activement maintenue par Étienne Dublé. Ma contribution dans ces travaux est constituée de :

- discussions concernant l'architecture de *WalT*,
- validation de la plate-forme pendant les différentes phases de développement — j'ai été l'utilisateur quasi exclusif de la plateforme au début du développement — et,
- retours concernant l'utilisation de la plate-forme afin d'améliorer l'expérience utilisateur.

La plate-forme est déployée dans le laboratoire et est composée d'un peu moins d'une centaine de nœuds répartis en îlots de moins de 10 nœuds dans différents bureaux ainsi que sur un banc d'essai d'une trentaine de nœuds.

2.1 Bancs d'essai pour des expériences dans les réseaux sans fil

Le support sans fil présente des caractéristiques complexes car il est sensible à de nombreux facteurs externes et aux conditions d'environnement difficiles à modéliser. Ainsi, l'ensemble de la pile de protocoles doit être robuste face à ces difficultés et doit être évalué dans des situations réalistes qui impliquent tous les facteurs qui peuvent influencer la robustesse. De plus, la recherche sur les réseaux sans fil requiert un vaste champ d'expertise puisque de nombreux aspects de la pile de protocoles jusqu'aux applications reposent sur des fonctionnalités spécifiques : la couche physique, le contrôle d'accès au medium, la maintenance et la découverte des voisins, le routage, pour n'en nommer que quelques uns. Les récentes activités de recherche sur les réseaux de capteurs sans fil ont redéfini tout un ensemble de normes pour répondre à des contraintes encore plus strictes dans le contexte de l'IoT : faible consommation, ressources contraintes (énergie, puissance de traitement et mémoire) et environnements à fort taux de pertes. IEEE 802.15.4 pour les couches physiques et de liaison, IETF 6LoWPAN pour la couche réseau, RPL pour le routage et CoAP pour la couche application sont quelques-uns des standards résultants pour lesquels de nombreux aspects restent à étudier.

2.1.1 Expérimentations dans les réseaux sans fil

Il existe deux principaux moyens de valider les propositions concernant les réseaux et leurs protocoles : la simulation et les expérimentations. Pour des raisons évidentes, la simulation, avec ou sans émulation, a été l'un des moyens privilégiés pour valider des protocoles à grande échelle car elle propose quelques avantages : développement facile, coût, répétabilité, environnements contrôlés. Dans ce cas, la validité des résultats dépend beaucoup de la précision du modèle du système entier à étudier. C'est là que la simulation trouve son point de rupture : la propagation électromagnétique dans un environnement réel est presque impossible à simuler avec précision car les modèles sont complexes et demandent beaucoup de calculs, et il existe une grande variété d'environnements à prendre en compte selon les scénarios de déploiement prévus.

De plus, lors de l'étude des protocoles sans fil, la mise en œuvre soulève de nombreuses questions : beaucoup de travail doit être fait au niveau des couches inférieures, et parfois le débogage et la validation des mises en œuvre nécessitent l'utilisation d'oscilloscopes et d'analyseurs numériques. C'est notamment le cas pour les études relatives à l'IoT dans lesquelles les solutions proposées doivent être mises en œuvre

sur des dispositifs très contraints [143] et les protocoles doivent être optimisés pour être économes en énergie avec des contraintes de synchronisation strictes de l'ordre de la μs par exemple.

Il y a donc un besoin d'expérimentations en conditions réelles avec des bancs d'essai allant d'une petite taille dans la phase de développement à des déploiements plus importants pour des expériences dans des conditions proches de la situation réelle d'utilisation.

2.1.2 Répétabilité et reproductibilité

Dans ce contexte, la répétabilité et la reproductibilité des expériences sont deux défis cruciaux. Tout d'abord, pour déboguer efficacement et valider les implémentations, il est primordial de pouvoir exécuter les mêmes suites de tests dans exactement les mêmes conditions. Deuxièmement, pour être valables pour les réseaux sans fil, les propositions scientifiques doivent démontrer leur robustesse dans différentes conditions d'exploitation que l'on retrouve dans les déploiements réels. Les bancs d'essai partagés à grande échelle sont d'excellents outils pour expérimenter de nouveaux protocoles et algorithmes, mais ils ont certaines limites : les déploiements artificiels utilisant des dispositions régulières avec une forte densité de nœuds, les conditions stables dans lesquelles ils fonctionnent et le choix limité des dispositifs. En plus de ces infrastructures, nous avons besoin d'un moyen de déployer des bancs d'essai en conditions réelles et de reproduire ces bancs d'essai à différents endroits pour montrer la reproductibilité des résultats.

Nous soutenons que la validation des expériences portant sur les réseaux sans fil ne repose pas seulement sur la démonstration que la même expérience répétée dans exactement la même situation donne toujours les mêmes résultats. Nous devons également montrer que les nouveaux protocoles et mécanismes donnent de bons résultats dans un large éventail de conditions réelles, ce qui signifie que nous devons être en mesure de déployer et d'exécuter facilement des expériences dans une grande variété de situations. C'est pour cela que *WalT* a été conçu, car une fois qu'un nouveau protocole a été développé sur un déploiement *WalT* donné, il peut être exécuté facilement sur toute autre plate-forme *WalT*. De cette manière, comme chaque plate-forme *WalT* est déployée dans un environnement réel avec une topologie de nœuds unique, lorsqu'une expérience donnée est reproduite et que les résultats sont similaires, nous pouvons conclure sur la validité de ces derniers.

Enfin, nous devons également garantir que nos collègues chercheurs seront en mesure de mener des expériences et de comparer leurs résultats avec les résultats des nouvelles propositions dans les mêmes conditions pour prouver leurs améliorations par la répétabilité de la comparaison. Ceci n'est possible que si tous les autres paramètres sauf l'élément testé sont similaires.

WalT vise à faciliter ce processus en s'appuyant sur deux caractéristiques principales : 1. il s'appuie sur du matériel standard à faible coût, 2. il utilise *Docker* pour construire et exécuter des expériences permettant de distribuer un *snapshot* de l'ensemble de l'environnement logiciel utilisé pendant l'expérience. Ces deux caractéristiques offrent les conditions de répétabilité, puisque le même logiciel sera exécuté pour chaque exécution dans exactement les mêmes conditions, les mêmes images étant redémarrées à partir de zéro dans un environnement réel, et de reproductibilité, puisque la même plate-forme peut être déployée dans tout autre endroit, exécutant la même expérience.

2.2 Architecture de WalT

2.2.1 Vue d'ensemble

La FIGURE 2.1 présente la structure fonctionnelle d'une plate-forme *WalT*. Du point de vue de l'utilisateur, *WalT* se compose d'un ensemble de nœuds gérés à distance. Il s'agit de SBC à faible coût, tels que les *Raspberry Pi*, sur lesquels les utilisateurs peuvent déployer un système d'exploitation donné.

Le cerveau de *WalT* est un serveur basé sur GNU / LINUX qui connecte les autres composants de la plate-forme, interagit avec le logiciel côté client et assure la gestion des *logs* (collecte, stockage, interrogation). Le logiciel côté client, qui délègue toutes les tâches complexes au serveur, consiste en un outil léger et portable, utilisé pour explorer la topologie logique, interagir avec les nœuds, déployer une image donnée de l'OS avec son environnement logiciel sur ces nœuds, personnaliser une telle image et explorer les *logs*.

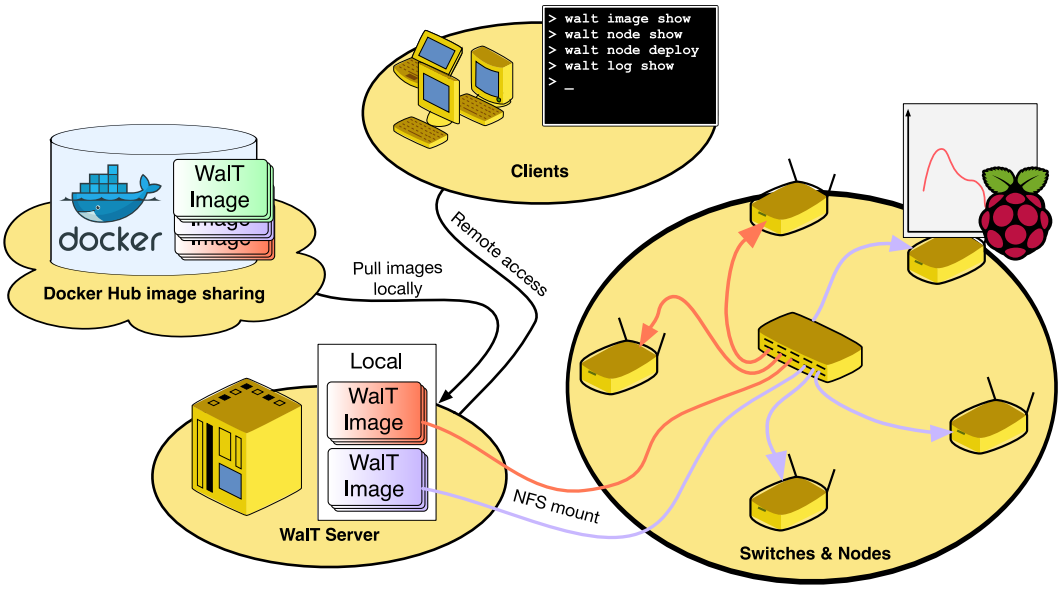


FIGURE 2.1 – Architecture fonctionnelle de *WalT*

Le réseau est composé de commutateurs économiques offrant un petit nombre de fonctions, notamment la gestion à distance et l'alimentation par POE (Alimentation électrique par câble Ethernet — *Power over Ethernet*). POE simplifie le déploiement en fournissant l'alimentation et la communication de données jusqu'à chaque nœud en utilisant le même câble Ethernet, ce qui permet de réutiliser tout câblage existant lors d'un déploiement dans un bâtiment par exemple.

WalT s'appuie en interne sur la plate-forme de virtualisation de *Docker*. *Docker* offre un mécanisme d'« empaquetage » efficace ainsi qu'une distribution facile des systèmes d'exploitation et environnements logiciels déployés sur les nœuds.

Le but final de la conception de *WalT* était la reproductibilité de la plate-forme — au lieu de faire des expériences sur un banc d'essai partagé publiquement, *WalT* propose de reproduire la plate-forme pour tout besoin d'expérimentation. *WalT* atteint la reproductibilité de la plate-forme parce qu'elle est composée de matériels standards à faible coût, de logiciels libres et qu'elle est facilement déployable grâce au POE. Le logiciel s'exécutant sur les nœuds est intégré dans une image *Docker* permettant à d'autres chercheurs de le déployer instantanément, de réexécuter des images d'autres expériences dans des conditions différentes ou d'effectuer de nouvelles expériences. Pour conclure, la communauté réseaux peut facilement reproduire la plate-forme *WalT* ainsi que les expériences *WalT*.

Le reste de cette section fournira plus de détails sur les composants clés de *WalT* et décrira nos utilisations du *WalT* au LIG.

2.2.2 Images WalT

WalT introduit le concept d'image *WalT* — celle-ci contient un système d'exploitation prêt à être déployé sur un nœud *WalT*. Le système d'exploitation est composé d'un système de fichiers et d'un noyau. Une image *WalT* est empaquetée en interne sous forme d'une image *Docker*, ce qui présente deux avantages immédiats. Tout d'abord, *WalT* réutilise les fonctions de gestion des images de *Docker*. Par exemple, la commande client `walt image shell <nom-image>` fonctionne en exécutant un `shell` dans un conteneur *Docker*. Ceci permet de modifier l'image d'une manière très pratique. Adapter une image à une expérience donnée — installer plus de logiciels ou de scripts d'expérimentation par exemple — ou à une configuration de plate-forme donnée — installer un pilote pour un périphérique WI-FI connecté au nœud par exemple — ne demande que quelques commandes. Le deuxième avantage est la possibilité pour *WalT* d'utiliser le *Docker Hub*, le dépôt public d'images *Docker*. Comme une image *WalT* est en fait une image *Docker*, les images *WalT* peuvent être publiées et partagées sur le *Docker Hub*. Le client *WalT* permet de rechercher les images publiées et de les cloner localement mais aussi de publier des images sur le *Docker Hub*.

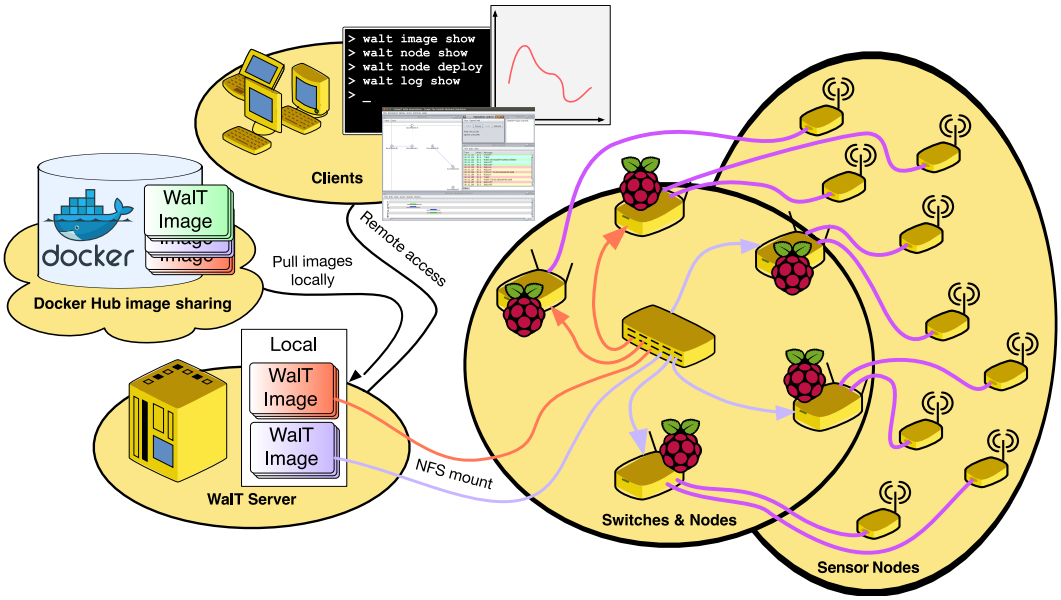


FIGURE 2.2 – Architecture spécialisée pour les réseaux de capteurs sans fil

Une fonctionnalité avancée du serveur *WalT* permet de monter une image *Docker* et de la rendre disponible en partage NFS (Système de fichiers sur le réseau — *Network File System*), ce qui permet aux nœuds *WalT* de démarrer le système contenu dans une image *WalT*. Pouvoir transformer instantanément un système d'exploitation virtualisé en un déploiement réel est l'une des caractéristiques les plus remarquables de *WalT*.

Les images *WalT* contiennent une fine couche middleware fournissant des outils pour générer des *logs* d'expériences. Les *logs* sont envoyés au serveur pour stockage et traitement.

2.2.3 Nœuds WalT

Les nœuds *WalT* sont des SBC économiques alimentés par POE. Actuellement, *WalT* supporte les cartes *Raspberry Pi B, B+, 2B, 3B* et *3B+*. Ces cartes ne supportant pas le POE, nous utilisons un séparateur POE externe.

La robustesse des SBC peu chers est un problème. Notre expérience a montré que leur principal point d'échec sont les cartes SD — celles-ci supportent mal les écritures répétées aboutissant à des corruptions de secteurs. Ce problème est résolu en gardant les cartes SD en lecture seule. Nous n'avons pas observé une seule défaillance de nœuds en plus de quatre ans d'utilisation depuis ce changement.

La procédure de démarrage des nœuds *WalT* est un processus en deux étapes. Un nœud démarre tout d'abord un *boot loader* réseau stocké en lecture seule sur la carte SD. Celui-ci sera ensuite chargé de démarrer le système d'exploitation stocké dans l'image *WalT* voulue, via NFS. Actuellement le *boot loader* utilisé est U-BOOT [144].

Toutes les modifications du système de fichiers sont stockées dans la RAM (Mémoire vive — *Random-Access Memory*) du nœud (donc supprimées lors du redémarrage du nœud). Ceci garantit qu'un nœud *WalT* associé à une image *WalT* donnée démarrera toujours exactement le même système d'exploitation, tout en évitant d'écrire sur les cartes SD.

Avoir un tel niveau de contrôle sur le système d'exploitation fonctionnant sur les nœuds rend la plateforme très polyvalente. Par exemple, pour réaliser des expériences sur les protocoles de routage dans les réseaux de capteurs sans fil, nous avons spécialisé l'architecture *WalT* pour obtenir celle illustrée à la FIGURE 2.2.

Dans cette configuration, chaque nœud *WalT* connecte et surveille un ou plusieurs capteur(s) sans fil. L'image *WalT* sélectionnée contient des outils pour flasher le ou les capteur(s) et surveiller les traces du ou des capteur(s) sur la ou les liaison(s) série USB.

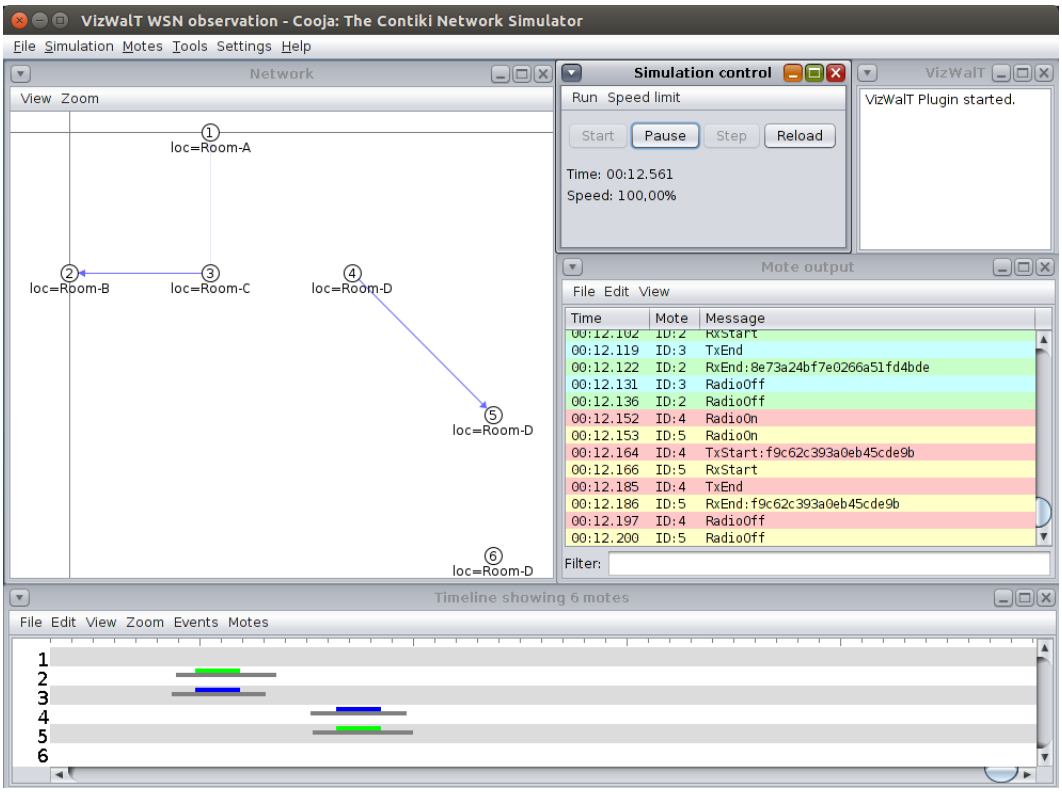


FIGURE 2.3 – Capture d'écran de Cooja représentant les échanges sur *WalT*

2.2.4 WalT Client

Les utilisateurs interagissent avec la plate-forme en utilisant un outil en ligne de commande appelé `walt`. Il fournit cinq catégories de commandes : `image`, `node`, `log`, `device` et `advanced`.

La catégorie `image` fournit des commandes pour lister les images *WalT*, les dupliquer, les copier, les renommer, les supprimer ou les modifier (en utilisant un `shell` virtualisé), transférer des fichiers entre la machine client et les images *WalT*, rechercher des images sur le *Docker Hub*, et cloner une image du *Docker Hub* vers la plate-forme locale.

La catégorie `node` permet d'exécuter une commande ou un `shell` sur un nœud, de lister les nœuds, de transférer des fichiers entre la machine cliente et les nœuds *WalT*, de déployer une image sur un ensemble de nœuds et d'effectuer des tâches de débogage telles que redémarrer un nœud, faire clignoter une led pour une identification physique du nœud, etc.

La catégorie `log` permet à l'utilisateur d'explorer les *logs* d'expériences. Ils sont sauvegardés dans une base de données sur le serveur. L'utilisateur peut interroger l'historique des *logs* à tout moment. Une autre option permet à l'utilisateur de visualiser les logs en pseudo temps réel.

Les catégories `device` et `advanced` sont orientées vers la gestion, la personnalisation et la maintenance de la plate-forme. Par exemple, on peut vouloir renommer des nœuds ou des commutateurs réseau d'une manière plus explicite — avec la position des nœuds par exemple.

Malgré ce contrôle étendu de la plate-forme offert à l'utilisateur, l'outil en ligne de commande reste léger et facile à installer, car le serveur gère la complexité. L'outil est publié sur PYPI (Gestionnaire de paquets *Python — Python Package Index*) et peut être installé en une seule commande en utilisant l'outil de gestion des paquets `pip`.

Un outil visuel pour les réseaux de capteurs sans fil appelé *VizWalT* a également été développé et est implémenté sous la forme de plugin Cooja. Cooja est un simulateur de réseaux de capteurs sans fil : il permet d'exécuter une simulation en émulant chaque nœud dans un réseau de capteurs sans fil. Lorsqu'il est chargé avec le plugin *VizWalT*, ce comportement est modifié : chaque nœud vu sur l'interface utilisateur reflète le comportement d'un nœud réel déployé dans le banc de test *WalT*. La FIGURE 2.3 est une capture d'écran de Cooja présentant les traces de *WalT*.

2.2.5 Expériences WalT

L'un des objectifs de conception de l'outil `walt` était de supporter le développement de scripts expérimentaux avec n'importe quel langage de programmation auquel l'utilisateur est habitué. Le script peut gérer à la fois le déploiement et le contrôle d'une expérience. Puisque la complexité est gérée dans le *middleware WalT*, l'écriture d'un tel script est très simple. Le partage d'un script d'expérience avec une description de la configuration physique (topologie physique du réseau, dispositifs WI-FI connectés aux nœuds, etc.) rend l'expérience reproductible, que ce soit dans un environnement physique similaire ou différent.

Nous travaillons toujours sur l'amélioration de l'outil Walt avec de nouvelles options pour faciliter la gestion des expériences *WalT*. À l'avenir, l'outil intégrera des options pour enregistrer, publier et rejouer les expériences, et pour publier ou télécharger les résultats des expériences.

2.2.6 Serveur WalT

Le serveur *WalT* contrôle la plate-forme, interagit avec le *Docker Hub*, les nœuds et les clients. L'une des principales fonctions du serveur est la gestion des *logs*. Le serveur reçoit les *logs* provenant des nœuds, les enregistre dans une base de données locale et les transmet éventuellement aux clients pour visualisation en temps réel. Lorsque l'utilisateur demande l'historique des *logs*, le serveur interroge simplement la base de données.

Le serveur gère également la communication avec le *backend* de *Docker* et monte ou démonte les images *WalT* en tant que partages NFS en fonction des besoins du client — typiquement les images que celui-ci a déployées sur ses nœuds.

Le serveur gère les données topologiques logiques de la plate-forme en interrogeant les commutateurs à l'aide du protocole SNMP (Protocole simple de gestion de réseau — *Simple Network Management Protocol*). Les commutateurs utilisent le protocole LLDP (Protocole de découverte sur la couche de liaison — *Link Layer Discovery Protocol*) pour découvrir leurs voisins immédiats.

Le réseau de la plate-forme et le réseau externe (permettant d'atteindre le *Docker Hub*) sont isolés sur des VLAN (Réseau local virtuel — *Virtual Local Area Network*) dédiés pour éviter toute perturbation du réseau pendant les expériences. Cette configuration réseau est entièrement automatisée.

L'installation d'un serveur *WalT* est facile et rapide grâce à l'utilisation d'une image amorçable créée avec debootstick [145].

2.2.7 Plateformes WalT au sein du laboratoire LIG

Bien que *WalT* soit encore dans une phase de développement, nous avons commencé à travailler avec les premières versions il y a plus de quatre ans. En conséquence, notre principal déploiement dans le bâtiment du laboratoire a déjà été utilisé pour plusieurs projets de recherche. Comme *WalT* est également bien adapté aux bancs d'essai de petite taille, nous avons mis en place une plate-forme de démonstration mobile basée sur *WalT* avec un petit nombre de nœuds, un mini-PC Intel NUC comme serveur, et une douzaine de capteurs sans fil *STMicroelectronics* 802.15.4. La démo mobile a montré la formation d'un réseau sans fil multi-saut sur Cooja via l'interface *VizWalT* lors de la revue finale du CALIPSO3, et d'une démonstration lors d'un *workshop* [146]. Cette plateforme mobile nous a permis par ailleurs d'observer qu'une expérience développée — par d'autres membres de l'équipe travaillant sur le 802.15.4 — dans nos locaux pouvait être reproduite dans un environnement différent, avec ses propres caractéristiques radio et une autre disposition des nœuds. Nous avons également utilisé *WalT* pour déboguer des protocoles pour l'IoT avec des installations temporaires sur le bureau ou simplement en étendant la plate-forme principale dans un bureau donné (débrancher un des nœuds déployés dans le bureau, le remplacer par un commutateur, y connecter quelques nœuds et exécuter `walt device rescan` pour la mise à jour de la topologie logique).

Les utilisateurs de *WalT* apprécient particulièrement de pouvoir utiliser la même plate-forme pour le débogage, les expériences et les démonstrations.

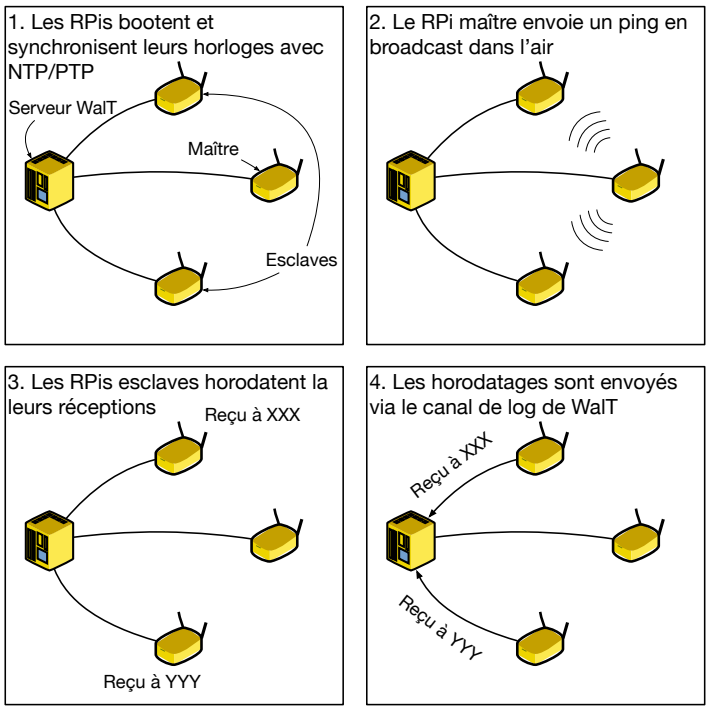
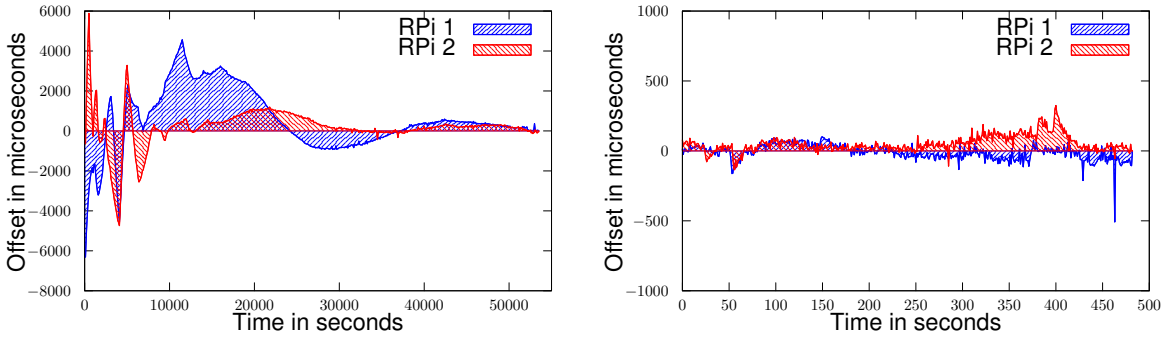


FIGURE 2.4 – Scénario de la mesure de dérive d'horloge



(a) NTP

(b) PTP

FIGURE 2.5 – Variation du décalage d'horloge entre deux nœuds et un référentiel temporel

2.3 Exemples d'expériences avec WalT

2.3.1 Comparaison de la synchronisation temporelle NTP/PTP

Synchroniser tous les nœuds *WalT* est très difficile car les *Raspberry Pi* n'ont pas d'horloge dédiée et chaque fois qu'un nœud redémarre, il perd la synchronisation. Nous avons décidé d'utiliser NTP et PTP pour la synchronisation du temps. L'évaluation de la dérive de l'horloge entre les nœuds était un bon cas de test pour montrer comment *WalT* peut être utilisé.

La FIGURE 2.4 illustre le scénario : nous utilisons trois nœuds avec dongles WI-FI. Nous avons construit des images *WalT* pour qu'elles soient connectées au même réseau WI-FI *ad-hoc*. Un nœud agit comme un maître et diffuse un ping à tous les esclaves. Ils reçoivent le message presque en même temps — durée de propagation et temps de traitement — qu'ils horodatent localement. Enfin, les horodatages de réception sont enregistrés sur le serveur afin que nous puissions calculer le décalage d'horloge.

WalT nous permet de répéter l'expérience sur une longue durée avec trois nœuds esclaves. Les résultats soulignent que NTP offre un décalage d'horloge de l'ordre du dixième de millisecondes juste après le démarrage, et qu'une synchronisation plus précise peut nécessiter un temps très long, comme l'illustre la FIGURE 2.5a - jusqu'à $13h$ pour réaliser une synchronisation de l'ordre de la μs. La FIGURE 2.5b montre une

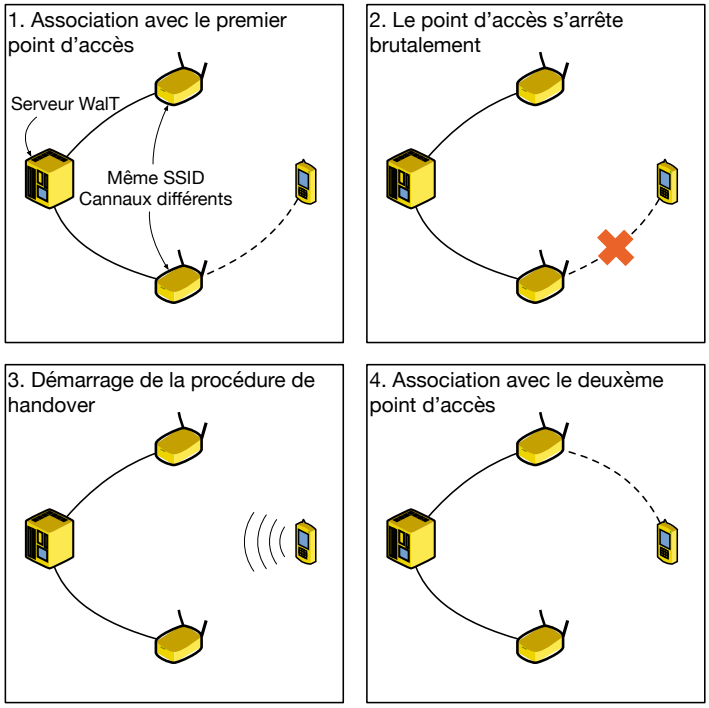


FIGURE 2.6 – Scénario de la mesure de *handover*

convergence beaucoup plus rapide pour la même expérience avec l'utilisation de PTP.

2.3.2 Mesure de la durée de *handover* en 802.11

Un autre exemple d'expérience a été mené avec *WalT* pour mesurer la durée du *handover* en IEEE 802.11 avec des appareils connectés récents. La FIGURE 2.6 présente le scénario impliquant deux nœuds *Raspberry Pi* modèle B équipés de *dongles* WI-FI. Nous avons construit une image *WalT* pour exécuter ces nœuds en mode *points d'accès*. Dans le scénario, le premier nœud démarre un *point d'accès* et attend qu'un périphérique se connecte. Nous considérons le périphérique connecté une fois qu'il est associé au *point d'accès* et qu'il peut échanger des paquets IP. Pour vérifier la connectivité IP, l'appareil envoie périodiquement des paquets UDP. Une fois la connexion établie, le premier nœud notifie également au second de démarrer son *point d'accès*. Dès que le deuxième *point d'accès* est opérationnel, nous arrêtons soudainement le premier, de sorte que l'appareil connecté ne reçoive aucune notification de la défaillance de son ancien *point d'accès*— pas de messages de disassociation ni de dé-authentification. Ensuite, nous mesurons le temps qui s'écoule jusqu'à ce que l'appareil rétablisse la connectivité IP par l'intermédiaire du deuxième *point d'accès*.

Grâce à *WalT*, nous pouvons automatiser l'ensemble du processus et répéter sans difficulté l'expérience un grand nombre de fois pour différents appareils. Ici, nous avons rassemblé 300 mesures pour chaque appareil, une tablette *Nexus 7 (2012)* (N7) utilisant *Android 4.4* et un *Moto X 2nd Gen* (MX) sous *Android 5.1*. La FIGURE 2.7 montre les durées de *handover* WI-FI mesurées : le temps d'association avec l'autre *point d'accès* (A) et le temps de reprise du trafic UDP (UDP), en utilisant soit une adresse IP fixe soit une adresse DHCP lorsque ce n'est pas précisé. Les résultats montrent des délais relativement longs pour *Nexus 7* et des variations importantes pour *Moto X*. Le *Moto X* obtient de meilleurs résultats que la *Nexus 7*, ce qui suggère que ce dernier implémente une ou plusieurs améliorations proposées au chapitre précédent. Une étude plus approfondie des implémentation de ces deux équipements mobiles permettrait de confirmer ces hypothèses. Néanmoins la présente étude démontre l'intérêt de la plateforme *WalT*.

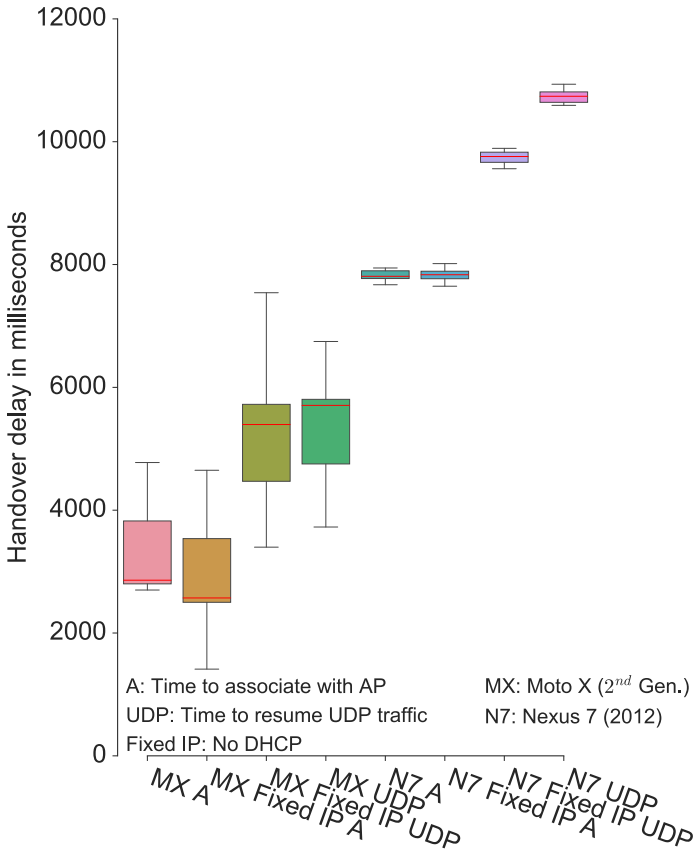


FIGURE 2.7 – Durées de *handover* mesurées

2.4 Reproductibilité des expériences en fonction du temps : Transformer la plate-forme WalT en machine à remonter le temps

WalT fournit déjà un moyen d'exécuter un *snapshot* d'expérience à un instant donné, le *snapshot* contenant l'environnement logiciel complet et les scripts requis pour l'expérience. L'objectif serait de renforcer le soutien à la reproductibilité et à la répétabilité en tenant compte de l'impact du temps. Tant les expériences que les bancs d'essai reposent sur du matériel et des logiciels qui peuvent changer au fil du temps — il est presque nécessaire de mettre à niveau les environnements d'expérience pour bénéficier des nouveaux développements ou des nouvelles fonctionnalités. Cependant, même une légère modification peut avoir un impact énorme sur les expériences et leurs résultats [147]. Comme la reproductibilité vise à évaluer les performances de l'expérience dans différentes topologies de déploiement ou différents environnements radio, il est essentiel de garantir que seules les conditions changent lorsque nous réexécutons une expérience et que la différence dans les résultats de performance ne provient pas de la modification du logiciel du banc de test. Par conséquent, l'environnement idéal du banc d'essai devrait fournir un moyen d'assurer non seulement la rétrocompatibilité, mais aussi un mécanisme permettant de recréer l'environnement logiciel et matériel exact d'une expérience donnée.

Cette partie se concentre sur les mécanismes envisagés au sein de la plate-forme *WalT* afin d'assurer la reproductibilité dans le temps. Nous abordons les enjeux et les défis liés à la reproductibilité sur le long terme ainsi que les solutions envisagées pour la plate-forme.

2.4.1 Expériences reproductibles indépendamment du temps

Comme nous l'avons vu précédemment, la reproductibilité des expériences consiste à relancer une expérience dans les mêmes conditions matérielles et logicielles en ne modifiant que certains autres facteurs

tels que la topologie physique du réseau. Cependant, il est fort probable qu'une expérience préparée il y a des années sur une plate-forme donnée ne puisse plus être exécutée, en raison de l'évolution de la plate-forme, à moins que celle-ci ne mette en œuvre ce que nous appelons la reproductibilité de l'expérience indépendamment du temps. Si jamais cette plate-forme peut encore faire fonctionner l'ancienne expérience, nous ne pouvons pas garantir que les résultats ne seront pas affectés par les changements apportés à celle-ci, tels que de nouveaux bogues, des correctifs sur d'anciens bogues ou l'ajout de nouvelles fonctionnalités.

Assurer la reproductibilité des expériences indépendamment du temps n'est pas aussi simple qu'il n'y paraît, car de nombreux paramètres peuvent avoir un impact sur les résultats d'expériences et sont généralement : 1. difficiles à décrire et/ou 2. en constante évolution. Nous soulignons ci-dessous quelques paramètres clés qui concernent les expériences ou la plate-forme sur laquelle elles sont menées.

2.4.1.1 Paramètres de l'expérience

Comme nous l'avons vu à travers l'utilisation de la plate-forme, pour pouvoir reproduire correctement une expérience en réseau nécessite une connaissance précise des différents paramètres de l'expérience :

- Il faut savoir quels nœuds ont été utilisés pour l'expérience, leur nombre, leur modèle et leur numéro de révision. Les changements de matériel peuvent avoir un impact majeur sur les performances, en particulier dans le cas des réseaux sans fil.
- Il faut savoir précisément quels logiciels les nœuds utilisent, y compris tous les fichiers du système d'exploitation, tous les programmes ou scripts d'expérience ajoutés, et tous les fichiers de configuration modifiés. Par exemple, si une expérience a été faite en utilisant une taille de tampon de socket accrue dans `sysctl.conf`, les résultats peuvent ne pas être les mêmes avec et sans ce paramètre.
- Il faut obtenir n'importe quel script global nécessaire à l'orchestration de l'expérience.

2.4.1.2 Description de la plate-forme

Bien que les paramètres de l'expériences soient déterminant dans les résultats de celle-ci, la plate-forme elle-même ne doit pas être oubliée. Comme l'ont montré Mesnard et Barba [147], quand on veut reproduire une expérience, les paramètres de la plate-forme ont un impact sur les résultats obtenus, par exemple la version d'une librairie utilisée. Dans le cas de *WalT*, ces paramètres sont :

- Le serveur qui pilote la plate-forme doit exécuter la même pile logicielle. Si cette condition n'est pas remplie, de nombreux facteurs peuvent entraîner des divergences dans les résultats des expériences. Un bogue dans la pile réseau d'une version donnée du noyau peut modifier l'horodatage d'un *log*. Un fichier de configuration PTP différent peut modifier le délai avant la synchronisation des nœuds au démarrage.
- Idéalement, le matériel du serveur et des commutateurs réseau devrait également être le même. Cependant, on peut considérer que le matériel du serveur a rarement un impact sur l'expérience (sauf dans le cas d'un problème majeur de performance dû à l'ancien matériel).

2.4.1.3 Discussion

Les paramètres évoqués ci-dessus montrent que pour pouvoir reproduire une expérience, une connaissance approfondie de l'expérience et de la plate-forme est nécessaire. Lorsque l'on utilise *WalT*, la plate-forme elle-même enregistre la plupart de ces informations dans l'image *Docker*. De plus, lorsqu'un utilisateur souhaite reproduire une expérience, la plate-forme pourra restaurer l'ensemble de sa pile logicielle dans l'état où elle se trouvait lors de la première exécution de l'expérience.

2.4.2 Application de ces mécanismes dans WalT

Dans cette section, nous présentons les mécanismes envisagés dans *WalT* afin d'assurer la reproductibilité des expériences indépendamment du temps.

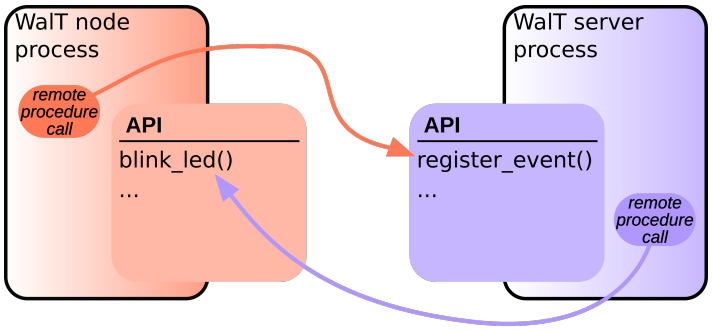


FIGURE 2.8 – API du serveur *WalT* et de ses nœuds

2.4.2.1 Versionnement des composants de la plate-forme

Les composants logiciels de la plate-forme *WalT* (nœuds, serveur et outils clients) évoluent au fur et à mesure que le projet met en œuvre de nouvelles fonctionnalités. Comme nous pouvons installer les composants à des moments différents, des problèmes de compatibilité peuvent survenir. Par exemple, l'image *WalT* d'une expérience récemment publiée peut s'appuyer sur les fonctionnalités *WalT* les plus récentes alors que l'exploitation du serveur *WalT* il y a deux ans peut être incapable de les gérer. Il faudra alors être en mesure de pouvoir revenir vers une version antérieur de la plate-forme et/ou des nœuds comme nous le verrons dans la Section 2.4.2.2.

La première étape pour résoudre ces problèmes de compatibilité est de les découpler. Pour ce faire, nous attribuons le numéro de version API à chaque interface de composant. L'API est l'ensemble des fonctions qu'un composant expose à un autre. Par exemple, sur commande d'un client, le serveur peut demander à un nœud donné de faire clignoter sa DEL — à des fins d'identification par exemple. Puisque cette fonctionnalité du nœud est exposée au serveur, nous pouvons la considérer comme faisant partie de l'API du nœud. De même, chaque fois qu'un nœud démarre, il avertit le serveur de cet événement, de sorte que la fonction d'enregistrement des événements fait partie de l'API du serveur. La FIGURE 2.8 illustre ces notions.

Si l'API du serveur change, certains des appels de traitement à distance effectués par le nœud échoueront. Ainsi, la version du serveur modifié n'est plus compatible avec la version précédente du nœud. De même, un changement dans l'API du nœud rompt également la compatibilité avec le serveur.

Pour détecter toutes les ruptures de compatibilité, nous considérons l'union des API serveur et nœud : nous inspectons toutes les fonctions exposées d'un côté et de l'autre (en utilisant l'introspection de code), construisons un document listant leurs prototypes, et calculons le *hash* du document entier. Si le hachage change, cela signifie que l'API a également changé, donc nous incrémentons le numéro de version de l'API. L'ensemble du processus s'exécute automatiquement lorsqu'une nouvelle version d'un composant est publiée. Le numéro de version de l'API est ensuite utilisé lors de l'exécution pour vérifier si les composants sont compatibles ou non. De plus, nous marquons une version de serveur donnée avec deux numéros d'API différents, l'un pour la compatibilité nœud-serveur et l'autre pour la compatibilité client-serveur.

Le numéro de version d'un composant donné n'est pas lié au numéro de version de l'API. Nous utilisons plutôt un numéro de téléchargement à cette fin.

2.4.2.2 Mise à niveau et rétrogradation des versions

Lorsque nous détectons un problème de compatibilité, nous devons mettre à niveau ou déclasser l'un des deux composants concernés.

Si le client et le serveur ne sont pas compatibles, le client mettra à niveau sa version utilisé de manière transparente pour l'utilisateur afin de correspondre au numéro API client-serveur attendu par le serveur. Une telle opération permet d'utiliser facilement plusieurs plates-formes *WalT* (par exemple, une pour le déboggage et une pour les expériences à grande échelle).

Lorsque le serveur n'est pas compatible avec le code intégré dans une image *WalT* (c'est-à-dire le code qui sera exécuté par un nœud si cette image est déployée), le problème est signalé à l'utilisateur qui a le choix entre adapter le serveur ou l'image. L'adaptation de l'image — mise à jour ou retour à une version antérieure d'un logiciel sur l'image d'un nœud *WalT* — est simple avec la commande : `walt image update`

`<image>`. Cependant, lorsque l'utilisateur est préoccupé par la reproductibilité de l'expérience, il peut préférer mettre à niveau ou déclasser le serveur au lieu de le faire correspondre à la version initialement utilisée pour l'expérience. La mise à niveau ou le retour à une version antérieure du serveur est une opération plus complexe. Puisqu'il a un impact sur tous les utilisateurs, l'utilisateur doit d'abord acquérir un verrou en tapant la commande : `walt platform lock`. Le verrouillage garantira à l'utilisateur un accès exclusif au serveur et à tous les nœuds jusqu'à ce qu'il lance la commande : `walt platform unlock`. Lorsque l'utilisateur lance la commande de déverrouillage, le serveur et les nœuds reviennent à leur état précédent.

Comme nous l'avons déjà mentionné dans la Section 2.4.1, lors de la reproduction d'une expérience, les paramètres du système d'exploitation ou du serveur peuvent avoir une incidence sur les résultats — paramètres PTP pour la synchronisation de l'heure par exemple. Ainsi, nous devons non seulement mettre à jour ou revenir à une version antérieure du logiciel de contrôle *WalT*, mais aussi l'ensemble du système d'exploitation du serveur. Afin d'effectuer efficacement cette opération lourde, nous avons décidé d'utiliser à nouveau *Docker*. Chaque fois qu'une nouvelle version du serveur est téléchargée, l'image du *Docker* du système d'exploitation entier est publiée. En conséquence, le comportement du serveur est maintenant similaire à celui des nœuds : un système d'exploitation minimal est démarré en premier, et, selon la version requise, l'image *Docker* adéquate est sélectionnée, et le système d'exploitation final qui y est stocké est démarré.

Il y a cependant une différence importante entre les nœuds et le serveur : les nœuds sont sans état, le serveur ne l'est pas. Par exemple, le serveur doit sauvegarder l'état de la plate-forme avant d'effectuer un verrouillage. Le contenu de la base de données des *logs* doit également être préservé lors de la mise à niveau ou de la rétrogradation. Par conséquent, l'état doit être sauvegardé en dehors de l'image *Docker*.

2.4.2.3 Gestion des expériences WalT

Comme indiqué dans la Section 2.2.4, l'outil client `walt` fournit une catégorie de commandes pour gérer les expériences : sauvegarder, publier, rechercher, cloner, déployer, exécuter. Lors de l'utilisation de ces commandes, les opérations complexes telles qu'une mise à niveau ou un retour à une version antérieure du serveur sont complètement cachées à l'utilisateur.

2.4.2.4 Discussion

WalT répond à la plupart des défis présentés dans la Section 2.4.1 afin de permettre la reproductibilité des expériences indépendamment du temps : il fournit un moyen de revenir en quelque sorte à l'état exact de l'environnement logiciel au moment exact où l'expérience a été menée pour la première fois.

La limitation de ce mécanisme vient en fait de la couche matérielle. En particulier, les SBC que nous utilisons actuellement pourraient être difficiles à trouver dans quelques années. Néanmoins, *WalT* permettra de vérifier rapidement si les résultats sont similaires sur le nouveau matériel.

Conclusions

WalT est une plate-forme reproductible permettant de réaliser des expériences reproductibles. Elle complète les plates-formes spécialisées existantes avec la possibilité de reproduire et de déployer sa propre plate-forme d'expérimentation pour le débogage, les comparaisons et les tests de réseaux dans des environnements de production en conditions réelles. *WalT* utilise des composants standards à faible coût, des logiciels libres et permet une installation facile grâce à l'utilisation du PoE. Nous l'avons notamment utilisée pour mener les expériences sur l'impact de la contention sur les algorithmes d'adaptation de débits, présentées dans le Chapitre 4 de ce manuscrit.

Il est par ailleurs question de pérenniser la reproductibilité et la répétabilité des expériences dans le temps. Notre expérience avec *WalT* montre que les chercheurs ont souvent besoin de reproduire des expériences faites précédemment dans une configuration spécifique, ce qui est difficile si la plate-forme expérimentale a évolué. Sur la base des développements actuels de *WalT*, nous proposons plusieurs mécanismes pour garder une trace des modifications et pouvoir revenir à une version donnée dans le passé.

À l'avenir, nous prévoyons d'améliorer la fonction de gestion de l'expérience dans *WalT* en sauvegardant les résultats bruts obtenus dans une expérience donnée. Une telle caractéristique faciliterait la comparaison entre deux séries d'expériences, chacune effectuée sur une plate-forme *WalT* différente. Fournir un ensemble de résultats obtenus sur plusieurs plates-formes *WalT* permettrait également de mieux élaborer les conclusions de la recherche.

802.11 à l'échelle de la ville

Sommaire

3.1 Vue d'ensemble 81
3.1.1 Topologie Wi-Fi à l'échelle de la ville 81
3.1.1.1 L'ensemble de données « Freeboxes » 82
3.1.1.2 L'ensemble de données généré 82
3.1.2 Modèle de mobilité 82
3.1.2.1 Modèle de trajectoires 83
3.1.2.2 Classes d'utilisateurs 83
3.1.3 Modèle de connectivité 83
3.1.3.1 Delta de connexion 83
3.1.3.2 Déclenchement du *handover* 84
3.2 Impact de la vitesse, du delta de connexion et de la densité sur la connectivité en mobilité 85
3.2.1 Analyse macroscopique 85
3.2.1.1 Taux de connectivité 87
3.2.2 Analyse microscopique 87
3.2.2.1 Etat connecté 88
3.2.2.2 État non connecté 88
3.2.3 Détails sur la densité des points d'accès 89
3.2.3.1 Impact sur les durées de connectivité et de déconnexion 89
3.2.3.2 Impact sur le taux de connectivité 89

Introduction

Le WI-FI est omniprésent dans les villes. Tout d'abord car le nombre de *points d'accès* publics — *hotspots* — ne cesse d'augmenter [1] avec un déploiement mondial de 124 millions en 2017 et un déploiement estimé de 549 millions d'ici 2022. Ensuite le nombre de foyers ayant une souscription internet fixe est important dans les pays développés — 74% des foyers de l'Union Européenne [148] par exemple. Il est envisageable de supposer qu'une écrasante majorité de ces lignes fixes utilisent un modem routeur WI-FI— notamment lorsque des *boxes* d'opérateurs sont fournies — venant s'ajouter aux nombres de *points d'accès* WI-FI opérants. Dans une ville nous avons alors une très forte densité de *points d'accès*, couvrant potentiellement une majeure partie de celle-ci.

En parallèle, le nombre de *smartphones* est en constante augmentation et ceux-ci représentent l'écrasante majorité du trafic IP mobile avec 88% de celui-ci. 46% de leurs communications passent par le réseau cellulaire tandis que les 54% restantes utilisent des *points d'accès* WI-FI ou des *femtocells* déployées chez des particuliers. Il faut cependant noter qu'une très grande partie de ce trafic *offloadé* s'effectue lorsqu'un utilisateur est chez lui ou dans un lieu possédant un *point d'accès* public, mais rarement lorsqu'il est en mobilité. Lorsqu'un utilisateur se déplace il va plutôt solliciter le réseau cellulaire pour assurer ses communications au lieu d'utiliser des *points d'accès* WI-FI car ceux-ci n'offrent pas une aussi bonne gestion de la mobilité que les réseaux cellulaires et ne sont pas forcément ouverts au public.

Or, la démocratisation des réseaux communautaires — dans lesquels un particulier offre une partie de sa bande passante pour un usage en libre service aux autres membres de sa communauté — ainsi que l'apparition d'initiatives comme *Hotspot 2.0*, permettent d'envisager l'utilisation de *points d'accès* privés par n'importe quel utilisateur. Si l'ensemble des *points d'accès* privés des quartiers résidentiels étaient utilisables, nous aurions à notre disposition un réseau peu utilisé en journée — les gens ayant majoritairement un lieu de travail différent de leur lieu de résidence. Il serait intéressant de pouvoir profiter de ces *points d'accès* dans un contexte de mobilité, afin de ne pas utiliser uniquement une connectivité cellulaire.

C'est pourquoi nous nous proposons d'estimer les possibilités offertes par l'utilisation de l'ensemble des *points d'accès* d'une ville. Pour cela il faut estimer la couverture radio que propose un tel déploiement, ainsi que les durées pendant lesquelles un utilisateur mobile est connecté ou déconnecté, en fonction de sa vitesse de déplacement et de son *handover*.

Plusieurs travaux ont été menés afin de caractériser le déploiement des *points d'accès* 802.11 dans un environnement urbain. Farshad et al. [149] se sont intéressés aux caractéristiques des *points d'accès* déployés dans la ville d'Édimbourg. Ils utilisent des téléphones Android afin de scanner passivement l'environnement lors de déplacements effectués en bus dans la ville afin de capturer les *beacons* émis par les *points d'accès* de leur voisinage. Il enregistre alors les information concernant le SSID, BSSID, si un le *point d'accès* est protégé ou non, et le canal sur lequel il opère, ainsi que les coordonnées GPS du téléphone lors de la capture. Ils observent qu'une vaste majorité des *points d'accès* découverts opèrent sur les canaux 1, 6 et 11, et que le nombre de *points d'accès* voisin moyen est de 13.4 pour un lieu donné. Dans leur étude, Castignani et al. [150] se sont intéressés aux caractéristiques des réseaux sans fil communautaires présents dans la ville de Rennes. Lors de leurs déplacements, leur équipement mobile effectue une recherche active sur l'ensemble des canaux toutes les $2s$, et enregistre ses coordonnées GPS toutes les secondes. Ils observent eux aussi que la majorité des *points d'accès* opèrent sur les 3 canaux orthogonaux, et que la distance moyenne aux *points d'accès* observée est de 32,3 m. Enfin, ils constatent que le réseau communautaire *FreeWifi* proposé par l'opérateur *Free* permet à lui seul de couvrir 67.43% de leurs trajets, et que lorsque l'ensemble des réseaux communautaires sont pris en compte, on obtient une couverture proche de 75%. Bychkovsky et al. [151] se sont intéressé à l'exploitation des *points d'accès* 802.11 présents dans les villes de Boston et Seattle par des utilisateurs se déplaçants en voiture. Les véhicules sont équipés d'un ordinateur ayant une carte 802.11B associée à une antenne avec un gain de 5,5 dBi. Lors du déplacement d'un véhicule, l'équipement embarqué effectue une recherche active sur l'ensemble des canaux, et enregistre le SSID, le BSSID, le canal et le RSSI de l'ensemble des *points d'accès* ayant envoyé une *Probe Response*. Il tente ensuite une association avec le *point d'accès* ayant le RSSI le plus élevé. Si celle-ci échoue alors une autre recherche active est initiée, sinon il tente de récupérer une adresse IP. Si celle-ci est obtenue, alors il envoie des messages *ping* vers un serveur connu, si cette opération réussi alors l'équipement commence à envoyer des *pings* vers le *point d'accès*, et essaye d'envoyer un fichier via une connection TCP vers un serveur distant. Les auteurs définissent la durée de connection à un *point d'accès* comme l'intervalle de temps

entre le premier *ping* vers celui-ci et le dernier. Ils observent que peu d'association sont faites lorsque la vitesse du véhicule est supérieure à 70 km/h, et que la durée de connection à un même *point d'accès* décroit avec l'augmentation de la vitesse. Dans d'autres travaux de Castignani et al. [152], ils se sont intéressé à l'utilisation des *points d'accès* 802.11 des réseaux communautaires, lors de phases de mobilité. Les auteurs ont développé une application (*Wi2Me Traces Explorer*) permettant aux smartphones équipés de collecter des informations concernant l'environnement cellulaire et 802.11 lorsqu'un utilisateur se déplace. L'application effectue une recherche active sur l'ensemble des canaux 802.11 afin de récupérer l'ensemble des *points d'accès* appartenant à différents réseaux communautaires, et tente ensuite de s'associer au *point d'accès* dont le signal reçu a le RSSI le plus élevé. Si cette association réussie et que l'appareil obtient une adresse IP, alors une tentative d'authentification au travers d'un portail captif est effectuée afin d'obtenir un accès à Internet. Si cette étape aboutit, alors l'application envoie une série de *pings* vers le *point d'accès* et un serveur distant, après quoi elle tente de récupérer et d'envoyer des fichiers via HTTP vers ce même serveur. En parallèle, l'application tente d'obtenir un accès à Internet via son interface cellulaire, et effectue les même transferts de fichiers que pour l'interface 802.11. Ils observent qu'au moins un *point d'accès* d'un réseau communautaire est présent 98.9% du temps, qu'il est possible de s'associer à un *point d'accès* et de récupérer une adresse IP respectivement 93.9% et 83.1% du temps, mais qu'un accès à Internet n'est possible que 53.8% du temps. La connectivité via le réseau cellulaire est quand à elle possible 99.2% du temps.

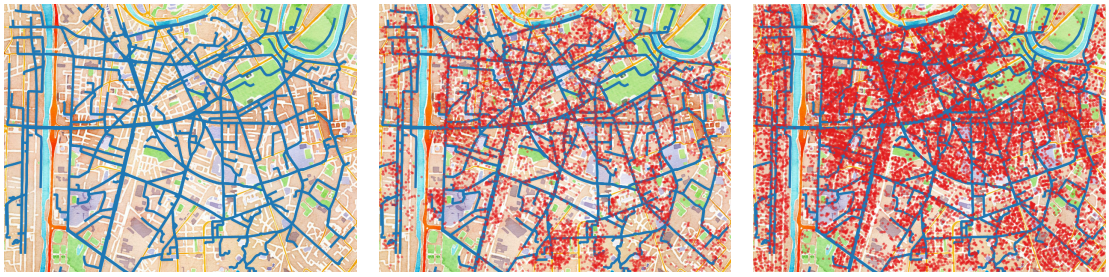
Dans ce chapitre nous présentons les résultats des simulations menées sur la ville de Grenoble afin d'évaluer le type de trafic applicatif qu'un utilisateur mobile peut espérer utiliser lorsqu'il se déplace. Nous présentons dans la première section les deux jeux de données utilisés, le modèle de mobilité composé notamment de différentes classes d'utilisateurs se déplaçant à des vitesses différentes, et le modèle de connectivité basé sur la durée de déconnexion utilisé. La deuxième section présente les résultats des simulations lorsque la vitesse de déplacement, la durée de connectivité ou la densité de *points d'accès* varient.

3.1 Vue d'ensemble

La problématique de ce chapitre pourrait s'exprimer ainsi : Si l'ensemble des *points d'accès* WI-FI déjà déployés dans les villes étaient accessibles au public, quelles seraient les applications possibles pour les utilisateurs mobiles ? Afin d'essayer de répondre à une telle question, il est nécessaire de procéder à un certain nombre d'ajustements dans le but de mettre en place notre environnement de simulations. Pour observer les relations entre les *points d'accès* WI-FI à l'échelle de la ville et les utilisateurs mobiles, il convient de préciser trois concepts clés : Topologie WI-FI à l'échelle de la ville, mobilité des utilisateurs finaux et comportement des appareils WI-FI.

3.1.1 Topologie WI-FI à l'échelle de la ville

Le déploiement de *points d'accès* WI-FI publics se généralise dans les villes, mais leur utilisation présente certains inconvénients. Tout d'abord un tel déploiement n'implique pas une couverture complète de la ville. Certains lieux d'intérêt sont couverts, comme les musées et mairies par exemple, mais d'autres



(a) Chemins générés (b) Carte des *points d'accès* « Freeboxes » (c) Carte des *points d'accès* générés

FIGURE 3.1 – Position géographique des *points d'accès* WI-FI dans la ville de Grenoble

zones publiques ne le sont pas ou peu comme les parcs et les chaussées. De plus, même si un déploiement généralisé était envisagé, celui-ci a un coût et ne pourrait pas atteindre une couverture totale instantanément. Enfin la maintenance d'une telle infrastructure représente un budget, et certaines villes peuvent ne pas y voir une dépense primaire.

D'un autre côté, des *points d'accès* WI-FI privés sont déjà déployés dans la plupart des bâtiments — cafés, restaurants, etc. — et lieux de résidence. Il serait donc envisageable de s'appuyer sur l'infrastructure existante plutôt que d'investir dans une infrastructure dédiée. Nous allons donc chercher à quantifier le taux de couverture offert par de tels déploiements ainsi qu'à évaluer la connectivité offerte par l'utilisation des *points d'accès* déjà existants si ceux-ci étaient libres d'accès. Pour cela nous avons besoin de simuler le déplacement d'un utilisateur au sein d'un espace couvert par différents *points d'accès*. Afin d'être au plus proche d'un déploiement réel, nous proposons d'utiliser des données topologiques réalistes de la position des *points d'accès* 802.11. Pour cela nous avons utilisé deux jeux de données différents.

3.1.1.1 L'ensemble de données « Freeboxes »

Nous avons obtenu un ensemble de données contenant les emplacements de toutes les « Freeboxes » de la ville de Grenoble, celui-ci est présenté sur la FIGURE 3.1b. Il correspond à l'ensemble des *points d'accès* Internet privés des abonnés à *Free*, un des quatre principaux FAI (Fournisseur d'accès Internet) français — *Orange*, *Free*, *SFR* et *Bouygues*. Ce jeu de données nous a été fourni par Mathieu Cunche, qui l'a obtenu en interrogeant un *Wi-Fi Positioning System* [153]. Afin d'obtenir les coordonnées géographique d'un *point d'accès*, il faut fournir au *Wi-Fi Positioning System* deux adresses MAC valides. Or, contrairement aux autres boxes d'opérateurs, les « Freeboxes » ont la caractéristique de posséder deux adresses MAC consécutives, facilitant dès lors les requêtes faites sur le *Wi-Fi Positioning System*. C'est pourquoi nous avons utilisé les données de cet opérateur en particulier.

3.1.1.2 L'ensemble de données généré

Nous voulons avoir un ensemble de données plus dense s'approchant du nombre de *points d'accès* correspondant aux quatre principaux FAI français. Comme il est difficile d'obtenir ces localisations auprès du FAI, nous avons décidé de générer les positions des *points d'accès* à partir des données topologiques de la ville. L'idée repose sur le fait que l'on peut supposer que des *points d'accès* privés sont déployés dans l'ensemble des bâtiments et qu'il existe une relation potentielle entre le nombre de *points d'accès* présents et la surface d'un bâtiment. La description des bâtiments est obtenue grâce à *OpenStreetMap* et peut être utilisée dans n'importe quel système d'information géographique (SIG) — tel que QGIS [154]. Nous avons développé un plugin pour QGIS qui évalue chaque surface de bâtiment et génère un certain nombre de *points d'accès*. Le nombre de *points d'accès* est configurable et est piloté par des « seuils ». Cela signifie que l'on peut définir une valeur de surface minimale pour générer un certain nombre de *points d'accès*, ainsi qu'un seuil maximal au delà duquel aucun *point d'accès* n'est plus généré. Il est également possible de définir des seuils intermédiaires qui génèrent différents nombres de *points d'accès* afin d'avoir des distributions spécifiques. Chaque *point d'accès* est situé au hasard — ils ont des coordonnées aléatoires — au sein de leur bâtiment. La FIGURE 3.1c représente un ensemble de données plus dense que nous générons à l'aide de cette méthode. Nous avons réglé ces différents paramètres afin d'obtenir quatre fois plus de *points d'accès* que l'ensemble de données « Freeboxes » dans le but de s'approcher d'une densité de *points d'accès* correspondant au nombre de FAI présents. Une fois les différentes coordonnées GPS des *points d'accès* obtenues, celles-ci sont exportées vers un fichier au format KML servant à gérer l'affichage de données géospatiales, et nous permettant de les afficher dans *Google Earth*.

3.1.2 Modèle de mobilité

Afin de mesurer la connectivité pour les utilisateurs en mouvement, nous devons définir un modèle de mobilité. Ce modèle est constitué de deux éléments : 1. la classe de l'utilisateur, caractérisée par sa vitesse de déplacement et 2. la trajectoire que cet utilisateur va suivre dans la ville.

3.1.2.1 Modèle de trajectoires

L'obtention d'un ensemble de données réelles représentant la mobilité des utilisateurs dans une ville spécifique peut être difficile s'il n'y a pas d'ensemble de données actuellement disponible — mise en place d'un échantillon d'utilisateurs sur lesquels on collecte l'ensemble de leurs déplacements sur plusieurs jours / mois. Nous avons décidé de générer des chemins « réalistes » — proches du chemin qu'un utilisateur utilisera pour aller d'un endroit à un autre — afin de faire nos simulations.

Un premier modèle de mobilité a été réalisé à partir de données existantes [155]. Le modèle extrait des propriétés de déplacement à partir d'un jeu de mobilité réel afin de générer des trajectoires « réalistes », évitant ainsi une génération basée sur une marche aléatoire. Malgré l'aspect réel des chemins (pas de boucles, rarement croisés, etc...), ce processus ne prend pas en compte les données topologiques de la ville comme les routes ou les bâtiments. Il en résulte une génération potentielle de chemins qui traversent des bâtiments ou des cours d'eau par exemple.

Dans notre modèle, nous considérons que les utilisateurs se déplacent dans la ville entre deux points en empruntant le plus court chemin. Afin de générer un tel chemin tout en considérant la topologie de la ville, nous utilisons l'API Google Maps [156]. Nous choisissons aléatoirement deux points dans la zone contenant la ville de Grenoble — sur laquelle nous basons nos simulations —, puis nous utilisons l'API pour générer le chemin entre ces deux points. Celle-ci renvoie un ensemble de points décrivant la trajectoire parcourue prenant en compte les intersections, les courbures de routes, etc. L'ensemble des chemins générés est illustré sur la FIGURE 3.1a. Il ne s'agit peut être pas des chemins que des utilisateurs réels auraient choisi — préférences personnelles, détours par un endroit particulier, etc. — mais ne s'en éloignent pas trop. Nous nous intéressons ici à l'analyse des possibilités offertes par le WI-FI, un utilisateur parcourant un chemin suivant la voirie — même si il ne s'agit pas exactement du chemin qu'il aurait suivi — introduit moins de biais qu'un utilisateur pouvant traverser un bâtiment.

3.1.2.2 Classes d'utilisateurs

Nous définissons quatre classes d'utilisateurs différentes qui se caractérisent par la vitesse qu'ils utilisent pour se déplacer dans la ville :

- $1 m.s^{-1}$: qui peut correspondre à la vitesse d'un piéton
- $5 m.s^{-1}$: qui peut être la vitesse d'un cycliste
- $11 m.s^{-1}$: qui peut être l'allure d'un transport public urbain
- $20 m.s^{-1}$: qui est une valeur de vitesse urbaine élevée

A partir de ces classes d'utilisateurs et des chemins obtenus nous approximons le déplacement par une succession de déplacements selon la latitude et la longitude à la vitesse voulue entre deux « sous-points » d'une trajectoire.

3.1.3 Modèle de connectivité

3.1.3.1 Delta de connexion

Le WI-FI a été conçu pour fournir une connectivité sans fil de proximité à des utilisateurs statiques. L'utilisation du WI-FI dans un contexte de mobilité est un défi pour les équipements grand public car la plupart des améliorations présentées dans le chapitre 1 sont rarement déployées. Les paramètres utilisés pour déclencher le processus de *handover* ne sont pas fiables — puissance du signal reçu ou taux de pertes par exemple. Cela conduit à un scénario dans lequel l'appareil se considère comme « connecté » — pour éviter les faux positifs dus à des interférences par exemple — mais ne dispose d'aucune connectivité. Ces phénomènes se produisent lorsqu'un utilisateur s'éloigne trop de son *point d'accès* et ne peut plus recevoir ou envoyer de messages à celui-ci. Ce temps de détection de déconnexion ($t_{déclenchement}$) peut prendre plusieurs dizaines de secondes. Lorsque l'appareil passe à l'état « déconnecté », il entre en phase de recherche ($t_{recherche}$). L'appareil balaie son environnement WI-FI afin de trouver les *points d'accès* environnants et choisit le *point d'accès* suivant auquel il sera connecté. Vient ensuite le processus d'association ($t_{association}$) : l'appareil envoie éventuellement une demande de désassociation à son *point d'accès* associé

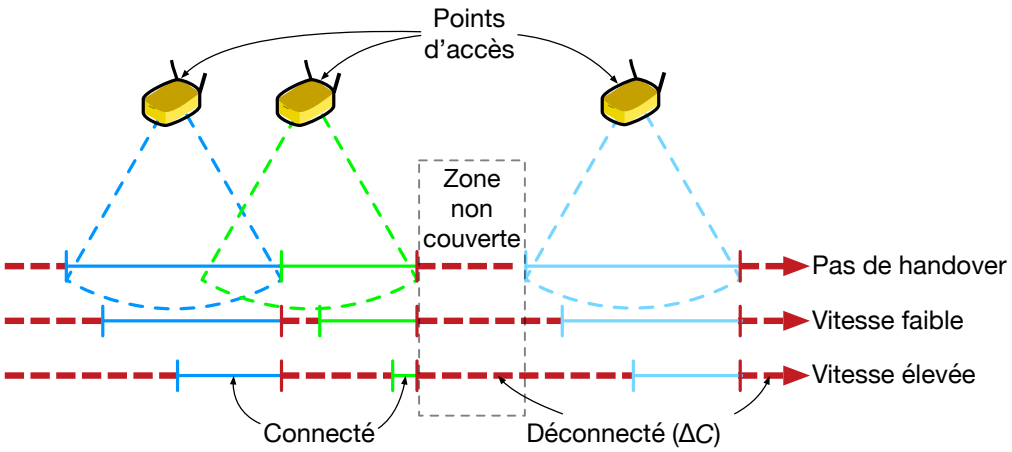


FIGURE 3.2 – Delta de connexion lors du déplacement

précédent, puis envoie des demandes d'authentification et d'association au *point d'accès* élu. Enfin, si l'utilisateur s'attend à une connexion Internet, il y a un délai supplémentaire (t_{IP}) pour obtenir une adresse IP et récupérer de l'interruption précédente.

Dans notre contexte nous définissons un delta de connexion (Δ_C) qui correspond à la durée entre deux connexions IP sur Internet utilisant le WI-FI du point de vue de l'utilisateur final — ou de l'application. Il est donc exprimé comme la somme du temps pris par toutes les étapes utilisées pour effectuer le *handover* :

$$\Delta_C = t_{déclenchement} + t_{recherche} + t_{association} + t_{IP}$$

Dans nos scénarios nous considérons plusieurs Δ_C constants. En attachant un Δ_C donné à un utilisateur se déplaçant à des vitesses différentes, on peut avoir l'intuition que ces deux valeurs auront un impact sur la durée globale de l'état « connecté ». La FIGURE 3.2 illustre ce concept : un appareil est connecté à un *point d'accès*, à un moment donné il ne pourra plus communiquer avec son *point d'accès*, pendant son delta de connexion il parcourra une distance donnée qui est couverte par un autre *point d'accès*, quand il se connectera finalement au nouveau *point d'accès* le temps restant « connecté » — avant le delta suivant —- dépend de la distance parcourue pendant la déconnexion, qui dépend de la vitesse de l'utilisateur.

Dans notre modèle, nous considérons 5 Δ_C différents :

- $0s$ la connexion entre les différents *points d'accès* est transparente. Ceci correspond à la connectivité maximale obtenable.
- $0.15s$ correspondent au délai maximum acceptable pour les communications vocales [157].
- $1s$ et $2s$ sont des Deltas de Connexion très optimistes.
- $5s$ un delta de connexion « réaliste optimiste » pour des équipements effectuant un *handover* rapide tels que le *Moto X 2nd Gen* illustré sur la FIGURE 2.7.

L'utilisation d'un Δ_C nous permet de nous affranchir de la complexité liée au *handover* en transformant son mécanisme en « boite noire ». Le fait d'utiliser plusieurs valeurs fixes de celui-ci nous permet d'observer les tendances sur son impact en mobilité, même si dans un cas réel un équipement ne change pas forcément de *point d'accès* en temps constant.

3.1.3.2 Déclenchement du *handover*

Nous avons basé le déclenchement du *handover* sur les valeurs de RSSI au niveau de l'utilisateur. Les *point d'accès* possèdent une portée radio de $50m$, et un modèle d'atténuation *Log-distance path loss* est appliqué afin d'avoir une atténuation de $-90db$ en limite de portée — restant acceptable à $1Mbps$. Lorsque

ce seuil est franchi on considère l'utilisateur déconnecté pendant une durée Δ_C prenant en compte le processus de handover et la récupération de la connectivité IP. À la fin de celui-ci l'utilisateur est reconnecté au *point d'accès* dont le signal reçu a la valeur de RSSI la plus élevée, parmi les *points d'accès* de son voisinage. Si aucun *point d'accès* n'est disponible dans l'entourage de l'utilisateur alors il reste dans un état déconnecté.

3.2 Impact de la vitesse, du delta de connexion et de la densité sur la connectivité en mobilité

Afin d'évaluer l'impact de la vitesse et de la durée de Δ_C sur la connectivité d'un utilisateur se déplaçant en ville, nous avons généré aléatoirement cent chemins différents sur lesquels nous avons fait parcourir l'ensemble des classes d'utilisateurs avec les différentes valeurs de Δ_C précédemment définies.

Pour cela nous avons développé un simulateur à évènements discrets en *Python*. Pour une classe d'utilisateur donnée, avec un Δ_C particulier, un chemin parcouru est discrétisé en une succession de points séparés d'une distance d'un mètre. L'état de connectivité d'un utilisateur est alors évalué à chaque « pas » :

- Si l'utilisateur est connecté à un *point d'accès* alors on détermine si la puissance du signal reçue est en dessous d'un certain seuil — $-90db$.
- Si ce seuil n'est pas franchi alors l'utilisateur reste connecté au *point d'accès* et celui-ci avance d'un « pas ».
- Sinon, l'utilisateur passe dans un état déconnecté d'une durée Δ_C — cette durée permettant d'abstraire la complexité liée au déclenchement, à la recherche et à la réassociation. Pendant cette période, l'utilisateur choisi son futur *point d'accès* en prenant celui avec la puissance reçue la plus élevée dans son entourage. L'utilisateur avance alors d'un nombre de « pas » nécessaire à l'écoulement de la durée Δ_C.

3.2.1 Analyse macroscopique

Un exemple de chemin simulé avec les différentes frises temporelles de connectivité associées est illustré sur la FIGURE 3.3. Le chemin présenté sur la FIGURE 3.3a est parcouru d'est en ouest et contient une zone non couverte par des *points d'accès* WI-FI— un cours d'eau. Les frises temporelles sont présentées pour le jeu de données des « Freeboxes »– à droite — et le jeu de données généré — à gauche — pour l'ensemble des classes d'utilisateurs avec l'ensemble des Δ_C envisagés. Elles montrent les périodes pendant lesquelles un utilisateur est connecté ou non le long de ce trajet.

On remarque tout d'abord qu'un « trou de connectivité » est présent pour les deux jeux de données au niveau du passage du pont — environ 75% du trajet. Ce « trou de connectivité » est cependant plus long sur le jeu de données des « Freeboxes » car moins de *points d'accès* sont présents autour du pont.

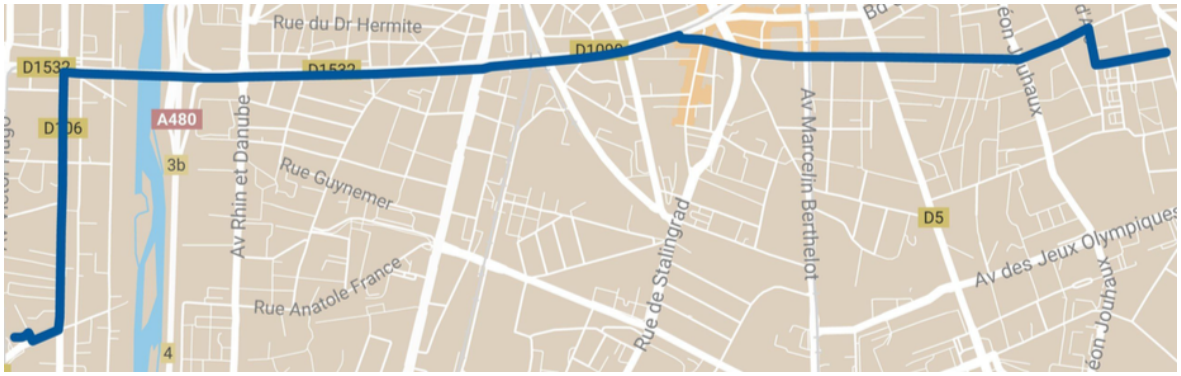
Avec un $\Delta_C = 0s$ — FIGURES 3.3b et 3.3c — on remarque que pour le jeu de données généré seul le pont présente une zone de non connectivité alors que pour le jeu de données des « Freeboxes » certaines zones après le passage de ce pont sont non couvertes.

Lorsque le $\Delta_C = 0.15s$ — FIGURES 3.3d et 3.3e — on observe le même comportement que lorsque celui-ci est nul. Dans la pratique une déconnexion de $150ms$ a lieu, mais n'est pas visible sur la frise temporelle.

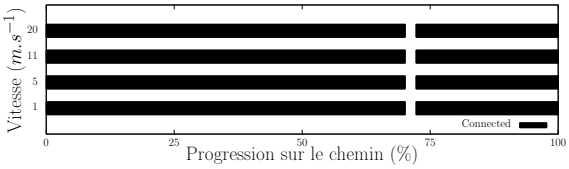
Quand $\Delta_C = 1s$ — FIGURES 3.3f et 3.3g — on observe une légère dégradation de la connectivité pour les utilisateurs se déplaçant à grande vitesse sur le jeu de données des « Freeboxes ».

Quand $\Delta_C = 2s$ — FIGURES 3.3h et 3.3i — des micro « trous de connectivité » plus visibles commencent à apparaître. Ce phénomène est plus marqué sur les utilisateurs se déplaçant à grande vitesse car ils parcourent un pourcentage de leur trajet plus grand quand ils ont besoin de changer de *point d'accès*.

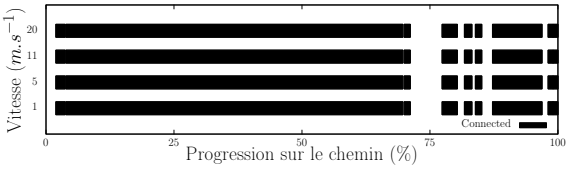
Pour $\Delta_C = 5s$ — FIGURES 3.3j et 3.3k — on observe que les utilisateurs se déplaçant à $11m.s^{-1}$ ne sont plus déconnectés uniquement lorsqu'ils changent de *point d'accès*, mais bien plus longtemps. En effet, lors de la déconnexion, le *point d'accès* élu comme *point d'accès* suivant lors de la recherche n'est plus disponible — hors de portée — lorsque Δ_C arrive à son terme. Il faut donc chercher un autre *point d'accès* dans le voisinage. Les utilisateurs se déplaçant à $20m.s^{-1}$ n'ont quasiment plus de périodes pendant lesquelles ils



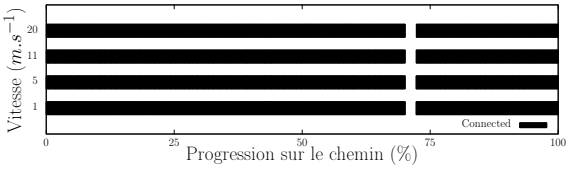
(a) Trajet parcouru d'est en ouest



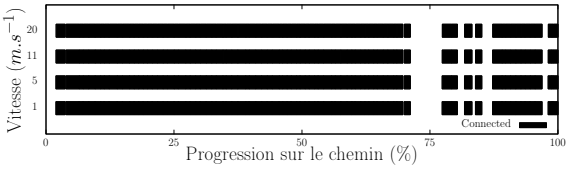
(b) $\Delta_C = 0s$ avec le jeu de données généré



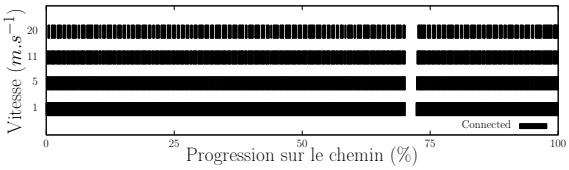
(c) $\Delta_C = 0s$ avec le jeu de données des « Freeboxes »



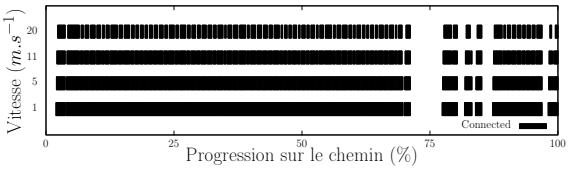
(d) $\Delta_C = 0.15s$ avec le jeu de données généré



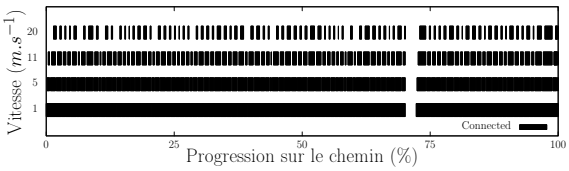
(e) $\Delta_C = 0.15s$ avec le jeu de données des « Freeboxes »



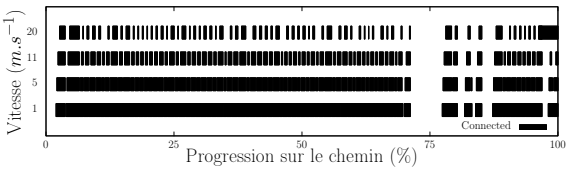
(f) $\Delta_C = 1s$ avec le jeu de données généré



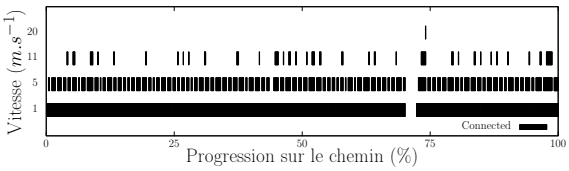
(g) $\Delta_C = 1s$ avec le jeu de données des « Freeboxes »



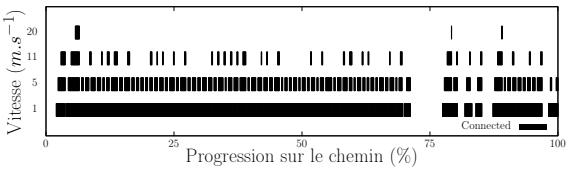
(h) $\Delta_C = 2s$ avec le jeu de données généré



(i) $\Delta_C = 2s$ avec le jeu de données des « Freeboxes »

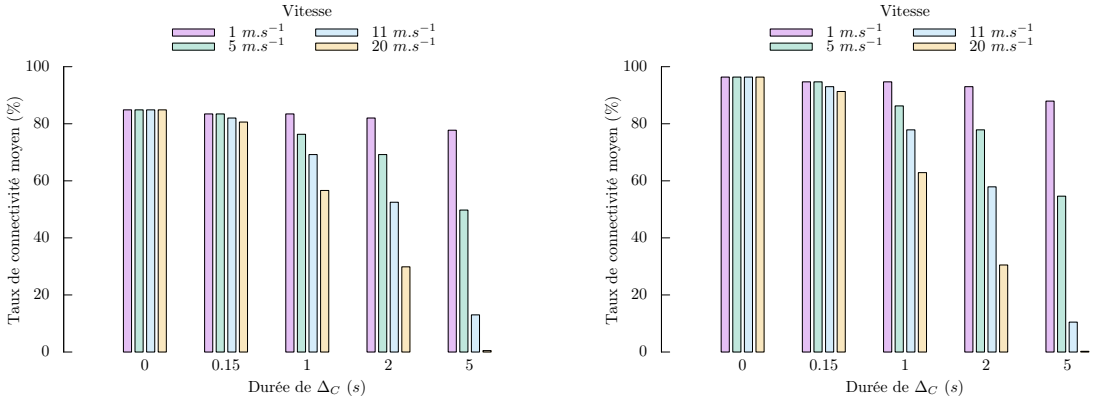


(j) $\Delta_C = 5s$ avec le jeu de données généré



(k) $\Delta_C = 5s$ avec le jeu de données des « Freeboxes »

FIGURE 3.3 – Frises temporelles de connectivité le long d'un chemin pour différentes classes d'utilisateurs avec le passage d'un pont sans *point d'accès* sur le jeu de données généré et celui des « Freeboxes »



(a) Jeu de données des « Freeboxes »

(b) Jeu de données généré

FIGURE 3.4 – Taux de connectivité moyen sur la durée d'un trajet sur l'ensemble des chemins pour différentes classes d'utilisateurs et différentes valeurs de Δ_C

sont connectés. Les utilisateurs se déplaçant à $5m.s^{-1}$ commencent à avoir des déconnexions plus longues que la durée Δ_C. On remarque que ces tendances sont partagées entre les deux jeux de données.

3.2.1.1 Taux de connectivité

La FIGURE 3.4 représente les taux de connectivité moyen des différentes classes d'utilisateurs avec différentes valeurs de Δ_C pour l'ensemble des chemins simulés. Le taux de connectivité correspond à la durée pendant laquelle un utilisateur est connecté par rapport à la durée totale de son trajet. Remarquons qu'avec un $\Delta_C = 0s$, la vitesse n'a pas d'influence sur les moments connectés et que les utilisateurs sont connectés 95% du temps sur le jeu de données simulé, et 85% du temps sur le jeu de données des « Freeboxes ». Ceci correspond à la couverture maximale espérée.

Cependant, dès que nous augmentons la durée du *handover*, nous observons une différence plus importante en termes de temps de connexion en raison des variations de vitesse.

Le taux de connectivité reste acceptable en utilisant une durée $\Delta_C = 0.15s$ pour les deux jeux de données.

Lorsque $\Delta_C = 1s$ le taux de connectivité commence à se dégrader significativement pour les utilisateurs se déplaçant à $20m.s^{-1}$. Ils ne sont connectés à un *point d'accès* que 60% du temps de leurs trajets. Les autres classes d'utilisateurs sont aussi affectées, mais ont toutes un taux de connectivité supérieur à 70% du temps. Il est par ailleurs intéressant de constater que les utilisateurs se déplaçant à $1m.s^{-1}$ ne sont quasiment pas affectés, et cela sur les deux jeux de données.

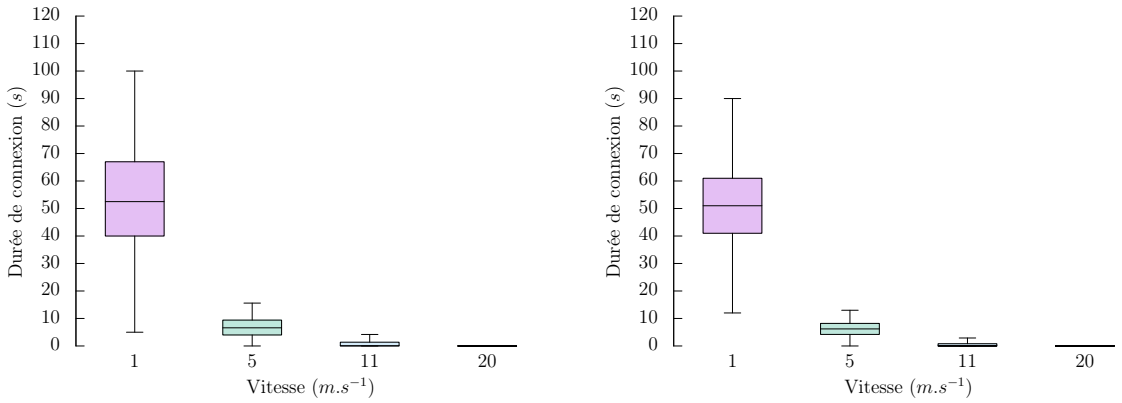
Un Δ_C de $2s$ impacte davantage les utilisateurs rapides. Ceux se déplaçant à $20m.s^{-1}$ sont connectés moins d'un tiers de leurs trajets tandis que ceux se déplaçant à $11m.s^{-1}$ le sont moins de 60% du temps.

Enfin quand Δ_C vaut $5s$, seuls les utilisateurs se déplaçant à $1m.s^{-1}$ peuvent espérer avoir un taux de connectivité acceptable. Ceux se déplaçant à $5m.s^{-1}$ sont connectés à un *point d'accès* moins de 60% du temps de leurs trajets, ceux se déplaçant à $11m.s^{-1}$ le sont moins de 20% du temps et ceux se déplaçant à $20m.s^{-1}$ ne sont plus connectés.

Ceci met en évidence le fait qu'avec une durée de *handover* élevée, combinée à des vitesses élevées, le temps total de connexion est considérablement réduit. Au contraire, avec une durée de *handover* plus rapide, nous pouvons atteindre un état de connexion proche du maximum offert par la couverture réelle. Les utilisateurs se déplaçant à des vitesses faibles — les piétons tout particulièrement — peuvent cependant avoir un taux de connectivité relativement important malgré des durées de *handover* élevées.

3.2.2 Analyse microscopique

Dans la partie précédente nous avons analysé le taux de connectivité des différentes classes d'utilisateurs sur l'ensemble de leurs trajets. Bien que cela donne une bonne vue d'ensemble de la couverture et de sa corrélation avec la vitesse de l'utilisateur, cela ne donne pas assez d'informations pour déterminer les



(a) « Freeboxes » (b) Générés

FIGURE 3.5 – Distribution des 95% des durées de connexion par *point d'accès* pour chaque classe d'utilisateurs sur les deux jeux de données pour un $\Delta_C = 5s$ et sur l'ensemble des chemins

utilisations possibles de ces *points d'accès*. En plus du taux de connectivité, nous avons besoin de quantifier les durées pendant lesquelles un utilisateur est dans l'état connecté — la durée pendant laquelle il reste sur un même *point d'accès*— ou déconnecté — la durée pendant laquelle il n'est pas associé.

3.2.2.1 Etat connecté

Les FIGURES 3.5a et 3.5b présentent les distributions de durée de connectivité d'un utilisateur pour différentes vitesses avec $\Delta_C = 5s$. On observe ici l'impact de la vitesse sur la durée de connexion à un *point d'accès*. Plus un utilisateur se déplace vite, moins il reste connecté longtemps sur un même *point d'accès*. D'une part car il parcourt la distance pendant laquelle il est connecté plus vite, mais d'autre part car il parcourt aussi la distance pendant laquelle il effectue un *handover* plus vite, comme illustré sur la FIGURE 3.2, réduisant d'autant plus le temps pendant lequel il sera connecté.

Tout d'abord nous observons que les piétons peuvent espérer avoir des durées de connexion sur un même *point d'accès* relativement longues, avec 50% de ces durées comprises entre $40s$ et $67s$ lors de l'utilisation des « Freeboxes » et comprises entre $41s$ et $61s$ lors du l'utilisation du jeu de données généré. Nous constatons cependant que cette classe d'utilisateurs possède une grande variabilité dans les durées de connectivité, celles-ci allant de quelques secondes à une centaine de secondes.

Les autres classes d'utilisateurs ont des temps de connectivité sur un même *point d'accès* beaucoup plus courtes : une dizaine de secondes pour les utilisateurs se déplaçant à $5m.s^{-1}$, moins de $5s$ pour ceux se déplaçant à $11m.s^{-1}$ et moins d'une seconde pour les plus rapides.

3.2.2.2 État non connecté

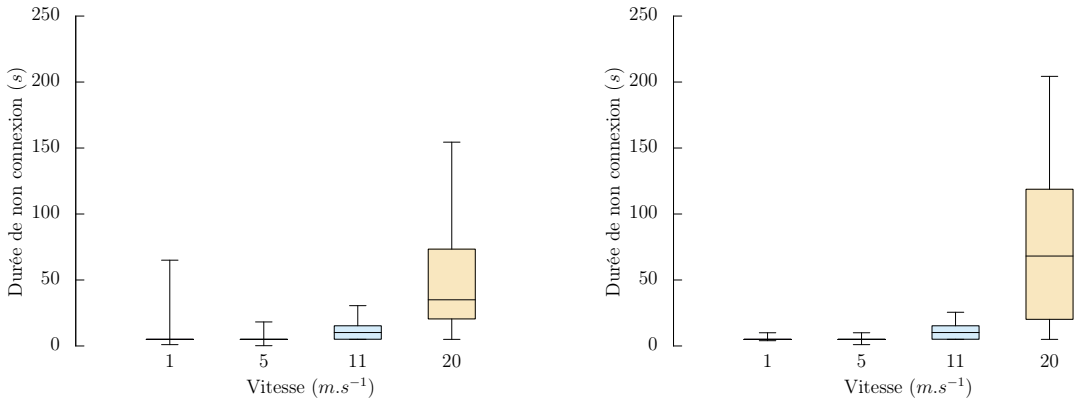
Bien que la durée de la connexion donne des informations sur le temps disponible pour un utilisateur, il est également important de caractériser les durées de déconnexion. Celles-ci donnent une information sur les durées espérées entre les périodes de connectivité.

Les FIGURES 3.6a et 3.6b présentent les durées de non connectivité observées dans 95% des cas avec $\Delta_C = 5s$.

Nous constatons tout d'abord que les utilisateurs se déplaçant à $20m.s^{-1}$ ont des temps de déconnexion très longs. Ceci est dû au fait qu'ils n'arrivent pas à s'associer à un *point d'accès* lors de leurs déplacements. Lors de la procédure de *handover*, l'équipement choisit un *point d'accès* auquel s'associer pendant une phase de recherche — celui avec le meilleur niveau de signal dans notre scenario —, or lorsque la nouvelle association est initiée, ce *point d'accès* peut ne plus être à portée radio car l'utilisateur en est sorti.

Les utilisateurs se déplaçant à $1m.s^{-1}$ et $5m.s^{-1}$ ont majoritairement des durées de déconnexion de $5s$ et correspondent à la durée de Δ_C nécessaire afin d'effectuer un *handover*.

Les utilisateurs se déplaçant à $11m.s^{-1}$ ont des durées de non connectivité de quelques dizaines de secondes. Il s'agit des mêmes déconnexions que pour les utilisateurs se déplaçant à $20m.s^{-1}$, elles sont



(a) « Freeboxes » (b) Générés

FIGURE 3.6 – Distribution de la durée de non connexion pour chaque classe d'utilisateurs sur les deux jeux de données pour un Δ_C de 5s dans 95% des cas

seulement moins fréquentes.

3.2.3 Détails sur la densité des points d'accès

3.2.3.1 Impact sur les durées de connectivité et de déconnexion

Sur les FIGURES 3.5 et 3.6 nous pouvons observer d'autres caractéristiques concernant les deux jeux de données étudiés.

Concernant les durées de connectivité, les valeurs du jeu de données généré sont plus resserrées que celles du jeu de données des « Freeboxes ». Ceci peut s'expliquer par la densité des *points d'accès* plus faible dans le cas du jeu de donnée des « Freeboxes ». En effet, si l'on se concentre sur un utilisateur se déplaçant à $1 m.s^{-1}$, lorsque celui-ci perd sa connectivité, il va chercher à s'associer au *point d'accès* environnant ayant le plus fort signal reçu.

Lorsque la densité de *points d'accès* n'est pas élevée, il est possible que le prochain *point d'accès* soit éloigné. Si celui-ci se situe dans le sens de la marche de l'utilisateur alors il va commencer par se rapprocher de celui-ci avant de s'en éloigner, et donc rester connecté plus longtemps. Si au contraire, le *point d'accès* ne se situe pas dans le sens de la marche alors l'utilisateur ne fait que s'en éloigner, et va donc rester moins longtemps associé à celui-ci.

Si la densité est élevée, l'utilisateur a plus de chance de s'associer avec un *point d'accès* relativement proche. Le même phénomène sera observé, mais il sera amoindri.

Concernant les durées de déconnexion, la densité de points d'accès étant moindre sur le jeu de données des « Freeboxes », on observe que les utilisateurs se déplaçant avec une vitesse de $1 m.s^{-1}$ peuvent avoir des durées de déconnection de plusieurs dizaines de secondes dans certains cas. Typiquement lorsqu'un utilisateur entre dans une zone non couverte, telle qu'un pont ou un parc, avec peu de bâtiments autour.

Au contraire, pour les utilisateurs se déplaçant à grande vitesse une plus faible densité semble réduire la durée moyenne totale de déconnexion. En effet, dans notre modèle, lorsqu'un utilisateur perd sa connectivité, il va tenter de s'associer au *point d'accès* avec le plus fort signal. Avec une forte densité de *points d'accès* les utilisateurs vont donc faire plus de *handover*, car ils ont une plus faible probabilité de s'associer à un point d'accès éloigné dans le sens de leur marche, comme illustré sur la FIGURE 3.7. Ceci impacte négativement les utilisateurs se déplaçant rapidement car ils passent leur temps à essayer de s'associer à un *point d'accès* qui n'est plus à portée radio.

3.2.3.2 Impact sur le taux de connectivité

La variabilité en termes de « trous de connectivité » entre les deux ensembles de données nous amène à examiner l'impact de la densité des *points d'accès* sur la connectivité globale.

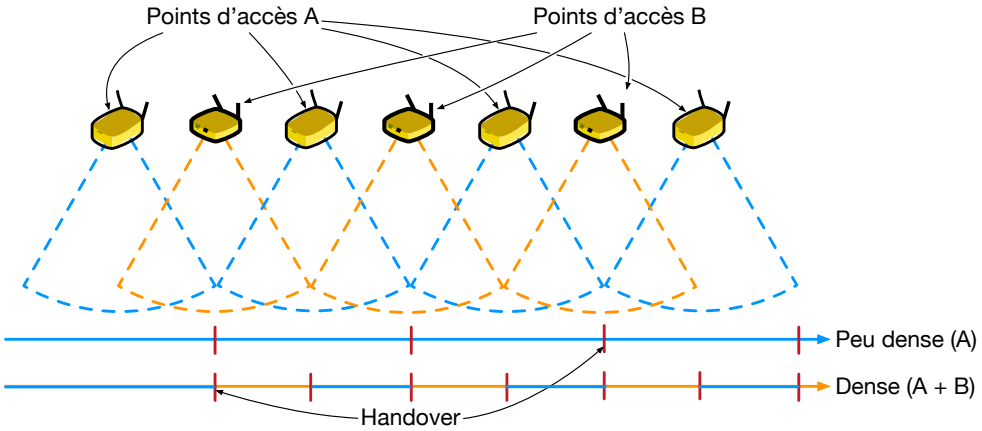


FIGURE 3.7 – Augmentation du nombre de *handovers* avec l'augmentation de la densité du nombre de *points d'accès*

Afin de mesurer cet impact, nous avons utilisé le jeu de données généré auquel nous avons enlevé des points d'accès aléatoirement afin d'obtenir des sous-ensembles de celui-ci : 75%, 50%, 25%, 12.5% et 6.25% de sa densité. Nous avons ensuite effectué les simulations présentées précédemment avec les différentes classes d'utilisateurs et les différentes valeurs de Δ_C pour chacun des sous-ensembles du jeu de données généré.

La FIGURE 3.8 illustre le taux de connectivité pour différentes densités de *points d'accès*, avec différentes valeurs de *handover* et de vitesse.

La FIGURE 3.8a représente le cas d'un *handover* instantané, $\Delta_C = 0$. La vitesse des utilisateurs n'a donc pas d'importance comme nous l'avons vu précédemment. On constate alors que la connectivité sur 80% du trajet est atteinte lorsque la densité correspond à 20% du jeu de données généré.

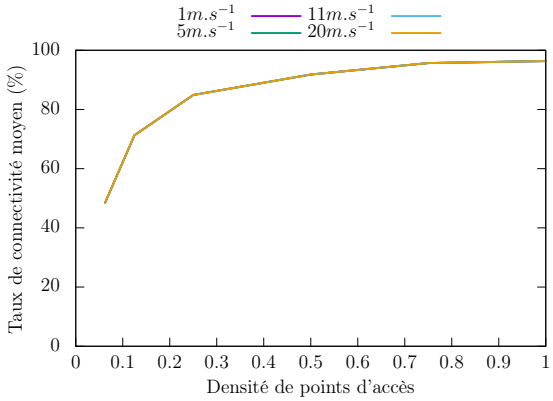
Lorsque l'on augmente la durée de *handover* jusqu'à $2s$ on remarque que le taux de connectivité est surtout altéré par la vitesse de déplacement de l'utilisateur plus que par la quantité de *points d'accès* présents tant que celle-ci est supérieure à 20% de la densité du jeu de données généré. Ceci étant dit, seuls les utilisateurs avec la vitesse la moins élevée parviennent à avoir un taux de connectivité supérieur à 80% de la durée de leur trajet.

Si l'on souhaite assurer un taux de connectivité supérieur à 70% pour les utilisateurs se déplaçant à $5m.s^{-1}$, il faut assurer une densité d'au moins 25% du jeu de données généré. Les utilisateurs se déplaçant à $11m.s^{-1}$ ne pourront espérer avoir qu'un taux de connectivité compris entre 50% et 60%, tandis que les utilisateurs se déplaçant à $20m.s^{-1}$ auront un taux de connectivité correspondant à 30% de la durée de leurs trajets. On constate d'ailleurs que dans ces conditions, la densité n'influe que très peu sur le taux de connectivité de ces derniers.

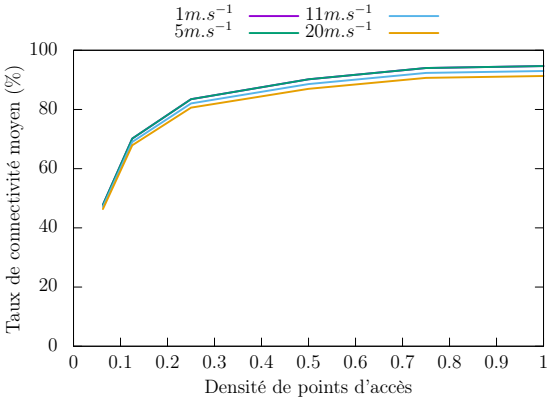
Lorsque le Δ_C vaut $5s$ les utilisateurs se déplaçant à $20m.s^{-1}$ n'ont plus de connectivité, peu importe la densité de *points d'accès*. Seuls les piétons peuvent espérer avoir un taux de connectivité supérieur à 80% mais il faudra alors avoir une densité de *points d'accès* de 35% de celle du jeu de données généré.

De manière générale, la densité à partir de laquelle les différents taux de connectivité semblent chuter se situe autour de 25% de la densité du jeu de données généré. Lorsque l'on augmente le nombre de points d'accès le gain en terme de pourcentage du trajet d'un utilisateur pendant lequel il est connecté n'augmente plus aussi franchement. À l'inverse lorsque l'on diminue le nombre de *points d'accès* les pertes en terme de taux de connectivité deviennent de plus en plus significatives.

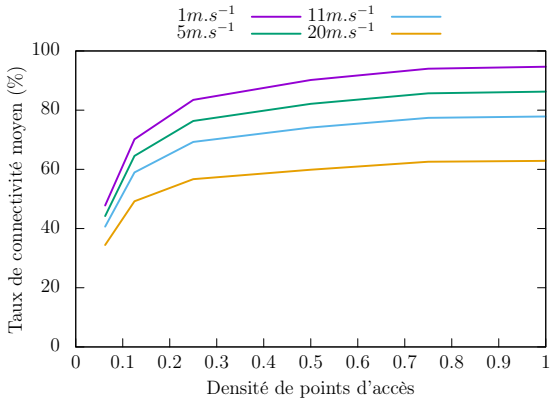
Enfin il est bon de noter que lorsque la densité de *points d'accès* est de 25% de celle du jeu de données généré, on se retrouve avec une densité de *points d'accès* proche de celle du jeu de données des « Freeboxes ». On peut d'ailleurs observer des similitudes dans les taux de connectivité illustrés sur la FIGURE 3.4.



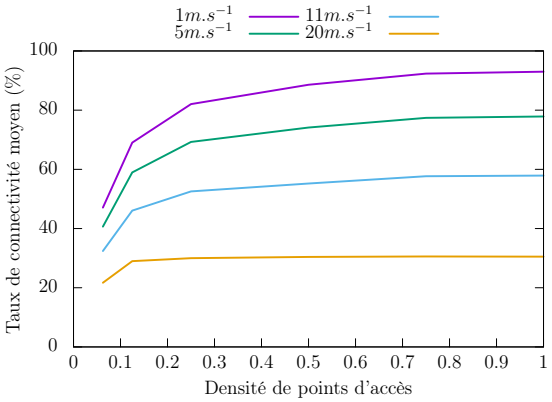
(a) Taux de connectivité en fonction de la densité avec $\Delta_C = 0s$



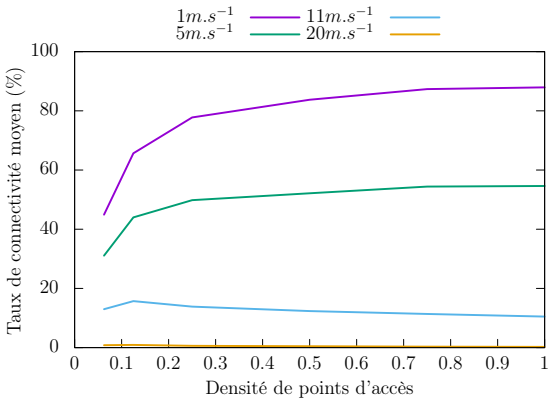
(b) Taux de connectivité en fonction de la densité avec $\Delta_C = 0.15s$



(c) Taux de connectivité en fonction de la densité avec $\Delta_C = 1s$



(d) Taux de connectivité en fonction de la densité avec $\Delta_C = 2s$



(e) Taux de connectivité en fonction de la densité avec $\Delta_C = 5s$

FIGURE 3.8 – Impact de la densité des *points d'accès* sur le taux de connectivité moyen pendant un déplacement

Conclusion

Dans ce chapitre nous avons présenté une méthode de génération de *points d'accès* en fonction des données topologiques de la ville. Ces *points d'accès* générés nous permettent de simuler un déploiement « réel » urbain. Nous avons utilisé ce jeu de données généré en parallèle d'un jeu de données réel comportant les positions géographiques des *points d'accès* de la ville de Grenoble, afin de réaliser des simulations concernant l'utilisation de *points d'accès* par un utilisateur mobile dans un contexte où l'ensemble de ces *points d'accès* seraient utilisables.

Nous avons pour cela défini un modèle de mobilité constitué d'une centaine de chemins « pseudo réalistes », de différentes classes d'utilisateurs ayant des vitesses différentes, et un modèle de *handover* englobant les différentes étapes allant de la perte de connectivité à la récupération de celle-ci. Nous avons ensuite déplacé l'ensemble des utilisateurs sur les différents chemins avec chacune des valeurs de *handover* sur les différents jeux de données afin de pouvoir estimer l'influence de chacun de ces paramètres sur la connectivité totale d'un utilisateur.

Le modèle comporte certaines limitations :

- nous ne considérons pas l'influence de l'éloignement d'un utilisateur par rapport au *point d'accès* auquel il est connecté en terme de QoS et de dégradation du signal,
- de la même manière nous ne considérons pas d'interférences, nos utilisateurs se déplacent avec une vitesse uniforme et ne font pas d'arrêts,
- nous sommes peut être optimistes avec une durée de *handover* maximale de $5s$,
- une vitesse de déplacement de $20m.s^{-1}$ est extrême dans un contexte urbain.

Cependant nous obtenons des résultats similaires à ceux obtenus par Castignani et al. [158] lors de simulations réelles en terme de taux de connectivité pour des utilisateurs pédestres, avec un taux supérieur à 80% de connectivité. Notons que ces résultats concernent une connectivité de niveau 3, à savoir que l'équipement mobile a récupéré une adresse IP, mais comme le précisent Castignani et al., cela ne garantit pas toujours un accès à internet du point de vue utilisateur.

Nos résultats indiquent plusieurs grandes tendances :

- La densité du nombre de *points d'accès* ne joue pas un rôle prédominant à partir du moment où une densité minimale est assurée.
- La vitesse de déplacement d'un utilisateur influe énormément sur la durée de son état connecté.
- La durée de *handover* impacte énormément les utilisateurs se déplaçant à des vitesses élevées.

De ces tendances nous pouvons essayer de dresser les cas d'usage possibles. Un piéton peut espérer avoir un taux de connectivité de plus de 80% sur l'ensemble de ses trajets. De plus la majorité de ses temps de connexion à un *point d'accès* unique durent plus de $40s$, et la majorité de ses durées de déconnexion durent le temps d'un *handover*. Cependant lorsque celui-ci emprunte une zone non couverte par des *points d'accès* il peut être déconnecté plusieurs dizaines de secondes si la densité de *points d'accès* n'est pas suffisante. Ainsi si la durée d'un *handover* ne permet pas d'effectuer des applications temps réel telle que la VoIP, il est tout à fait envisageable d'utiliser ce réseau pour les applications tolérantes au délai. Les durées de connexion relativement longues permettent par ailleurs d'envisager l'usage de ces *points d'accès* afin d'utiliser des applications gourmandes en bande passante comme le *streaming* plutôt que d'utiliser le réseau cellulaire. Pour celles-ci il serait d'ailleurs intéressant de se pencher sur les politiques de caching applicables sur un réseau comme celui-ci — sur la quantité de données à pré-charger afin de palier au pire cas par exemple.

Les utilisateurs se déplaçant à $5m.s^{-1}$ peuvent aussi tirer parti d'un tel déploiement. Le taux de connectivité d'un peu plus de 50% lorsque le processus de *handover* prend $5s$ laisse à penser que l'utilisation de ces *points d'accès* est compromise lors de la mobilité. Cependant il faut se pencher sur les durées passées dans l'état connecté et déconnecté. Ces utilisateurs restent associés à un *point d'accès* pendant une dizaine de secondes dans la majorité des cas. De plus les durées de déconnexions restent courtes et valent la grande majorité du temps la durée de *handover*. Si nous considérons que le lien entre l'utilisateur et le *point d'accès* est d'assez bonne qualité, alors des applications gourmandes en bande passante sont aussi envisageables.

Lorsqu'un utilisateur se déplace à $11m.s^{-1}$ les durées de connectivité deviennent trop courtes par rapport aux durées pendant lesquelles il est déconnecté. Il semble impossible d'envisager des applications

comme le streaming. Cependant de la collecte ou récupération de données est envisageable avec des délais relativement contraints.

Les utilisateurs se déplaçant à grande vitesse ne peuvent décemment pas envisager d'utiliser un tel réseau pour envoyer ou recevoir des données qui ne sont pas fortement tolérantes au délai. Cependant ce constat est à mettre en balance car dans nos simulations un utilisateur ne s'arrête jamais, ce qui n'est pas réaliste en ville — feux de signalisation, stops, ralentissements, etc. Il est donc très probable qu'un tel utilisateur puisse utiliser de manière correcte ces *points d'accès* lorsqu'il est à l'arrêt. Il serait donc intéressant d'étendre le modèle de mobilité afin de prendre en compte ces comportements.

Enfin ces résultats mettent en évidence que l'amélioration de la durée de handover est primordiale si l'on souhaite pouvoir utiliser des *points d'accès* en mobilité sans interruption. Dans le chapitre suivant nous étudierons les schémas de retransmission de différentes cartes 802.11 lorsque celles-ci perdent leur connectivité, afin d'analyser leur comportement à la rupture par rapport au standard.

Analyse du comportement des implémentations de 802.11 lors de la perte de connectivité et de la contention

Sommaire

4.1 Comportement de différentes implémentations de 802.11 lors de la perte de connectivité 96

4.1.1 Les retransmissions dans le standard . . . 96

4.1.2 Les retransmissions dans la pratique . . . 97

4.2 Comportement des cartes TP-Link TL-WN722N lors de la contention . . . 100

4.2.1 Description de la plateforme de test . . . 101

4.2.2 Analyse des débits observés . . . 101

4.2.3 Analyse des intervalles inter-trames . . . 103

4.2.3.1 Comportement attendu selon le standard . . . 103

4.2.3.2 Comportement des cartes TP-Link TL-WN722N . . . 104

4.2.4 Étude des anomalies concernant les intervalles inter-trames . . . 105

Introduction

En phase de mobilité, un équipement Wi-Fi doit changer de point d'accès lorsqu'il s'éloigne trop de celui auquel il est associé. Une des difficultés majeures est de se rendre compte de la sortie progressive de la zone de couverture, ce qui se fait implicitement en constatant une diminution de la puissance du signal reçu, ainsi qu'une augmentation des pertes de paquets envoyés à un débit donné, et donc une utilisation de modulations de plus en plus robustes. Dans ce cas, les retransmissions jouent un rôle important, car elles permettent de récupérer les pertes, mais également de tester les modulations : plus une modulation est robuste, plus le temps de transmission est important et le débit faible, mais plus la probabilité de perte diminue lorsque la puissance du signal diminue au récepteur à cause de l'éloignement. Finalement, lorsque l'équipement se retrouve hors de portée, les retransmissions sont sans effet et il doit constater la perte de connectivité.

Le mécanisme de retransmission a donc un impact important sur le temps et la qualité du processus de *handover* qui permet de rejoindre un nouveau point d'accès pendant la mobilité. Nous étudions ce mécanisme dans une première partie, tout d'abord en analysant le comportement décrit dans le standard [159], puis en observant le comportement de plusieurs cartes 802.11 lorsque celles-ci perdent la connectivité avec leur point d'accès, ce qui entraine la perte de tous les paquets qu'elles essaient de transmettre. Nous avons constaté que les politiques de retransmission variaient énormément d'une implémentation à l'autre en terme d'accès au canal et de changement de modulation, et que certaines d'entre elles pouvaient retransmettre pendant plusieurs secondes avant de conclure à une perte de connectivité.

À l'inverse lorsque nous sommes dans un contexte de contention, des collisions apparaissent au niveau du *point d'accès*, et des retransmissions sont aussi nécessaires afin d'assurer la bonne réception d'un message. Cependant les algorithmes d'adaptation de débits se basent sur les pertes afin de déterminer si une modulation plus robuste doit être utilisée pour assurer la bonne transmission d'un message.

Dans une seconde partie nous étudions les débits observés sur un *point d'accès* lorsque son nombre de *stations* augmente. Ces *stations* utilisent des modulations fixes, ainsi que l'algorithme d'adaptation de débits embarqué dans leurs carte 802.11. Nous observons que dans des conditions de contention l'utilisation de l'algorithme d'adaptation de débits donne des performances beaucoup plus faibles que l'utilisation d'un débit fixe.

Nous avons par ailleurs étudié les intervalles inter-trames des paquets reçus au niveau du *point d'accès*. Ceci nous a permis de trouver des anomalies dans les valeurs de fenêtres de contention ainsi que dans les files de priorités. Elles sont dûes à des erreurs d'implémentation au niveau des pilotes et des microcodes des cartes 802.11 étudiées.

4.1 Comportement de différentes implémentations de 802.11 lors de la perte de connectivité

4.1.1 Les retransmissions dans le standard

Lorsqu'un paquet est perdu, c'est-à-dire qu'aucun acquittement n'a été reçu par la station émettrice, celle-ci tente de récupérer l'erreur par une procédure de retransmission, que la perte soit dûe à une collision ou une dégradation du signal.

La procédure maintient plusieurs compteurs : SRC (*Short Retry Count*) et LRC (*Long Retry Count*) attachés à chaque trame, limités par les paramètres *dot11ShortRetryLimit=7* et *dot11LongRetryLimit=4*, ainsi que SSRC (*Station SRC*) et SLRC (*Station LRC*) globaux pour une station. La station maintient aussi la taille de la *Contention Window* CW dans $[aCWmin, aCWmax]$ (elle change en fonction de la modulation utilisée, $[15, 1023]$ pour les exemples). SRC et SSRC (respectivement LRC et SLRC) sont évalués et / ou incrémentés lors de chaque retransmission afin de savoir si le paquet doit être abandonné ou non. Cette procédure varie selon la taille de paquet :

Cas 1. Paquets "courts" (longueur ≤ seuil RTS/CTS) : on incrémente SRC et SSRC à chaque perte jusqu'à la limite de 7 et on augmente CW de manière exponentielle jusqu'à son maximum 1023 ; on remet

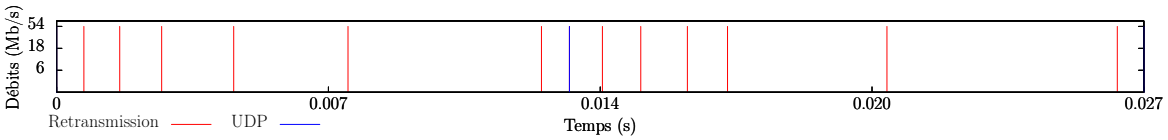


FIGURE 4.1 – Frise temporelle représentant l'émission des 2 premiers paquets par NS-3 lors de la perte de connectivité.

Carte	Type	Chipset	Pilote	Firmware	Type de MAC	
NETGEAR WN11V2	USB	AR9001U-NG*	`carl9170`	`carl9170`	SoftMAC	*AR9170 + AR9101
TP-LINK TL-WN722N	USB	AR9271	`ath9k_htc`	`htc_9271`	SoftMAC	
NEXUS 7 (2012)	Intégrée	BCM4330	`wl0`	`bcmdhd`	HardMAC	

TABLEAU 4.2 – Description des différentes cartes testées

ensuite SRC à 0, CW à 15 et le paquet est abandonné. Pour le paquet suivant, SSRC est à 7 et on recommence l'augmentation de SRC, SSRC et CW pour les pertes suivantes jusqu'à atteindre SRC=7 et CW=1023, et abandonner. Dans cet état, SSRC dépasse 7, alors CW n'est plus remis à 0 lors des échecs suivants. Pour la transmission des paquets suivants, on remet SRC à 0, on incrémente SRC et SSRC à chaque perte jusqu'à SRC=7 et l'abandon, et CW reste à la valeur maximale de 1023. Pour une suite de 3 paquets à transmettre, on devrait observer la séquence suivante :

	paquet 1							paquet 2							paquet 3						
SRC	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
SSRC	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
CW	15	31	63	127	255	511	1023	15	31	63	127	255	511	1023	1023	1023	1023	1023	1023	1023	1023

Cas 2. Paquets "longs" (longueur > seuil RTS/CTS) : ils sont précédés d'un échange des paquets RTS/CTS considérés comme courts, donc si un RTS est perdu, la station se comporte suivant le Cas 1. Pour le paquet de données en cas de succès du RTS/CTS, le comportement est similaire au Cas 1 avec LRC et SLRC.

À titre indicatif, la FIGURE 4.1 correspond à la politique de retransmission de NS-3 pour les 2 premiers paquets suivant la perte du lien. On remarque que CW est remise à zéro à la sixième retransmission tel que décrit dans le standard, néanmoins, NS-3 traite le 3-ème paquet et les suivants de la même manière contrairement aux indications du standard (aucune distinction n'est faite entre SRC/LRC et SSRC/SLRC).

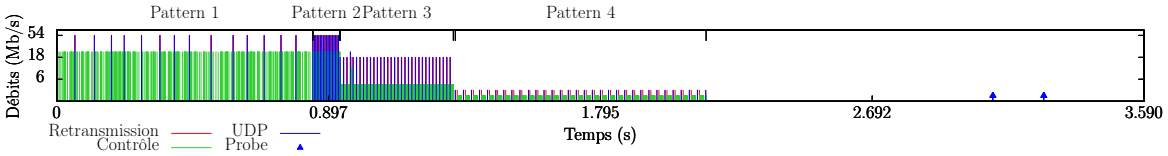
4.1.2 Les retransmissions dans la pratique

Afin d'observer la politique de retransmissions mise en place par différentes cartes lorsque celles-ci perdent la connectivité, nous avons mis en place le protocole expérimental suivant : 1) on associe l'interface testée avec un point d'accès 802.11G que nous maitrisons, 2) on sature le lien en générant du trafic UDP (paquets de 1480 octets) sans politique particulière vers le point d'accès à l'aide d'`ipmt` [160], 3) on arrête brutalement le point d'accès afin de s'assurer que celui-ci n'envoie ni *Deauthentication* ni *Disassociation Frame*. Ce scénario est effectué sur un canal ne présentant aucun trafic parasite et nous capturons l'ensemble des messages émis par la station à l'aide d'une autre machine équipée d'une interface Wi-Fi en mode moniteur.

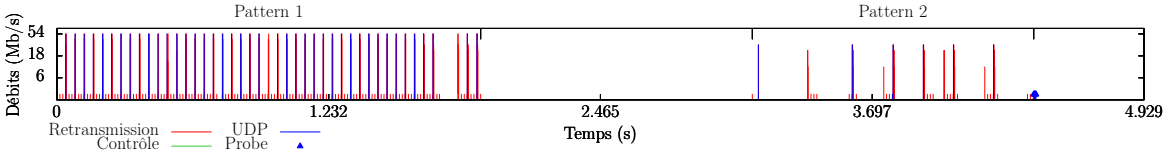
Nous avons testé les cartes détaillées dans la Table 4.2, 5 fois pour les cartes NETGEAR et la NEXUS 7, et une dizaine de fois pour la carte TP-LINK. Les cartes TP-LINK et NETGEAR se basent sur des chipsets ATHEROS utilisant des drivers et firmwares open source tandis que la NEXUS 7 utilise un chipset BROADCOM répandu. Le point d'accès est opéré par *hostapd v2.7* sur une machine Intel *Ubuntu 16.04* avec un noyau LINUX 4.4.

La FIGURE 4.2 présente le comportement des cartes étudiées à partir du moment où le point d'accès est éteint jusqu'au moment où les cartes arrêtent d'émettre des paquets retransmis. Les mêmes comportements ont été observés pour l'ensemble des mesures effectuées dans les mêmes conditions.

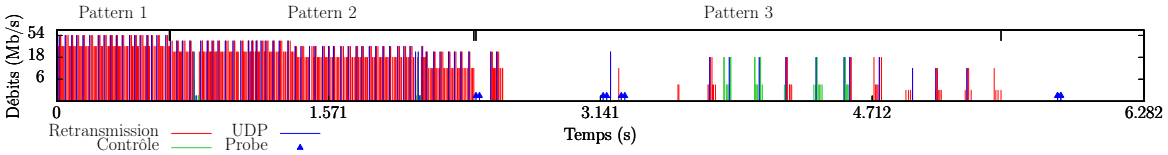
La FIGURE 4.3 présente la distribution des *backoffs* calculés à partir des temps inter-trames. Ces valeurs ont été obtenues en capturant les temps inter-trames lorsque le *point d'accès* est coupé, auxquels on enlève la valeur DIFS (Intervalle inter-trames utilisé par la fonction de coordination distribuée — *DCF Interframe Space*). On comptabilise alors le nombre de valeurs observées dans chacune des fenêtres de contention



(a) TP-LINK TL-WN722N



(b) NETGEAR WN11v2



(c) BROADCOM BCM4330

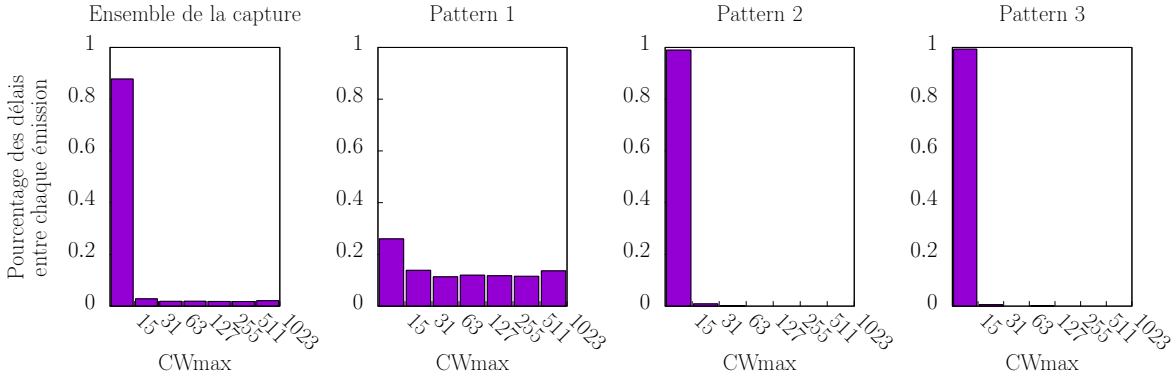
FIGURE 4.2 – Frises temporelles des retransmissions lors de la perte de connectivité pour différentes cartes — les pics bleus correspondent à l'émission d'un paquet UDP, les rouges à des retransmissions, les verts à l'émission d'un message de contrôle tel que RTS et les flèches marquent les *Probe Requests*

théoriques. La FIGURE 4.3b présente la distribution attendue pour 10 retransmissions conformément au standard. En comparant le comportement des cartes à cette implémentation du standard, nous pouvons en déduire leurs dérives par rapport à celui-ci.

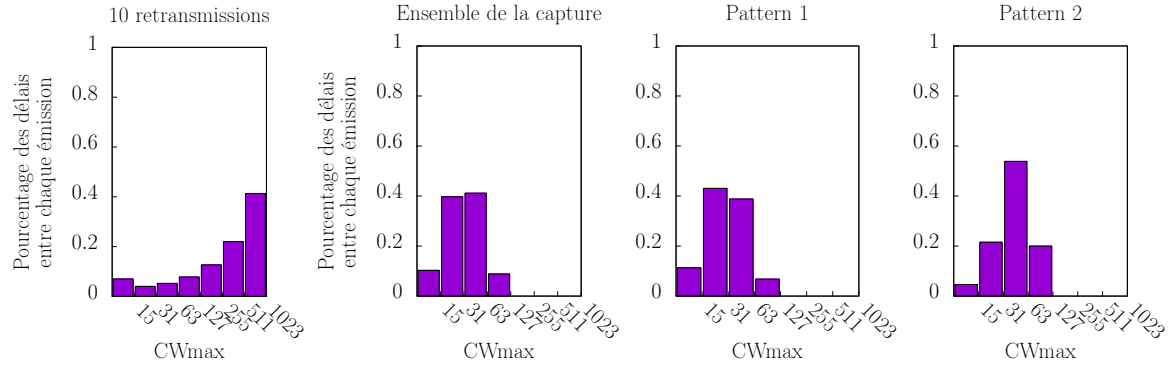
Analyse. De manière macroscopique on remarque que 1. les cartes présentent toutes une période pendant laquelle le nombre de retransmissions est important avant de passer à une période pendant laquelle les retransmissions sont plus espacées ; 2. les cartes TL-WN722N et BCM4330 passent progressivement vers des modulations plus robustes alors que la WN11v2 garde le même débit ; 3. les délais avant abandon varient d'une carte à l'autre, allant de 3 à 6 secondes.

TP-Link TL-WN722N. Les retransmissions de cette carte peuvent être séparées en quatre schémas (patterns) distincts. La FIGURE 4.4c présente les retransmissions effectuées pour un paquet lors du premier pattern. Elles consistent en une unique retransmission à 54 Mb/s du paquet UDP non acquitté suivi de 30 RTS envoyés à 24 Mb/s. CW est doublée à chaque émission et réinitialisée tous les 10 paquets. On observe une distribution des délais inter-trames quasiment uniforme sur la FIGURE 4.3a. Le deuxième pattern, illustré sur la FIGURE 4.4d, se comporte comme le premier sauf que CW semble rester à la valeur de *aCWmin* comme nous pouvons le voir sur la FIGURE 4.3a. Le troisième pattern agit comme le deuxième en réduisant le débit d'émission des paquets UDP à 18 Mb/s et des RTS à 5,5 Mb/s — FIGURE 4.4e. Le dernier pattern réduit le débit des paquets UDP à 2 Mb/s et des RTS à 1 Mb/s tout en ayant la même politique que les 2 patterns précédents pour les délais entre émissions. En n'augmentant pas la valeur de CW, les patterns 2, 3 et 4 contreviennent totalement au standard. Pour l'ensemble des patterns, la carte effectue plus de retransmissions que les six décrites dans le standard, et la majorité des retransmissions sont constituées d'un RTS et non du paquet original.

Netgear WN11v2. Cette carte présente 2 patterns de retransmissions. Le premier consiste pour la majorité des cas en une émission à 54 Mb/s d'un paquet UDP non acquitté suivi de 7 retransmissions comme illustré sur la FIGURE 4.4a. Ces différents débits correspondent aux valeurs retournées par l'algorithme d'adaptation de débits Minstrel. Minstrel exploite le taux de réussite des précédentes transmissions sur différentes modulations afin de proposer 4 modulations à essayer lors de la transmission courante : le meilleur débit observé (ici la première transmission à 54 Mb/s), le second meilleur débit (les 3 transmissions à 48 Mb/s), la modulation ayant la plus haute probabilité de succès (la transmission à 54 Mb/s) et la modulation la plus robuste (les 3 transmissions à 1 Mb/s).

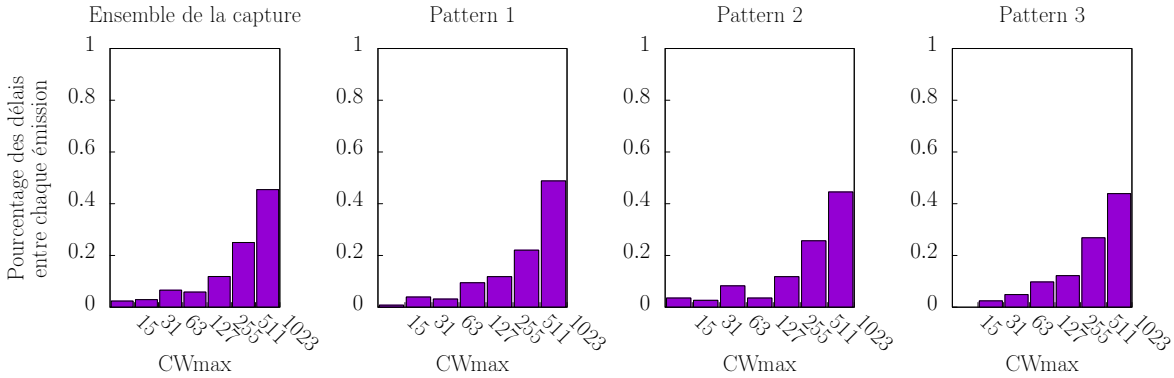


(a) TP-LINK TL-WN722N



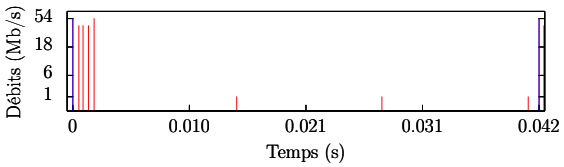
(b) Simulation R, 10 paquets émis

(c) NETGEAR WN11V2

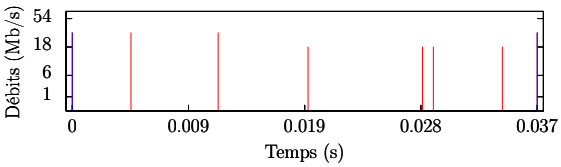


(d) BROADCOM BCM4330

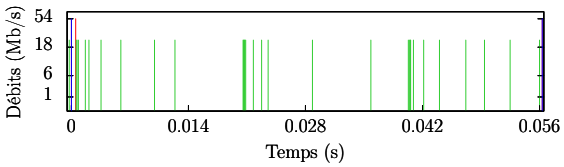
FIGURE 4.3 – Distribution des délais entre chaque transmission pour les différents patterns étudiés. Ces délais sont agrégés suivant les valeurs de CW



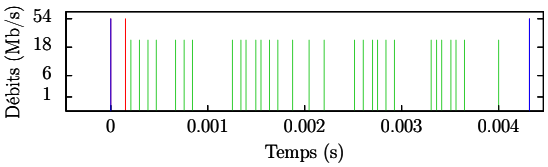
(a) NETGEAR WN11V2, pattern 1



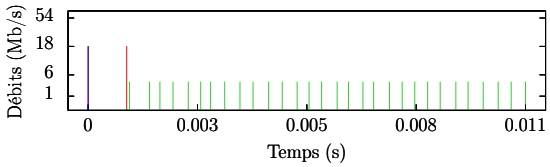
(b) BROADCOM BCM4330, pattern 2



(c) TP-LINK TL-WN722N, pattern 1



(d) TP-LINK TL-WN722N, pattern 2



(e) TP-LINK TL-WN722N, pattern 3

FIGURE 4.4 – Détail d'une retransmission pour certains patterns de différentes cartes

Cette carte effectue une retransmission de plus que ce que propose le standard. Le deuxième pattern est plus chaotique. Certains paquets sont marqués comme des retransmissions sans que l'on capture la transmission originale. La distribution des délais entre les émissions présentée FIGURE 4.3c indique que ce temps n'est pas uniforme et que CW n'est pas augmentée au delà de la valeur maximale de 127 slots.

Broadcom BCM4330. Cette carte présente 3 patterns de retransmissions. Le premier pattern consiste en 6 retransmissions : 2 à 54 Mb/s et 4 à 36 Mb/s. Le deuxième pattern (FIGURE 4.4b) se comporte comme le premier avec des débits différents : 2 retransmissions à 48 Mb/s suivies de 4 à 36 Mb/s dans un premier temps, puis ces débits sont réduits à 36 Mb/s et 18 Mb/s puis à 24 Mb/s et 11 Mb/s. Le troisième pattern démarre avec l'émission de deux *Probe Request* — correspondant à une recherche active d'un nouveau point d'accès auquel s'associer liée à la détection de la perte de connectivité — puis présente un comportement chaotique. Comme pour la WN11v2, certains paquets sont marqués comme des retransmissions sans que l'on ait capturé le paquet original. La carte notifie le point d'accès qu'elle va passer en économie d'énergie en levant le drapeau correspondant dans certaines trames contenant des messages UDP ou des *Null Frames* envoyées à 1, 5.5, 6, 18 ou 24 Mb/s. Le passage en économie d'énergie explique le saut temporel entre les émissions dans le troisième pattern. La figure 4.3d montre que les délais entre les transmissions tendent à devenir très grands et correspondent à la répartition attendue d'après le standard lorsque CW est grande. Chaque paquet est retransmis 6 fois avant d'être abandonné comme proposé dans le standard.

4.2 Comportement des cartes TP-Link TL-WN722N lors de la contention

Nous nous intéressons dans cette partie au comportement des cartes TP-LINK TL-WN722N lorsque celles-ci se trouvent en situation de contention. Nous nous intéressons d'une part au débit observé au niveau du *point d'accès* lorsque ces cartes utilisent l'algorithme d'adaptation de débits embarqué (hwrca — *hardware rate control algorithm*), notamment car ces algorithmes se basent en général sur les pertes et retransmissions afin d'adapter les modulations. Il s'agit d'un algorithme embarqué dans le microcode des cartes, et non d'un algorithme fonctionnant dans le noyau de la machine hôte. D'autre part, nous nous intéressons aussi à l'évolution des intervalles inter-trames au niveau du *point d'accès*. De plus, ces cartes ont

l'avantage de proposer à la fois un pilote et un microcode *open source*.

4.2.1 Description de la plateforme de test

Nous avons déployé un banc d'essai orchestré par *WalT*, composé de 30 *Raspberry Pi*, chacun équipé d'un dongle TP-LINK TL-WN722N, et d'un APU équipé d'une carte 802.11 Atheros.

Lors du démarrage de l'expérience l'APU est redémarré et démarre un *point d'accès* en démarrant un démon `hostapd`. Le *point d'accès* est configuré pour fonctionner en 802.11G, en annonçant les modulations 6 Mb/s, 9 Mb/s, 12 Mb/s, 18 Mb/s, 24 Mb/s, 36 Mb/s, 48 Mb/s et 54 Mb/s, et opérant sur le canal 6 car il s'agit du canal le moins occupé dans la salle d'expérimentation et annonçant le support des WME (Extensions multimedia sans fil — *Wireless Multimedia Extensions*). Ensuite les *Raspberry Pi* sont redémarrés et s'associent au *point d'accès*. Un premier *Raspberry Pi* va commencer à saturer le lien en envoyant des paquets UDP de manière continue vers le *point d'accès* à l'aide de l'outil `udpmt` — intégré dans `ipmt` [160]. Les paquets générés correspondent à du trafic dont la catégorie d'accès a une priorité BE. Toutes les 60*s* un nouveau *Raspberry Pi* tentera lui aussi de saturer le lien selon le même procédé et jusqu'au moment ou l'ensemble des *Raspberry Pi* seront en train d'accéder au canal en concurrence.

Afin de mesurer le comportement des cartes dans différentes conditions lors de la contention, ces mesures sont effectuées dans un premier temps à débit fixe, puis en laissant l'algorithme d'adaptation de débit intégré aux cartes. Afin de s'assurer que les cartes utilisent un débit fixe nous avons modifié le microcode pour se passer de l'algorithme d'adaptation de débit et employer une modulation fixée. L'ensemble des messages échangés sont capturés sur un *MacbookAir* de 2013 configuré en mode `monitor`. Les options de débogage du driver `ath9k` sont activées dans le noyau des LINUX déployés sur l'APU et les *Raspberry Pi* afin de pouvoir récupérer les informations du microcode et du driver concernant la réception et la transmission de paquets.

Le déploiement est dense, l'ensemble des *Raspberry Pi* et l'APU sont disposés sur deux étagères. De plus l'ensemble des équipements émettent à des puissances relativement faibles au vu de leurs proximités les uns des autres avec une puissance de 1 dBm pour les *Raspberry Pi* et de 3 dBm pour l'APU. *WalT* nous a permis d'aisément rejouer l'expérimentation avec des modulations différentes.

4.2.2 Analyse des débits observés

Nous avons effectué plusieurs mesures lors desquelles les modulations utilisées par l'ensemble des *Raspberry Pi* étaient changées.

Ainsi nous avons mesuré les débits au niveau du *point d'accès* avec l'ensemble des *Raspberry Pi* ayant leurs modulations fixées à 6 Mb/s, 24 Mb/s, 36 Mb/s, 48 Mb/s ou 54 Mb/s. Pour chacune de ces mesures, le trafic envoyé avait une priorité de type BE.

Nous avons aussi fait des mesures avec des priorités plus élevées lors desquelles les *Raspberry Pi* voyaient leur modulation fixée à 54 Mb/s, mais leur trafic se voyait attribué une priorité de type VI (54-vi) ou VO (54-vo). De la même manière l'ensemble des *Raspberry Pi* utilisaient la même configuration au cours de la mesure.

Enfin nous avons mesuré le débit lorsque les *Raspberry Pi* laissent leurs carte TP-LINK TL-WN722N utiliser leur algorithme d'adaptation de débit.

La FIGURE 4.5 représente le nombre d'octets reçus par le *point d'accès* chaque seconde en fonction des modulations employées.

Pour les modulations fixées à 24 Mb/s, 36 Mb/s, 48 Mb/s et 54 Mb/s — avec ou sans priorité de trafic —, on observe une augmentation du débit reçu lorsque le nombre de *stations* augmente jusqu'à 3. Ensuite lorsque le nombre de *stations* augmente, le débit observé diminue. Cependant même avec une trentaine de *stations* le débit reçu au niveau du *point d'accès* reste proche du débit de départ.

Ce n'est cependant pas le cas lorsque les cartes utilisent leur algorithme d'adaptation de débit (hwrca). On remarque alors que le débit total constaté au niveau du *point d'accès* chute dramatiquement. Ainsi lorsque le nombre de *stations* simultanées reste faible le débit observé est de l'ordre de celui atteignable lorsque l'ensemble de ces *stations* utilisent le débit maximum possible en 802.11G. Mais lorsque le nombre de *stations* augmente le débit chute jusqu'à atteindre celui observé lorsque l'ensemble des *stations* utilisent

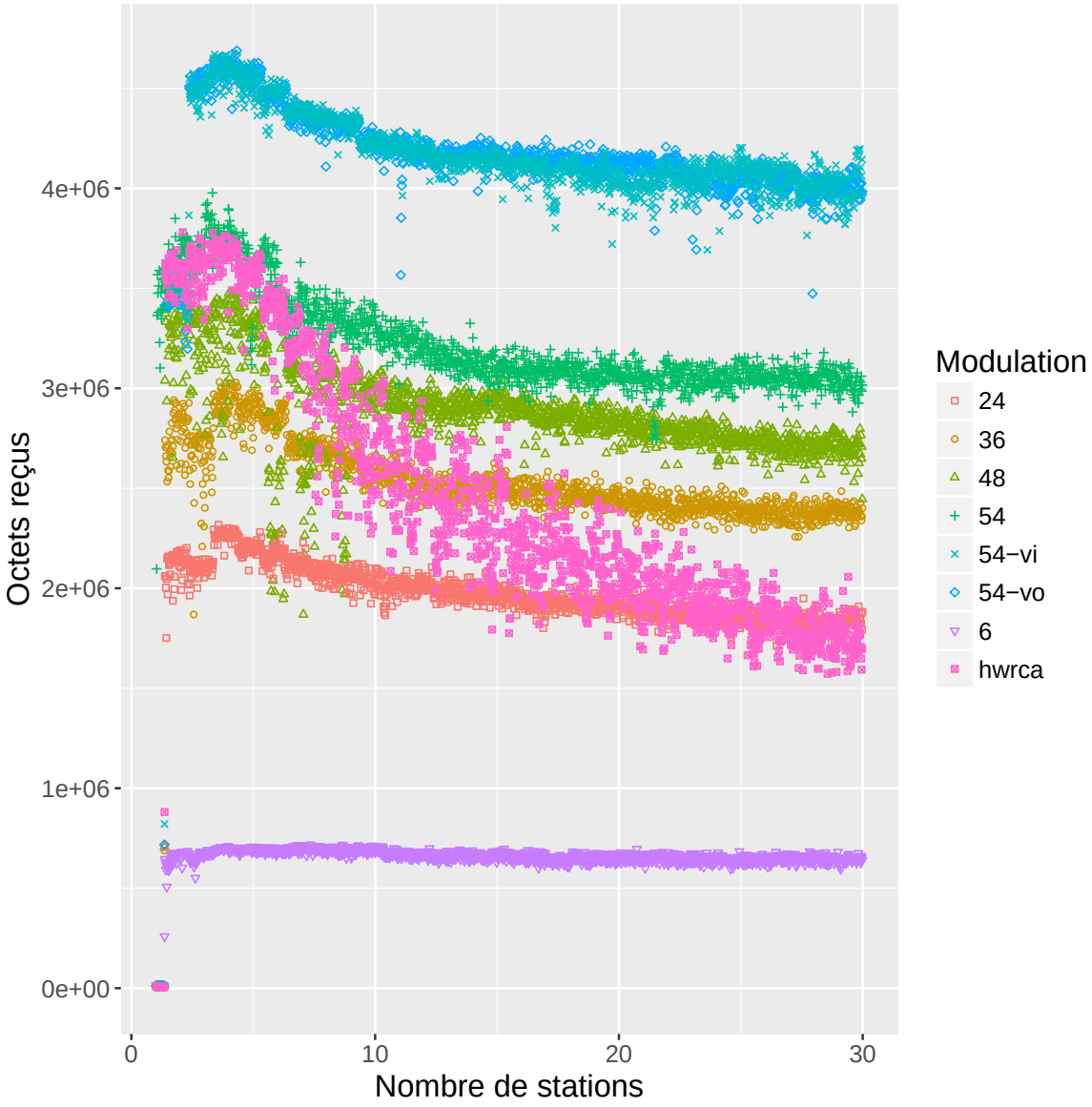


FIGURE 4.5 – Nombre d'octets reçus par seconde au niveau du *point d'accès* lors de la montée en charge du nombre de *stations* tentant de saturer le lien — une nouvelle *station* démarre son émission de paquets UDP toutes les 60s — pour différentes modulations fixes et l'algorithme d'adaptation de débit intégré dans le microcode des cartes testées (hwrca)

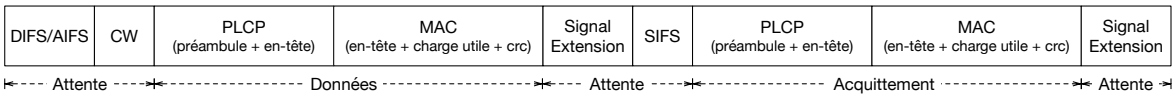


FIGURE 4.6 – Représentation des différentes étapes nécessaires à l'envoi d'un paquet de données et la réception de l'acquittement associé en 802.11G

Priorité	t_{slot}	AIFSN	IFS	CWMIN	CWMAX
AC_BK	$9\mu s$	7	$AIFS = SIFS + AIFSN * t_{slot} = 73\mu s$	$aCWmin = 15$	$aCWmax = 1023$
AC_BE	$9\mu s$	3	$AIFS = SIFS + AIFSN * t_{slot} = 37\mu s$	$aCWmin = 15$	$aCWmax = 1023$
AC_VI	$9\mu s$	2	$AIFS = SIFS + AIFSN * t_{slot} = 28\mu s$	$\frac{aCWmin+1}{2} - 1 = 7$	$aCWmin = 15$
AC_VO	$9\mu s$	2	$AIFS = SIFS + AIFSN * t_{slot} = 28\mu s$	$\frac{aCWmin+1}{4} - 1 = 3$	$\frac{aCWmin+1}{2} - 1 = 7$
Pas de WME	$9\mu s$		$DIFS = SIFS + 2 * t_{slot} = 28\mu s$	$aCWmin = 15$	$aCWmax = 1023$

TABLEAU 4.3 – Valeurs des intervalles inter-trames et des fenêtres de contention pour les différentes classes de priorités

une modulation fixée à 24 Mb/s, et est bien en dessous de celui observé lorsque l'ensemble des *stations* utilisent une modulation fixée à 54 Mb/s. Nous remarquons aussi de grande variations dans le nombre d'octets reçus lors de l'augmentation du nombre de *stations*.

Cette observation est liée à l'augmentation du nombre de collisions lorsque le nombre de *stations* augmente. Les *stations* voient ainsi leur nombre de messages non acquittés augmenter, ce phénomène pouvant être lié aux collisions mais aussi à une dégradation de la qualité du lien. Lorsque la qualité du lien se dégrade il faut changer de modulation afin d'en utiliser une plus robuste, au contraire lorsque les pertes sont liées aux collisions il ne faut pas changer de modulation — afin de ne pas monopoliser un lien déjà saturé plus longtemps — mais augmenter sa fenêtre de contention. Or il semblerait que l'algorithme d'adaptation de débit embarqué dans les cartes décide de baisser les modulations utilisées lorsque le nombre de collisions atteint un certain seuil, affectant alors le débit global.

En effet, lorsque l'on analyse son implémentation au sein du microcode, l'algorithme se base sur le PER (Taux d'erreur de paquets — *Packet Error Rate*) afin de déterminer la modulation maximale à utiliser. Lorsque le PER dépasse un certain seuil et que la modulation actuellement utilisée n'est pas la plus basse, alors la modulation directement inférieure à celle utilisée devient la modulation maximale. Afin de pouvoir réaugmenter le débit, l'algorithme essaie d'envoyer certains paquets avec des modulations plus élevées afin de voir si celles-ci redeviennent utilisable. Si un tel envoi réussi avec une seule retransmission au maximum, alors la modulation maximale devient la modulation utilisée.

4.2.3 Analyse des intervalles inter-trames

4.2.3.1 Comportement attendu selon le standard

Dans nos expériences, une *station* communique avec son *point d'accès* en envoyant des messages UDP de manière à saturer le lien. L'envoi d'un message et de son acquittement au niveau MAC se décompose de la manière illustrée sur la FIGURE 4.6.

Avant d'envoyer un paquet de données la *station* doit s'assurer que le canal est libre pendant une durée DIFS si les extensions WME relatives à la QoS — basée sur EDCA — ne sont pas activées, sinon il s'agit d'une durée AIFS (Intervalle inter-trames d'arbitrage — *Arbitration Interframe Space*) variable, dépendant de la priorité associée à la donnée à transmettre.

La *station* doit ensuite attendre un nombre de slots correspondant à la valeur de sa fenêtre de contention pendant lequel le canal reste libre. Les durées des AIFS ainsi que les différentes valeurs de fenêtres de contention indiquées dans le standard en fonction des différentes classes de priorités de trafic sont rappelées sur le TABLEAU 4.3. La transmission du paquet de données peut alors commencer avec l'envoi du préambule et de l'entête PLCP (Protocole de convergence de la couche physique — *Physical Layer Convergence Protocol*) suivi de la trame MAC à proprement parler. Après l'émission de la trame un intervalle de silence d'une durée *Signal Extension* doit être respecté.

La durée *Signal Extension* est spécifique aux trames 802.11 utilisant la couche physique ERP-OFDM

(Extension de la couche physique afin de supporter des modulations OFDM — *Extended Rate Physical* OFDM). Cette extension vaut $6\mu s$ et est utilisée afin de permettre aux équipements recevant des trames ERP-OFDM d'avoir le temps nécessaire au décodage convolutif. Cette durée est respectée au niveau de la couche physique en l'intégrant au calcul de la durée d'occupation du canal après avoir décodé un préambule ERP-OFDM. Elle est aussi considérée au niveau de la couche MAC en l'intégrant au champ *Duration/ID* indiquant au mécanisme de détection de porteuse virtuelle que le canal est occupé. Lors de l'envoi d'une trame ce champ est rempli avec le temps en microsecondes nécessaire à l'attente d'un SIFS (Intervalle inter-trames court — *Short Interframe Space*) et à l'envoi d'une trame d'acquittement. Lorsque ERP-OFDM est utilisée, $6\mu s$ sont ajoutées à ces valeurs. Une *station* ayant ainsi décodé un message ne lui étant pas destiné n'a pas à effectuer de CCA durant cette période.

Ensuite, la procédure nécessaire à l'envoi de la trame d'acquittement est entamée. Celle-ci commence avec l'écoute du canal par le *point d'accès* pendant une durée SIFS. La transmission du message d'acquittement est alors effectuée selon la même procédure que pour un paquet de données.

Pour l'émission de trafic utilisant EDCA dans un environnement sans collisions, l'intervalle de temps entre chaque trame acquittée vaut :

$$IFS = SignalExtension + SIFS + (AIFSN + CW) * 9\mu s$$

Avec CW compris entre 0 et $aCWmin$, ce qui nous permet de déterminer les valeurs que peuvent prendre ces intervalles pour chaque classe de priorité :

- 16 valeurs comprises entre $79\mu s$ et $214\mu s$ pour AC_BK
- 16 valeurs comprises entre $43\mu s$ et $178\mu s$ pour AC_BE
- 8 valeurs comprises entre $34\mu s$ et $97\mu s$ pour AC_VI
- 4 valeurs comprises entre $34\mu s$ et $61\mu s$ pour AC_VO

Enfin, lorsque les extensions WME ne sont pas activées, l'intervalle de temps entre chaque trame acquittée vaut :

$$IFS = SignalExtension + DIFS + CW * 9\mu s$$

Il n'y alors plus de classes de priorité, et nous devrions obtenir 16 valeurs d'intervalles différentes comprises entre $34\mu s$ et $169\mu s$.

4.2.3.2 Comportement des cartes TP-Link TL-WN722N

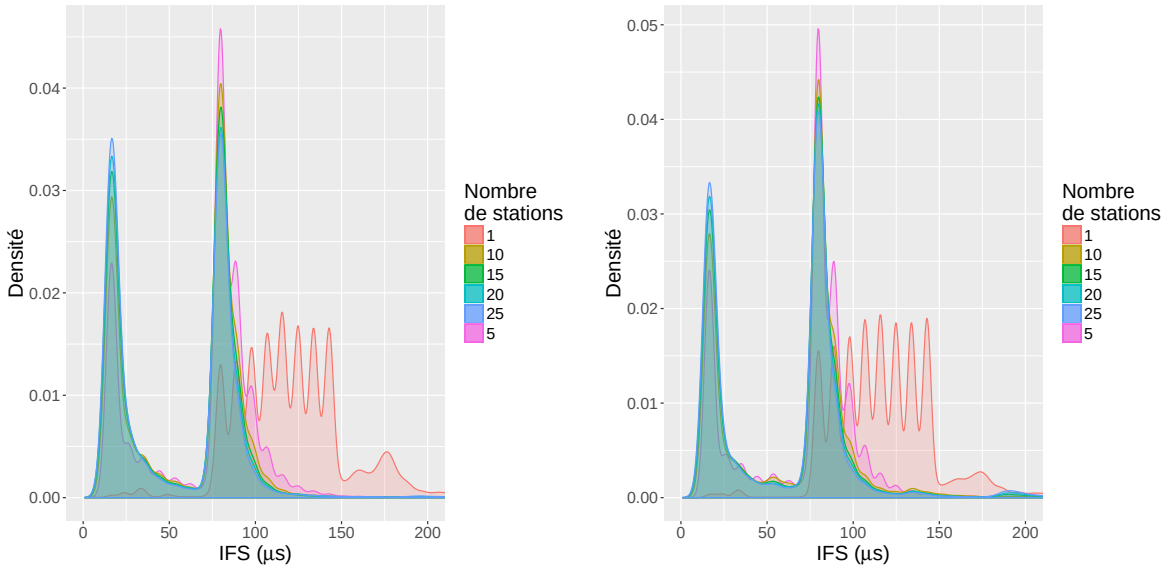
Nous avons mesuré les intervalles inter-trames au niveau du *point d'accès* lors de l'augmentation du nombre de *stations*. La répartition de ceux-ci, lorsque l'ensemble des *Raspberry Pi* utilisent une modulation de 54 Mb/s, en fonction du nombre de *stations* est illustrée sur la FIGURE 4.7.

Nous observons que lorsque le nombre de *stations* augmente, les intervalles inter-trames au niveau du point d'accès ont tendance à se concentrer. En effet lorsqu'une unique *station* est présente, ces intervalles sont équitablement distribués entre $79\mu s$ et $143\mu s$.

Lorsque le nombre de *stations* augmente, la quantité d'intervalles inter-trames les plus longs diminue tandis que celle des plus courts augmente, jusqu'à avoir une concentration quasi exclusive d'intervalles valant $79\mu s$ lorsque le nombre de *stations* devient important. Ce phénomène est dû au fait que lorsque le nombre de *stations* augmente, il y a de plus fortes chances qu'une *station* ait une valeur de fenêtre de contention à zéro après l'émission d'une trame. En d'autres termes, il y a de faibles chances que toutes les *stations* aient une fenêtre de contention supérieure à 0.

On remarque aussi une augmentation du nombre de transmissions intervenant avec un intervalle inter-trame de $16\mu s$. Comme nous l'avons vu lors de l'analyse des retransmissions, les cartes TP-LINK TL-WN722N envoient des messages RTS pour regagner l'accès au canal. Or, l'augmentation du nombre de *stations* augmente le nombre de collisions, et donc le nombre de retransmissions. Lorsqu'un message CTS est envoyé par le *point d'accès* après qu'une *station* ait réussi à transmettre un RTS, elle doit envoyer son message après une durée SIFS à laquelle s'ajoute la durée *Signal Extension*, expliquant ainsi l'inter-trame observé de $16\mu s$.

Cependant l'analyse des intervalles inter-trames laisse entrevoir des anomalies. Lorsqu'une seule *station* est présente, nous observons notamment que le nombre d'intervalles distincts est de 8, ce qui ne



(a) Débit fixé à 54 Mb/s pour l'ensemble des *stations*

(b) Utilisation de l'algorithme d'adaptation de débit intégré aux cartes

FIGURE 4.7 – Répartition des IFS en réception au niveau du *point d'accès* en fonction du nombre de *stations* en concurrence

semble pas être en adéquation avec une fenêtre de contention de 15. De la même manière, le premier intervalle se situe à 79μs, ce qui n'est pas normal pour du trafic envoyé avec une priorité BE.

4.2.4 Étude des anomalies concernant les intervalles inter-trames

Nous avons décidé d'étudier dans le détail les intervalles inter-trames lorsqu'une unique *station* émet des messages afin de comparer le comportement des cartes lors de l'accès au canal par rapport au standard.

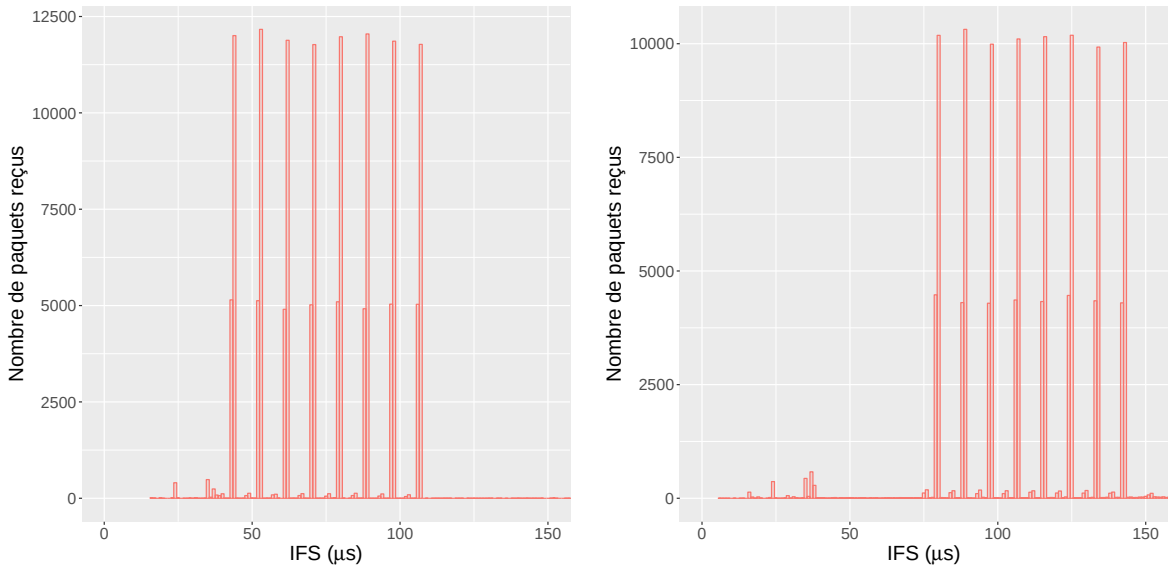
Nous avons mesuré les intervalles inter-trames pour les différentes classes de priorité avec les cartes TP-LINK TL-WN722N et ils sont présentés sur la FIGURE 4.8. Ces mesures ont été effectuées avec une unique carte associée à un *point d'accès* sur un canal inoccupé. Les graphiques représentent le nombre de trames reçues par intervalle inter-trame en μs.

On remarque que pour les messages envoyés avec la file de priorité AC_BK — FIGURE 4.8a —, les envois sont majoritairement envoyés à huit « moments » différents. Il s'agit de groupement d'IFS de 2μs : [43μs;44μs], [52μs;53μs], [61μs;62μs], [70μs;71μs], [79μs;80μs], [88μs;89μs], [97μs : 98μs] et [106μs;107μs]. Les premières valeurs de chaque couple correspondent à celles dérivées du standard, cependant la majorité des émissions de trames ont un IFS avec un retard d'une microseconde. Ce retard est probablement dû à une incertitude dans la mesure ou un temps de traitement avant l'envoi légèrement plus long.

De la même manière les messages envoyés avec une priorité AC_BE — FIGURE 4.8b — sont groupés en 8 couples de 2 IFS : [79μs;80μs], [88μs;89μs], [97μs;98μs], [106μs;107μs], [115μs;116μs], [124μs;125μs], [133μs;134μs] et [142μs;143μs]. On observe ici aussi que la majorité des envois ont un IFS retardé de une microseconde par rapport au standard.

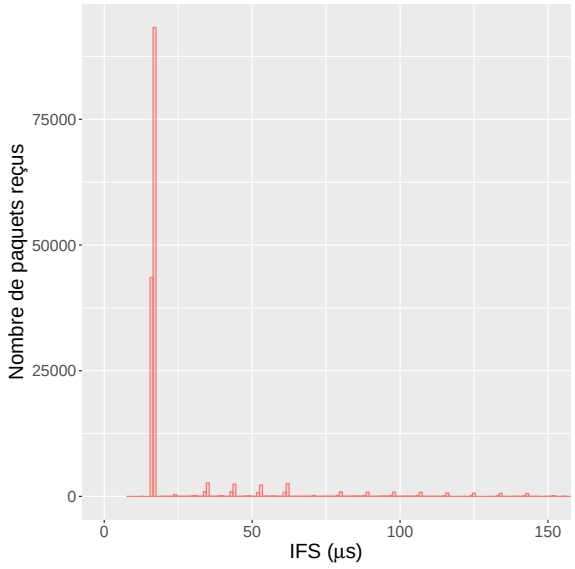
Concernant ces deux files de priorités, nous remarquons que les valeurs des IFS observées ne concordent pas avec les valeurs théoriques dérivées du standard. Ces deux files semblent inversées car nous observons un IFS minimal de 43μs au lieu de 79μs pour la file AC_BK et un IFS de 79μs au lieu de 43μs pour la file AC_BE. Nous observons aussi que le nombre de valeurs d'IFS observées est inférieur au nombre de valeurs possibles dans le standard. Ces deux files devraient avoir une fenêtre de contention initiale de quinze et nous devrions alors observer seize valeurs d'IFS distinctes, or nous n'observons que huit valeurs d'IFS distinctes. Cette observation laisse supposer que la fenêtre de contention initiale utilisée est deux fois plus petite que celle décrite dans le standard.

Nous pouvons aussi remarquer des émissions de paquets beaucoup moins importantes avec une

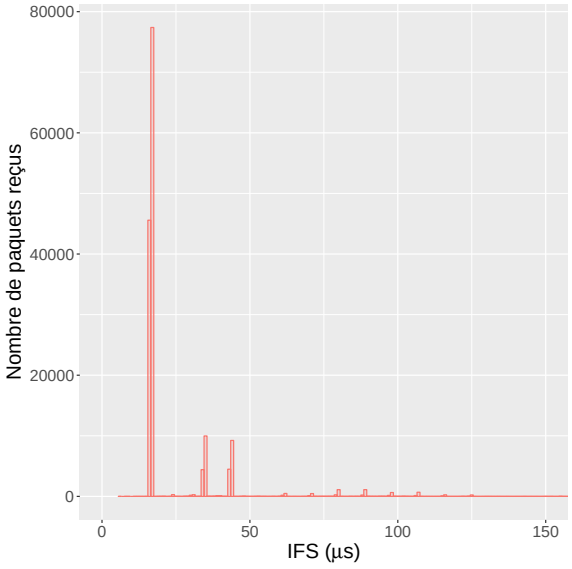


(a) BK

(b) BE



(c) VI



(d) VO

FIGURE 4.8 – Répartition des IFS en réception au niveau du *point d'accès* pour les différentes queues EDCA avec une seule *station* saturant le lien

```
@@ -695,8 +695,8 @@ static void ath_tgt_txq_setup(struct ath_softc_tgt *sc)
     sc->sc_uapsdq   = &sc->sc_txq[UAPSDQ_NUM];
     sc->sc_cabq     = &sc->sc_txq[CABQ_NUM];

-    sc->sc_ac2q[WME_AC_BE]  = &sc->sc_txq[0];
-    sc->sc_ac2q[WME_AC_BK]  = &sc->sc_txq[1];
+    sc->sc_ac2q[WME_AC_BK]  = &sc->sc_txq[0];
+    sc->sc_ac2q[WME_AC_BE]  = &sc->sc_txq[1];
     sc->sc_ac2q[WME_AC_VI]  = &sc->sc_txq[2];
     sc->sc_ac2q[WME_AC_VO]  = &sc->sc_txq[3];
```

FIGURE 4.9 – Modification du microcode concernant l'inversion des files AC_BK et AC_BE

```
@@ -195,7 +195,7 @@ int ath_htc_txq_update(struct ath9k_htc_priv *priv, ...);

     qi.tqi_aifs = qinfo->tqi_aifs;
-    qi.tqi_cwmin = qinfo->tqi_cwmin / 2; /* XXX */
+    qi.tqi_cwmin = qinfo->tqi_cwmin; /* XXX */
     qi.tqi_cwmax = qinfo->tqi_cwmax;
     qi.tqi_burstTime = qinfo->tqi_burstTime;
     qi.tqi_readyTime = qinfo->tqi_readyTime;
```

FIGURE 4.10 – Modification du pilote `ath9k_htc` concernant les tailles des fenêtres de contention

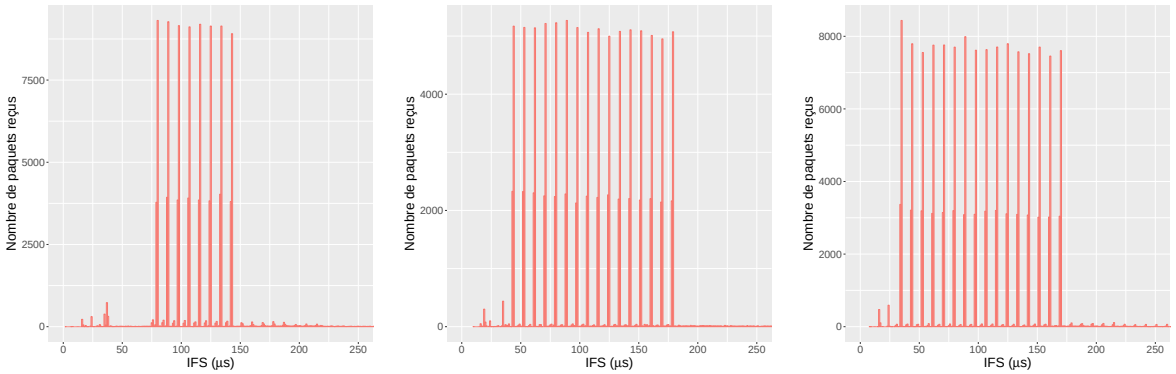
avance de $3\mu s$ par rapport aux IFS réglementaires. Lors de l'analyse du trafic nous constatons que ces avances interviennent après la réception de paquets à 1 Mb/s tels que les *beacons*. Nous n'avons pas trouvé d'explications relatives à ces avances dans le standard.

Concernant les files de priorités AC_VI — FIGURE 4.8c — et AC_VO — FIGURE 4.8d — nous observons les mêmes anomalies concernant les fenêtres de contention utilisées. En effet pour la file AC_VI nous observons quatre valeurs au lieu de huit et pour la file AC_VO nous observons deux valeurs au lieu de quatre. Les valeurs de départ des IFS sont cependant bien respectées avec un IFS minimal de $34\mu s$ pour ces deux files. Il est important de constater un nombre très important de paquets émis avec un IFS de $16\mu s$ ou $17\mu s$. Ceci est dû au fait que les files de priorité AC_VI et AC_VO utilisent un mécanisme de TXOP. Lorsqu'une *station* envoie un paquet d'une de ces deux files et que celui-ci est acquitté, celle-ci gagne un TXOP pendant lequel le canal lui est réservé pendant une durée déterminée pour envoyer d'autres paquets qui seraient actuellement en attente sur la même file. Cette réservation de canal est mise en pratique par la possibilité pour la station d'accéder au canal après une attente valant SIFS au lieu de AIFS et de la fenêtre de contention après chaque acquittement. La durée pendant laquelle une *station* possède un accès prioritaire au canal vaut au maximum $4.096ms$ pour AC_VI et $2.080ms$ pour AC_VO quand la couche ERP-OFDM est utilisée.

Le driver LINUX et le microcode utilisés dans les cartes TP-LINK TL-WN722N étants ouvert, nous les avons analysés afin de déterminer l'origine des anomalies observées. Nous avons ainsi mis en évidence que les files de priorité AC_BK et AC_BE sont inversées au niveau du microcode exécuté sur la carte comme illustré sur la FIGURE 4.9. Dans le noyau LINUX les files de priorités BK, BE, VI et VO sont stockées dans un tableau respectivement aux index 0, 1, 2 et 3. Or lorsque le microcode initialise l'association entre les files de priorité internes à la carte et celles du noyau on remarque que la file AC_BE utilise l'index 0 et AC_BK l'index 1. Afin de régler ce problème il suffit d'inverser cette initialisation au sein du microcode.

Les anomalies concernant les fenêtres de contention ayant des valeurs initiales valant la moitié de celles définies dans le standard sont liées à l'initialisation de celles-ci dans le pilote `ath9k_htc` comme illustré sur la FIGURE 4.10. On constate ici que les valeurs de *aCWmin* utilisées dans le pilote et fournies à la carte sont les valeurs de *aCWmin* fournies par le noyau mais divisées par deux. Il faut noter ici qu'il s'agit peut être d'un *bugfix* ancien, ou d'une adaptation temporaire pour un microcontroleur particulier. La présence du commentaire `/* XXX */` laisse supposer que cette division ne devait être que temporaire.

Nous avons réeffectué les mesures précédentes avec la modification du microcode ainsi que la modifi-



(a) Microcode de base

(b) Après les modifications du microcode et du pilote concernant les fenêtres de contention

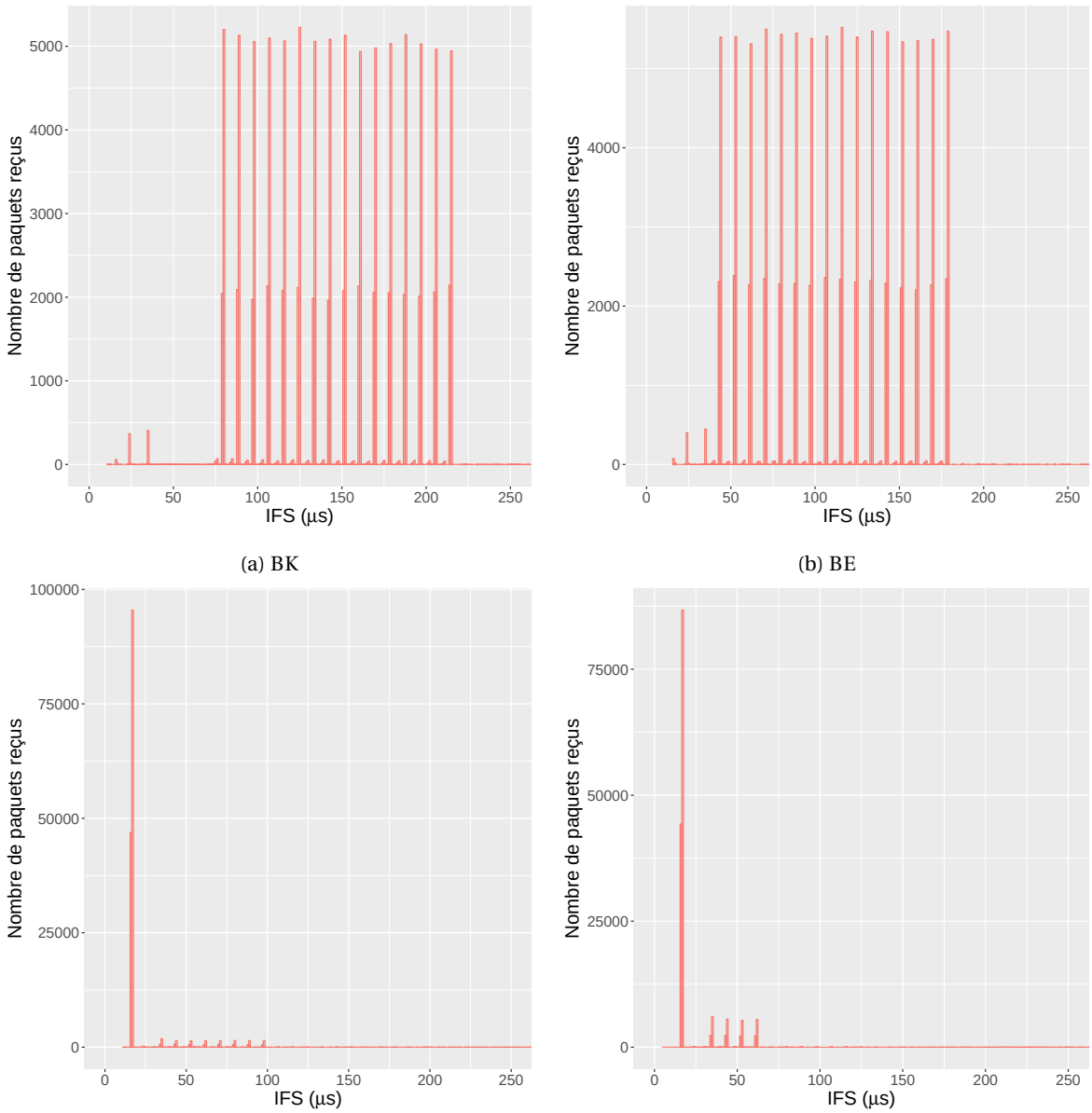
(c) Après les modifications du microcode et du pilote afin de prendre en compte le fait qu'un *point d'accès* n'annonce pas de support WME

FIGURE 4.11 – Répartition des IFS en réception au niveau du *point d'accès* lorsque le point d'accès n'annonce pas de support pour les WME

cation du pilote et les résultats sont présentés sur la FIGURE 4.12. Nous pouvons constater que les valeurs initiales des fenêtres de contention sont conformes aux valeurs spécifiées dans le standard. Nous observons que les paquets étant dans les files de priorité AC_BK et AC_BE sont émis lors de 16 valeurs distinctes d'IFS correspondant à une fenêtre de contention de 15. De la même manière nous observons que la file AC_VI a maintenant une fenêtre de contention de 7 et la file AC_VO une fenêtre de contention de 3. Finalement les valeurs des AIFSN pour les files AC_BK et AC_BE sont conformes au standard car les émissions sont effectuées respectivement entre 79μs et 214μs et entre 43μs et 178μs.

Enfin nous avons effectué les mêmes tests lorsque le *point d'accès* n'annonçait pas de support WME. Les résultats sont illustrés sur la FIGURE 4.11. Lorsqu'aucune des modifications précédentes n'est présente nous constatons que les cartes utilisent la file AC_BE, avec les écarts au standard précédemment énoncés. Après l'application des modifications concernant la taille des fenêtres de contention et l'inversion des files AC_BK et AC_BE, les cartes continuent d'utiliser EDCA avec une valeur $AIFSN = 3$ au lieu d'utiliser $DIFS$ — ou $AIFSN = 2$ — avant de décrémenter leur fenêtre de contention.

Une modification afin de rajouter une vérification de la présence des extensions WME lors de la détermination des différentes valeurs de AIFSN à utiliser au moment où le driver initialise les files permet de résoudre ce décalage. Les valeurs que peuvent prendre les intervalles inter-trames commencent alors à 34μs comme l'illustre la FIGURE 4.11c.



(a) BK

(b) BE

(c) VI

(d) VO

FIGURE 4.12 – Répartition des IFS en réception au niveau du *point d'accès* pour les différentes queues EDCA avec une seule *station* saturant le lien après avoir modifié le microcode et le pilote `ath9k_htc`

Conclusions

Lorsqu'un équipement mobile s'éloigne du *point d'accès* auquel il est associé, la qualité du lien se dégrade et le nombre de paquets perdus augmente. Le mécanisme de retransmission entre alors en jeu pour tenter de récupérer des pertes. Cependant, lorsque les retransmissions ne permettent plus d'assurer une connectivité avec le *point d'accès*, elles signifient que l'équipement mobile doit changer de *point d'accès*, et initier un *handover*.

Dans ce chapitre, nous avons vu que les politiques de retransmission varient grandement d'une carte à une autre en terme de débits choisis et de temps d'attente entre les transmissions. Le standard précise que la remise à zéro de la fenêtre de contention doit être effectuée, soit lors de la bonne transmission d'un paquet, soit lorsque SSRC vaut 7, or NS-3 et les cartes NETGEAR WN11V2 et TP-LINK TL-WN722N ne respectent pas cette contrainte.

Concernant le nombre de retransmissions, nous observons que les cartes NETGEAR WN11V2 et TP-LINK TL-WN722N retransmettent un nombre de fois plus important que ce que spécifie le standard. La carte TP-LINK TL-WN722N a de plus une politique très agressive concernant l'accès au canal : elle transmet de nombreux RTS sur une courte durée. L'utilisation de RTS dans le cas d'une retransmission permet d'éviter d'envoyer des messages longs qui pourraient ne pas être reçus, limitant ainsi l'occupation sur le canal et les risques de collisions grâce au mécanisme de RTS/CTS.

Seule la carte BROADCOM BCM4330 semble respecter le standard tant en terme d'augmentation de sa fenêtre de contention qu'en nombre de retransmissions avant d'abandonner le message.

Finalement, l'ensemble des cartes étudiées retransmettent un grand nombre de fois lors de la perte de connectivité, ce qui impacte les performances globales du réseau avec une monopolisation du canal « inutile », mais aussi les performances au niveau de la carte avec une augmentation du temps avant l'abandon et la recherche d'un nouveau point d'accès.

Ce constat nous amène à penser qu'il est possible de diminuer le nombre de retransmissions afin d'améliorer le *handover*.

Nous avons ensuite étudié le comportement des cartes TP-LINK TL-WN722N dans un contexte de contention, dans lequel nous avons un déploiement dense d'une trentaine de cartes autour d'un *point d'accès* unique. Cette étude nous permet d'analyser l'impact des collisions ainsi générées sur l'algorithme d'adaptation de débit de ces cartes, en particulier lorsque la qualité du lien est détériorée par la quantité de trafic environnant — la contention —, et non par un éloignement de l'équipement sans fil. Dans ces conditions, nous avons observé que l'algorithme d'adaptation de débit n'est pas en mesure de déterminer qu'il est dans un contexte de contention — et non d'un bilan de liaison dégradé — et fait changer la modulation de la carte afin d'utiliser une modulation plus robuste , ce qui ne permet pas d'améliorer les performances en contention.

En effet, dans un contexte de contention dans lequel la qualité du lien est bonne, il ne faut pas changer de modulation pour une modulation plus robuste, mais plutôt augmenter ses fenêtres de contention afin de réduire les probabilités de collisions. Utiliser une modulation permettant un débit moins élevé ne permettra pas d'éviter les collisions et aura l'effet pervers de monopoliser le canal plus longtemps lorsque cette station y gagne l'accès, impactant les performances globales du réseau.

A l'inverse lorsque la qualité du lien se dégrade il ne faut pas forcément augmenter les fenêtres de contentions car le « non acquittement » d'un message envoyé n'est pas dû à une collision mais à la présence d'interférences. Il faut donc diminuer au plus vite le débit afin d'utiliser une modulation plus robuste et éviter des retransmissions inutiles, impactant elles aussi les performances globales du réseau.

Il serait alors intéressant d'analyser les schémas de retransmissions afin de déterminer si ces retransmissions sont liées à une compétition pour l'accès au canal ou à de mauvaises conditions de propagation.

Enfin nous avons mis en évidence des anomalies concernant les intervalles inter-trames sur les cartes TP-LINK TL-WN722N. Celles-ci sont dûes à une mauvaise implémentation de la taille des fenêtres de contention au sein du pilote de ces cartes, ainsi qu'à une inversion des files de priorités AC_BK et AC_-BE au sein de leur microcode. Cette dernière peut avoir un impact important car lorsque du trafic ne se voit pas attribuer une priorité particulière il utilise la priorité BE. L'inversion de files fait que ce trafic lambda est envoyé avec une priorité BK.

Nous travaillons actuellement sur l'implémentation d'un prototype d'algorithme d'adaptation de débit essayant d'une part de déterminer les cas de contention afin de ne pas réduire aussi drastiquement les

débits des cartes, et d'autre part de determiner les cas où le lien se dégrade afin d'adapter la modulation utilisée ou d'initier une procédure de *handover*, afin de limiter le nombre de retransmissions.

IP Geocast

Sommaire

5.1 Communications géocentriques pour DataTweet 114
5.2 IP Geocast 116
5.2.1 Géopréfixes 117
5.2.2 Encodage au format multicast IPv6 119
5.2.3 Régions aux frontières des quadtrees 120
5.2.4 Propagation des géopréfixes 121
5.3 Discussion 121

Introduction

L'objectif de l'IoT est d'offrir de nouvelles façons de communiquer et d'interagir avec des objets ou des dispositifs connectés. Les objets intelligents et les communications M2M (Machine à machine — *Machine to Machine*) sont des éléments constitutifs essentiels de l'IoT et conduisent à de nouveaux moyens de « connecter le monde physique aux gens », de permettre de nouvelles interactions entre les citoyens et leur environnement, ouvrant ainsi de nouvelles applications pour les villes intelligentes.

Par ailleurs, les *smartphones* et les communications sans fil sont devenus omniprésents dans le monde entier, les *smartphones* jouant un rôle de passerelle vers le « monde connecté » pour la plupart des gens : les *smartphones* sont les dispositifs de prédilection par lesquels nous interagissons avec Internet. Ils sont également utilisés pour des mesures relatives au *Crowdsensing* [161] et, en tant que tel, ils fournissent un moyen naturel d'interagir avec des objets intelligents.

Parallèlement à ces transformations, un changement de paradigme s'opère : nous nous éloignons des communications centrées sur les « machines », dans lesquelles l'accent était mis sur les sources et destinations des communications, pour nous orienter vers des communications centrées sur les données, en mettant l'accent sur les données elles-mêmes. Dans ce contexte de communication de données omniprésentes, il devient important de savoir d'où viennent les données et où elles seront pertinentes, plutôt que de savoir qui a envoyé un message à qui. Par exemple, une mesure de température n'est intéressante que si elle est combinée avec l'endroit où elle a été prise et que l'utilisateur ne se soucie pas de savoir quel appareil particulier a recueilli les données. De plus, pour la diffusion des données, le lieu où les données sont significatives est très important, car il donne la portée géographique de la validité des données. Par exemple, la diffusion de l'horaire du prochain autobus est principalement pertinente au voisinage de l'arrêt d'autobus ciblé.

Dans un monde aussi connecté, les utilisateurs mobiles avec leurs *smartphones* ou leurs montres connectées sont non seulement des citoyens générant beaucoup de données pouvant être significatives, mais sont surtout des consommateurs de données ayant des attentes de plus en plus exigeantes. Ils sont susceptibles de se déplacer dans de nombreuses localités différentes, produisant ainsi beaucoup de données précieuses, mais s'attendant aussi à obtenir des données spécifiques dignes d'intérêt à des endroits particuliers. Cependant, il n'existe aujourd'hui aucun moyen fiable de transmettre des données aux utilisateurs situés dans des régions géographiques spécifiques en utilisant l'infrastructure existante d'Internet. Une façon d'aborder cette question est de faire en sorte que les dispositifs interrogent les serveurs en envoyant leurs coordonnées GPS par exemple, ou en concevant un moyen de récupérer les adresses IP de tous les dispositifs dans une zone ciblée pour leur envoyer des données *unicast*, mais de telles approches ne seraient pas viables à grande échelle.

Notre objectif est d'améliorer l'acheminement du trafic géographique vers des *points d'accès* sans fil couvrant une destination géographique ciblée, afin qu'elle puisse ensuite être « arrosée » par les dispositifs présents dans la région. Dans ce chapitre nous nous penchons tout d'abord sur les communications géocentriques en donnant un aperçu de la vision de *DataTweet* dans la première partie. Nous décrirons *IP Geocast* dans la partie suivante. Il s'agit d'un mécanisme de diffusion géographique multicast qui permet d'adresser des appareils dans un environnement géographique et hiérarchique. Les appareils calculent pour cela un *géopréfixe* à l'aide de leurs coordonnées GPS de sorte qu'ils se trouvent dans un quadrilatère 2D récursif. Nous pouvons encapsuler les *géopréfixes* dans une adresse IPv6 multicast pour rendre *IP Geocast* compatible avec l'infrastructure Internet déployée. Nous proposons par ailleurs un moyen de propager les *géopréfixes* pour construire l'arbre de distribution *IP Geocast* de manière multicast à l'intérieur d'un AS (Système autonome — *Autonomous System*) et en utilisant BGP (Protocole pour les routeurs de bordures — *Border Gateway Protocol*) au-delà. Enfin nous présentons les limites liées aux choix que nous avons fait dans l'élaboration d'*IP Geocast* dans la dernière partie.

5.1 Communications géocentriques pour DataTweet

Le projet *DataTweet* explore l'idée d'un service de données omniprésent pour la création de services IoT et M2M. Notre hypothèse de base est que n'importe quel appareil peut transmettre et recevoir des messages courts de la manière la plus simple : envoyer un message radiodiffusé « dans l'air » afin qu'il soit ensuite saisi

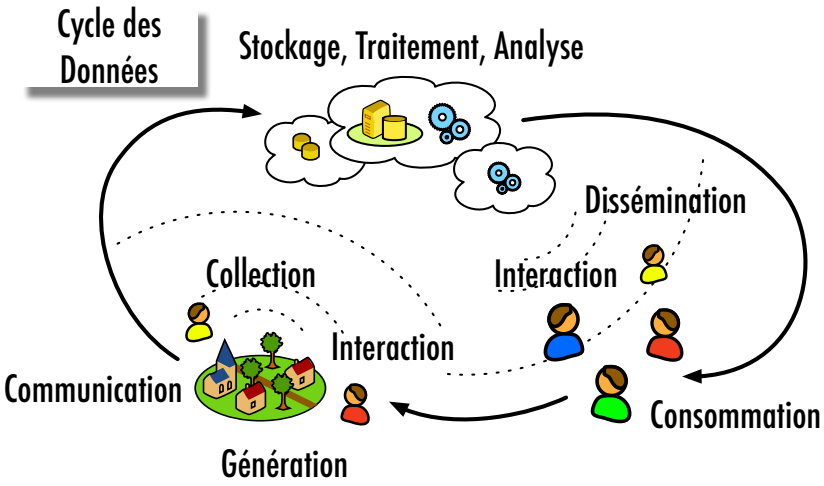


FIGURE 5.1 – Analogie avec le cycle de l'eau appliqué aux données

par les réseaux sans fil avoisinants et amené jusqu'à destination, ou inversement, diffuser « dans l'air » des messages ayant un intérêt à proximité.

Cette hypothèse s'appuie sur le fait que la plupart des zones habitées sont entièrement couvertes par une connectivité sans fil dense et nous supposons qu'une petite fraction de la bande passante des ressources sans fil pourrait être affectée au transport du trafic *DataTweet* composé de messages courts envoyés avec un faible débit.

Dans le contexte de l'IoT, l'un des attributs clés des données est la localisation, qui peut être utilisée pour qualifier l'origine ou la destination des données. Combinée à une approche centrée sur le contenu, de nombreux services peuvent être construits sur l'hypothèse exprimée dans *DataTweet* : les données envoyées dans l'air sont transmises et filtrées par les utilisateurs intéressés en fonction de leurs contenus, de leurs attributs ou de leurs fonctionnalités.

Le fonctionnement de *DataTweet* est illustré par l'analogie du cycle de l'eau présentée à la FIGURE 5.1. Des foules d'appareils et d'utilisateurs mobiles génèrent des données envoyées vers les serveurs *Cloud* ou *Fog* via les *points d'accès* les plus proches comme l'eau s'évaporant dans le ciel pour former des nuages. Les données collectées deviennent des informations utiles dans le *Cloud*, en raison du sens et de l'interprétation que nous pouvons donner aux données brutes, ainsi que des nouvelles connaissances qui peuvent être extraites par traitement et analyse. Ensuite, les données contenues dans le *Cloud* peuvent être utilisées pour « arroser » les utilisateurs et les appareils dans des zones ciblées par le biais de messages courts.

Les données recueillies dans toute une région, par exemple la position des véhicules, peuvent conduire à la création de services à valeur ajoutée, comme la gestion intelligente du trafic routier : l'analyse des données dans le *Cloud* peut fournir des informations de haut niveau en temps quasi réel qui peuvent être redistribuées aux utilisateurs participants.

La FIGURE 5.2 illustre l'architecture avec les utilisateurs et équipements IoT qui envoient les messages *DataTweet* aux nœuds *DataTweet* les plus proches servant de *points d'accès*— pouvant être colocalisés ou intégrés aux *points d'accès* 802.11 existants —, qui les transfèrent ensuite soit directement aux utilisateurs du secteur soit à d'autres destinations sur Internet comme les serveurs *Cloud* ou *Fog* où ils seront traités.

Dans ce chapitre, nous nous concentrons sur *IP Geocast*, qui est l'un des éléments clés de l'architecture *DataTweet* : il prend en charge la transmission du trafic de données vers des régions spécifiques et « envoie » les données sur tous les hôtes présents dans une région cible. Nous partons du principe que les nœuds *DataTweet* connaissent leurs coordonnées géographiques via GPS ou par configuration, desservent des zones locales et peuvent transmettre des messages *DataTweet* à des destinations par le biais d'*IP Geocast*.

L'infrastructure de transmission *DataTweet* décrite ci-dessus tire parti d'*IP Geocast* pour atteindre des destinations géographiques identifiées par des *géopréfixes*. La proposition d'agrégation des préfixes géographiques, similaire aux préfixes de réseau IP, permet aux routeurs de créer une arborescence de distribution évolutive pour la propagation des *geocasts* dans les régions géographiques concernées. Enfin, les nœuds de destination diffusent des messages *DataTweet* dans les zones locales et les utilisateurs les filtrent en fonction des métadonnées caractérisant leur contenu en fonction de leurs intérêts personnels.

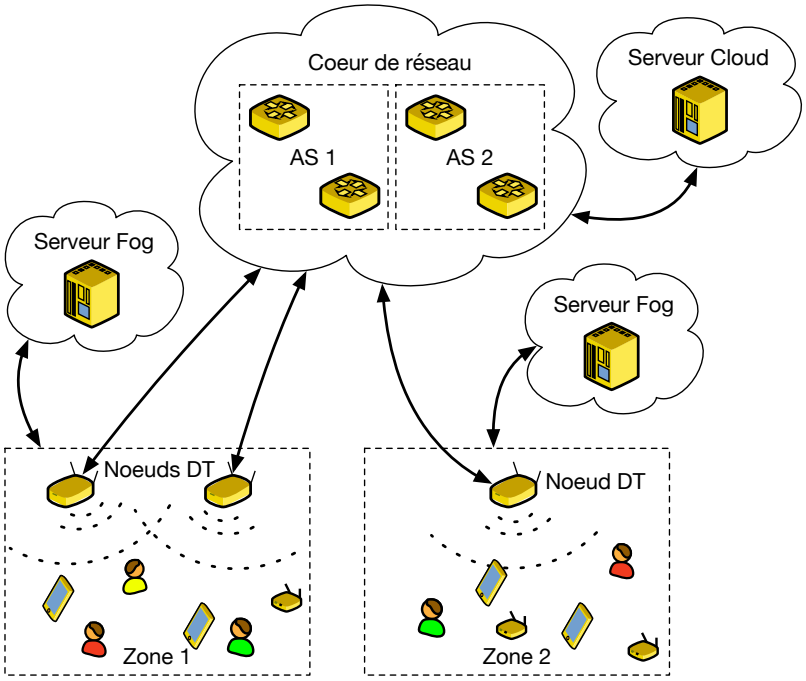


FIGURE 5.2 – Architecture de *DataTweet*

5.2 IP Geocast

Nous voulons que *DataTweet* puisse transmettre des messages aux localisations dans lesquels les données sont pertinentes pour les utilisateurs, car elles concernent le voisinage utilisateur. Par exemple, nous pourrions recevoir l'information concernant le prochain autobus arrivant à un arrêt voisin et l'heure à laquelle le prochain tramway arrivera à un autre arrêt pour que nous puissions décider d'aller à l'arrêt offrant le meilleur temps de transit. Dans ce contexte, l'emplacement cible des données est très important car il s'agit d'un domaine spécifique auquel l'infrastructure doit transmettre les messages *DataTweet*.

Nous voulons tirer parti des réseaux existants afin que notre conception s'adapte au mieux à la topologie actuelle d'Internet— couverture capillaire des régions géographiques par un FAI — et à sa structure — AS multiples de différents FAIs. Il est très difficile d'assurer le transfert géographique dans l'architecture d'Internet actuelle, car la topologie ne tient pas compte des considérations géographiques.

Nous supposons que les nœuds *DataTweet* connaissent leur emplacement géographique et qu'une source *DataTweet* souhaite envoyer un message à tous les nœuds *DataTweet* dans une région géographique donnée. Notre approche consiste à utiliser le maillage Internet existant pour transmettre les messages *DataTweet* à des régions géographiques choisies. Le problème est que les adresses IP sont logiques, donc découplées des emplacements géographiques des hôtes. Par conséquent, nous ne pouvons pas compter sur le schéma d'adressage IP actuel pour cibler une zone géographique particulière. Une approche consisterait à se renseigner sur l'emplacement géographique d'un hôte donné et à lui envoyer un paquet IP standard — des travaux récents se sont concentrés sur l'obtention de ces dits emplacements géographiques [162, 163]). Cependant, l'envoi de paquets à un groupe d'hôtes dans une région géographique donnée obtenus de cette façon ne passe pas à l'échelle.

Un problème similaire se pose dans les VANETs (Réseaux Ad-Hoc de véhicules — *Vehicular Ad-Hoc Networks*) [164] : une source veut envoyer un paquet à des routeurs d'accès près de l'emplacement des véhicules de destination sur Internet. Ce problème a été étudié selon deux approches principales : 1. DNS (Système de noms de domaine — *Domain Name System*) étendu : l'enregistrement de ressources DNS LOC maintient la correspondance entre l'emplacement du routeur d'accès et son adresse IP, 2. *GeoServer* dans une zone géographique donnée qui sert de réflecteur aux messages. La solution basée sur le DNS étendu ne prend en charge que le transfert de paquets unicast et la seconde solution nécessiterait le déploiement de *GeoServers* dans chaque région géographique cible, ce qui n'est pas pratique car les régions ne sont pas définies a priori.

Nous proposons de définir des régions géographiques de taille de plus en plus réduite, organisées hié-

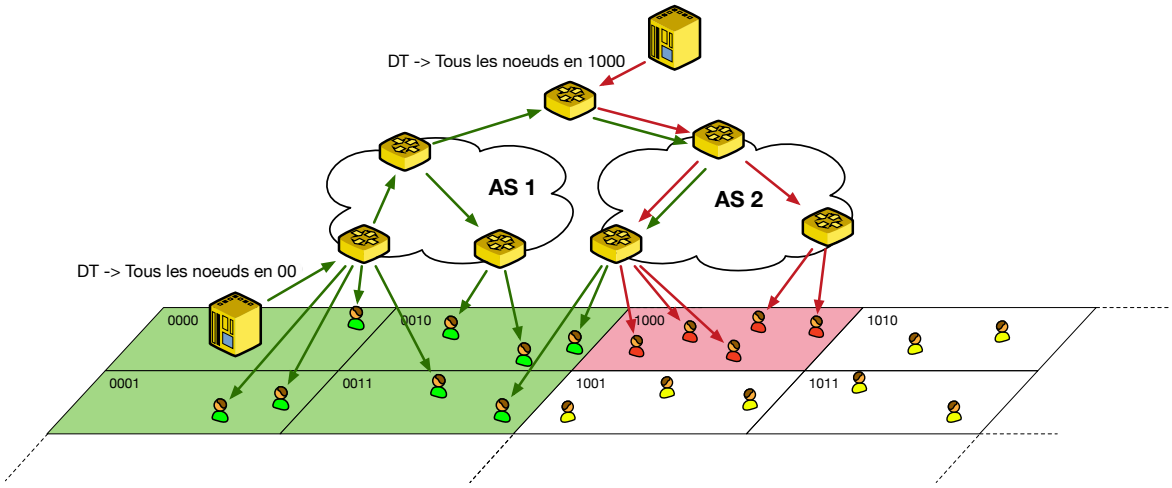


FIGURE 5.3 – Architecture d'*IP Geocast*

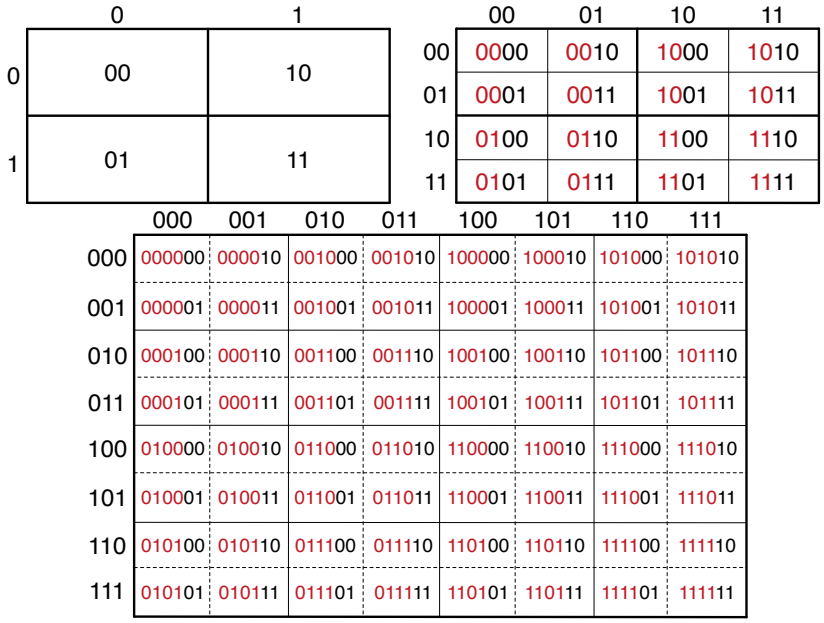


FIGURE 5.4 – Partitionnement de l'espace 2D en utilisant un *quadtree* avec des *géopréfixes* obtenus en utilisant le schéma de codage de Morton

rarchiquement en utilisant une division en *quadtree* de l'espace d'adressage GPS 2D et de tirer parti de l'acheminement standard de paquets IPV6 pour atteindre ces régions.

La FIGURE 5.3 illustre le principe de la géodiffusion IP : le monde est divisé en régions suivant un *quadtree* et nous associons un identificateur à une région — un préfixe géographique, *géopréfixe*, correspondant aux coordonnées GPS et à la taille de la région. De cette façon, un *géopréfixe* identifie de façon unique une région à un endroit donné avec une taille donnée et représente tous les hôtes dans la région. Les nœuds *DataTweet* propagent des *géopréfixes* dans le réseau pour créer une arborescence de distribution similaire à celle effectuée en *multicast*. Un hôte peut envoyer un message *DataTweet* à une région en spécifiant son préfixe de sorte que le message suive l'arbre de distribution pour atteindre les nœuds *DataTweet* qui le diffusent à tous les nœuds dans la région.

5.2.1 Géopréfixes

La FIGURE 5.4 présente le principe du partitionnement en *quadtrees* : l'espace de coordonnées GPS est divisé en deux parties égales qui sont ensuite divisées en deux moitiés. Le partitionnement récursif en *quadtrees* permet d'obtenir des régions ayant une taille arbitrairement petite.

TABLEAU 5.4 – Précision en fonction du nombre de décimales et de la taille du *géopréfixe*

# of figures	# of bits	Equator	45°N/S
0	9	111.3200 km	78.710 km
1	12	11.1320 km	7.871 km
2	16	1.1132 km	787.100 m
3	19	111.3200 m	78.710 m
4	22	11.1320 m	7.871 m
5	26	1.1132 m	787.100 mm

Les *géopréfixes* doivent avoir les propriétés suivantes :

- Une région 2D d'un *quadtree* identifiée par ses coordonnées GPS et sa taille doit correspondre à une seule valeur de *géopréfixe*.
- L'agrégation de 4 *géopréfixes* correspondant à des régions contiguës donne un préfixe correspondant à la région englobante. Le fait de raccourcir un *géopréfixe* de 2 bits donne le préfixe englobant.

Le système de coordonnées de référence WGS84 utilisé dans le GPS permet de diviser la sphère parfaite du monde en 360 degrés de longitude horizontale et 180 degrés de latitude verticale. Un format usuel pour une coordonnée est $ddd.dddddddddd$, où d représente un chiffre de l'angle exprimé en degré et les degrés sont exprimés sous la forme d'un nombre compris entre −180 et +180 pour la longitude, et un nombre compris entre −90 et +90 pour la latitude — les positions à l'ouest et au sud sont négatives —, par exemple $(+28.61, -80.61)$ correspond à la position du centre spatial de Cape Canaveral. Il est à noter que dans ce système de référence, la précision de longitude n'est pas la même selon la latitude d'un point. Plus un point est éloigné de l'équateur, moins la distance entre deux degrés longitudinaux est grande. Le TABLEAU 5.4 représente la résolution longitudinale à l'équateur et à une latitude de $45° N/S$ avec l'augmentation du nombre de décimales et du nombre de bits correspondants. L'idée est de relier la taille d'une région au nombre de bits utilisés pour représenter une coordonnée GPS donnée, reliant ainsi la taille d'une région à la taille d'un *géopréfixe*.

De plus, nous devons encoder les coordonnées GPS en une seule valeur géoréférencée. Il y a plusieurs schémas de codage possibles comme les codes Gray [165], les fonctions d'appariement Cantor [166], ou les codes de Morton [167]. Morton a proposé de calculer une valeur unidimensionnelle à partir des coordonnées GPS en entrelaçant les représentations binaires des coordonnées : la valeur Z est un point sur la courbe ordonnée en Z (*Z-order*) — une courbe spécifique d'une seule dimension capable de remplir un espace à n dimensions. Les courbes *Z-order* ont la propriété de conserver un certain degré de localité pour les emplacements codés. Une fois qu'un point est projeté sur la courbe *Z-order*, il y a de fortes chances qu'il partage le même préfixe binaire avec les points qui sont proches dans l'espace multidimensionnel. Nous avons choisi l'encodage de Morton pour cette raison ainsi que la simplicité de transformation des coordonnées en valeurs binaires.

Les préfixes résultants sont ordonnés de la même manière que la hiérarchie obtenue lors de la partition en *quadtree* illustrée dans la FIGURE 5.4, par exemple, la valeur 00111111 correspond aux coordonnées GPS de $(011, 011)$ en binaire. A chaque étape du partitionnement en *quadtree*, le nombre de bits du préfixe augmente, ce qui permet d'avoir une division spatiale de plus en plus précise utilisant une courbe *Z-order* plus complexe. On peut remarquer que raccourcir un géopréfixe de 2 bits signifie représenter le carré englobant : un emplacement dans un carré plus large à l'étape précédente est dans un carré plus petit dont le préfixe binaire correspond au préfixe carré plus large, par exemple, un point avec les coordonnées dans le carré 001111/6 (nous utilisons la notation CIDR (Routage inter domaines sans classes — *Classless Inter-Domain Routing*) : la longueur du préfixe est 6 bits) est aussi dans le carré plus grand 0011/4 ainsi que dans le carré entourant 00/2 (sur le schéma, les bits en rouge correspondent au préfixe de la case englobante au stade précédent).

Pour coder les coordonnées GPS comme décrit ci-dessus, nous proposons de représenter les coordonnées GPS comme des entiers positifs : nous décalons la longitude et la latitude de 180 et 90, respectivement, pour les transformer en entiers positifs avec le nombre de chiffres décimaux que nous voulons considérer. Avec 26 bits pour coder à la fois la latitude et la longitude, le codage Morton donne un préfixe de 52 bits permettant de diviser la surface du monde en régions géographiques ayant une résolution de l'ordre du mètre. Comme les réseaux 802.11 fournissent une portée de quelques dizaines de mètres, cette résolution est suffisamment précise pour exprimer les régions que nous voulons traiter tout en maintenant un schéma

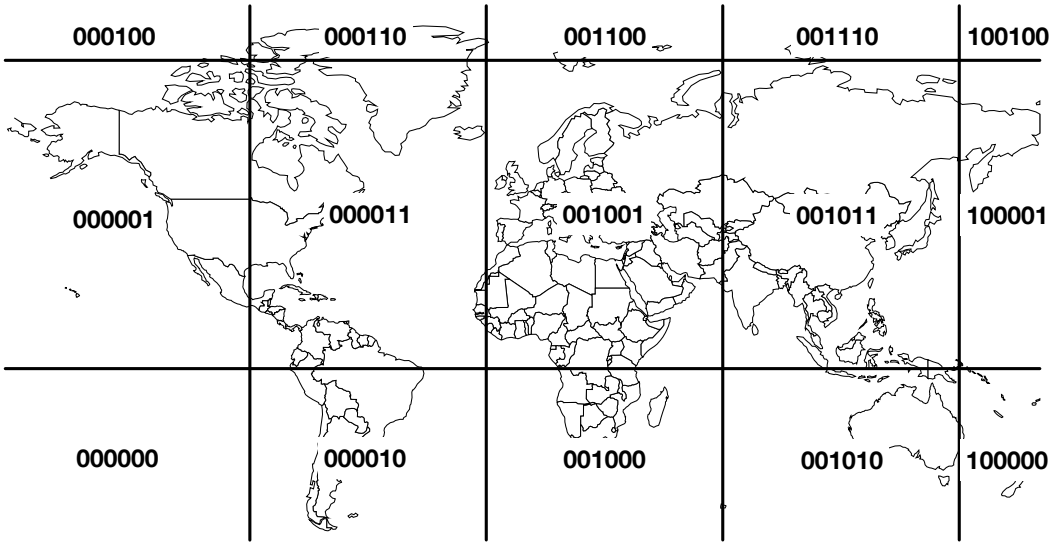
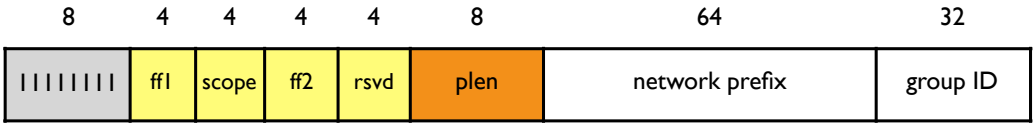
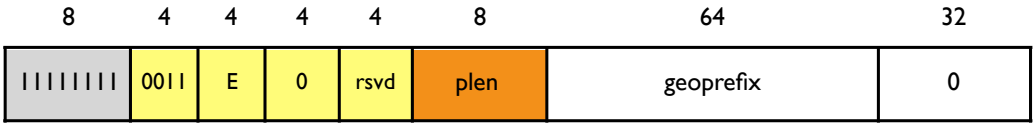


FIGURE 5.5 – Partitionnement du monde en *quadtree* avec des exemples de *géopréfixes* pour certaines régions



(a) *multicast*



(b) *Geocast*

FIGURE 5.6 – Structures des adresses IPV6 *multicast* et *geocast*

d'adressage relativement court. Notez que cette façon d'exprimer la taille des régions est similaire aux préfixes IP du CIDR : on peut spécifier de grandes régions en utilisant un petit nombre de bits comme un petit nombre de bits dans un préfixe CIDR correspond à un grand bloc d'adresses IP.

La FIGURE 5.5 présente un exemple d'application du codage de Morton à une projection Mercator de la surface terrestre. Par exemple, les coordonnées GPS $(+28.61, -80.61)$ correspondent au préfixe 32 bits 0000 1100 0111 0011 1010 1101 1101 1101 1101.

5.2.2 Encodage au format multicast IPv6

Pour profiter de l'architecture actuelle de transmission IPV6, nous proposons d'utiliser une partie de l'espace d'adressage IPV6 multicast [168] pour interpréter les *géopréfixes* en adresses. La FIGURE 5.6 présente la structure des adresses IPV6 *multicast* et *geocast* telles que nous les définissons — les bits les moins significatifs sont à droite. Le champ `scope` est mis à `E` qui correspond à un `scope` global, le *géopréfixe* est placé dans les 64 bits les moins significatifs et sa longueur — équivalent à la résolution voulue — est renseignée dans le champ `plen`.

FIGURE 5.7 – *Géoprefixe* de taille *n*

TABLEAU 5.5 – Préfixes dans la région de Boston

Region	32 bit geoprefix
Massachusetts Institute of Technology Campus	0000 1101 1000 0110 1001 0101 1101 1011
Cambridge	0000 1101 1000 0110 1001 0101 1101 0000
Boston Area	0000 1101 1000 0110 1001 0101 1100 0000

Mettre les $64-n$ bits les moins significatifs à 0 et `plen` à n nous permet de représenter un géopréfixe avec n bits significatifs permettant de définir une région de la taille correspondant aux valeurs présentées dans le TABLEAU 5.4. La FIGURE 5.7 illustre l'encodage des coordonnées GPS $(+28.61, -80.61)$ sous forme de *géopréfixe* avec $n = 32$ bits. Cela correspond à une superficie de 1,1 kilomètre carré à cette latitude.

Une telle représentation est similaire aux masques de sous-réseaux dans lesquels le nombre de bits les moins significatifs mis à 0 indique la taille du préfixe. Avec la propriété d'agrégation — une région avec un *géopréfixe* de n bits représente l'agrégation de quatre régions ayant des *géopréfixes* de $n+2$ bits — ceci permet l'utilisation de *géopréfixes* dans les tables de routage pour la transmission des paquets en s'appuyant sur l'infrastructure d'Internet existante.

Le TABLEAU 5.5 présente un autre exemple dans lequel nous spécifions des préfixes pour des régions de taille croissante : un préfixe 32 bits correspond au campus du *Massachusetts Institute of Technology* — de $(+42.352, -71.0976)$ à $(+42.36224, -71.08736)$. Si nous mettons 4 bits à 0, nous pouvons identifier une région plus grande incluant les campus de *Cambridge* du *Massachusetts Institute of Technology* et de *Harvard* — de $(+42.34174, -71.12832)$ à $(+42.38272, -71.08735)$. En mettant 2 bits supplémentaires sur 0, on obtient une région englobant *Boston* et sa banlieue sud — de $(+42.3008, -71.12832)$ à $(+42.38272, -71.04638)$.

O'Daniel et al [169] ont envisagé d'encoder les coordonnées GPS dans une adresse IPV6 *unicast* d'une manière différente : ils utilisent le champ `Subnet-ID` pour encoder la latitude et la longitude avec une granularité d'un degré et ajoutent deux décalages de 24 dans le champ d'identification de l'interface. Notre encodage au format *multicast* a l'avantage de représenter les coordonnées sous forme de préfixes permettant de faire un *Longest Prefix Match*.

Pesavento et al [166] ont envisagé d'encoder les coordonnées GPS dans un URI (Identifiant de ressource uniforme — *Uniform Resource Identifier*) dans le contexte des NDNs (Communication de données nommées — *Named Data Networking*s). Ils ont utilisé la fonction de couplage de Cantor pour faire correspondre les coordonnées GPS à une valeur scalaire et concaténer ses chiffres décimaux avec l'URI correspondant au nom d'un lieu géographique.

5.2.3 Régions aux frontières des quadtrees

Comme nous l'avons mentionné précédemment, les courbes *Z-order*, et plus généralement la division en *quadtree*, conservent un certain degré de localité, ce qui signifie que les points géographiquement proches les uns des autres sont susceptibles de partager le même préfixe ou au moins partagent une racine de préfixe commune. Considérons la FIGURE 5.4 : les points situés à $(110,010)$ et $(110,011)$ partagent le préfixe $1011/4$ et les points dans $(110,010)$ et $(110,001)$ partagent le préfixe $10/2$. Dans les deux cas, si nous voulons envoyer un message à une zone qui chevauche deux de ces carrés, nous pouvons exprimer la destination comme un préfixe partagé, ce qui signifie que nous n'avons besoin de transmettre qu'un seul paquet aux zones suffisamment larges pour couvrir toutes les zones que nous ciblons. Ensuite, le paquet doit être dupliqué pour atteindre toutes les sous-zones.

Cependant, le partitionnement en *quadtree* crée des régions qui ne partagent pas un préfixe commun qui permettrait de transférer un seul paquet vers les régions qui se chevauchent. Par exemple, si nous voulons adresser un message à une zone composée de 4 carrés au centre du quadrilatère de la FIGURE 5.4 : 001111, 100101, 011010 et 110000, il n'y a pas de préfixe pour les représenter, car ils se trouvent dans différents *quadtrees*. Donc, si nous voulons envoyer un paquet dans cette région, les préfixes des 4 carrés doivent être dans les tables de routage et 4 paquets doivent être transférés vers les régions. Dans le pire des cas, nous enverrons quatre paquets sur toute l'arborescence de distribution *IP Geocast* au lieu d'un seul paquet, si les préfixes des régions ne peuvent pas être agrégés. Nous pensons que cela en vaut toujours la peine, car chaque paquet dessert une zone entière et génère donc beaucoup moins de trafic que l'utilisation de plusieurs paquets *unicast* pour atteindre tous les nœuds ciblés.

Nous réfléchissons à l'élaboration d'un système de codage d'un plus grand nombre d'informations qui

aidera à réduire le nombre de préfixes qui doivent être propagés pour soutenir les régions non agrégées qui se chevauchent. Une idée pour contourner ce problème est de considérer plusieurs divisions en quadrilatères, chacune étant légèrement décalée d'une autre, et de définir un *flag* dans l'adresse IPv6 *multicast* pour spécifier qu'un *géopréfixe* donné est décalé.

5.2.4 Propagation des géopréfixes

Enfin, nous devons proposer un moyen de propager les *géopréfixes* dans le réseau pour construire une infrastructure de transit. L'idée est de gérer la propagation d'un *géopréfixe* de manière similaire à PIM SM (Protocole *multicast* indépendant du protocole de routage utilisant un point de rendez-vous — *Protocol-Independent Multicast, Sparse Mode*) [170] à l'intérieur d'un AS et d'utiliser l'approche similaire au *Free Riding Multicast* en dehors d'un AS [171].

Free Riding Multicast se concentre sur le routage inter-domaines : un AS annonce les groupes *multicast* actuellement utilisés dans le domaine. Les annonces sont ensuite propagées par BGP afin que les routeurs frontaliers aient une description des groupes présents pour chaque préfixe de destination. Pour découvrir l'arbre de diffusion d'un groupe G, le routeur de bordure d'une source *multicast* analyse sa table BGP pour identifier les préfixes avec les membres de G et calcule l'arbre de diffusion à partir de l'union des chemins *unicast* BGP vers tous les domaines de destination. Il achemine ensuite un seul paquet *multicast* à chaque saut suivant sur cet arbre de dissémination avec un codage du sous-arbre, chaque saut suivant doit à son tour acheminer le paquet.

Dans le cas de *Free Riding Multicast*, les préfixes de groupe sont difficilement agrégeables — il considère les adresses de groupe comme des identificateurs plats propagés dans l'interconnexion des AS grâce à BGP. Dans notre cas, nous pouvons agréger les préfixes de *geocast* d'une manière hiérarchique qui reflète la topologie capillaire des FAIs existants.

A l'intérieur d'un AS, nous proposons d'organiser la distribution *geocast* de la même manière que PIM SM : nous mettons en place un point de rendez-vous au niveau d'un routeur central qui reliera les autres routeurs régionaux. Les nœuds *DataTweet* envoient des messages *Join* aux points de rendez-vous pour annoncer les *géopréfixes*. Les routeurs traitent les messages *Join* saut-par-saut et mettent à jour leur tables de routage en agrégeant éventuellement les *géopréfixes*. Une source IP *geocast* envoie son premier paquet encapsulé dans un message *Register* à son point de rendez-vous qui extrait le paquet *geocast* et l'envoie sur l'arbre de distribution. Lorsqu'un point de rendez-vous reçoit le paquet encapsulé dans un message *Register*, il envoie un message *Join* vers la source, créant ainsi l'arbre de distribution partagé. La source peut passer de l'utilisation de l'arbre partagé à l'utilisation de l'arbre du plus court chemin en envoyant un message *Join* vers la source, qui change l'état des routeurs le long du chemin vers la source. Lorsque les paquets commencent à arriver en utilisant l'arbre du plus court chemin, un noeud *DataTweet* envoie un message *Prune* vers la source en utilisant l'arbre partagé afin d'éviter le trafic dupliqué. La principale différence avec l'utilisation de PIM SM de base est l'agrégation des géopréfixes, ce qui permet de partager les tables de transfert des routeurs par rapport à une entrée par groupe *multicast*.

5.3 Discussion

Selon le type de données qu'un message contient, il peut être intéressant d'envoyer le message à une zone plus large que la zone d'intérêt initiale. Supposons que certaines données présentent un intérêt dans une zone de quelques centaines de mètres carrés, des utilisateurs mobiles qui traversent une telle zone assez rapidement ne recevront peut être pas l'information. Une approche pour s'assurer que les utilisateurs intéressés recevront le message avec une probabilité élevée serait d'envoyer périodiquement le message dans l'air avec une fréquence élevée. Une autre approche consisterait à échanger la fréquence contre une portée géographique plus grande. Lorsque nous considérons des utilisateurs mobiles, nous pouvons envoyer le message dans une zone légèrement plus large et profiter du fait que les utilisateurs pourront avoir déjà reçu les données avant d'entrer dans la région où elles auront un intérêt.

De plus, il peut ne pas être nécessaire d'envoyer des messages à tous les nœuds d'une zone donnée. Si les nœuds peuvent relayer les données sur leur liaison sans fil, nous pouvons tirer parti du relais opportuniste pour aider à propager les messages *geocast* du réseau.

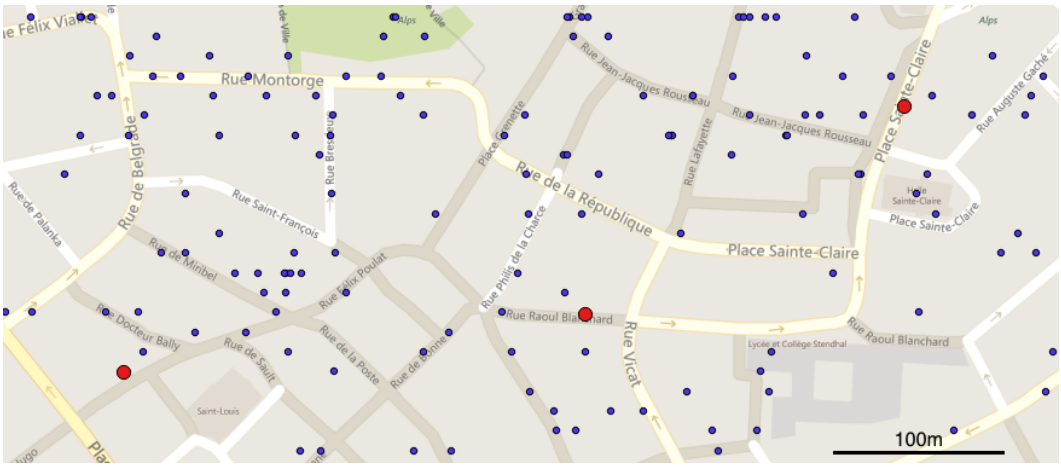


FIGURE 5.8 – « Freeboxes » (points bleus) et arrêts de tramway (points rouges) dans le centre ville de Grenoble

En outre, il est probable qu'une zone sera « couverte » par plusieurs AS. Dans ce cas, il est concevable de n'envoyer les données qu'à un sous-ensemble d'AS et de pouvoir tout de même couvrir cette zone géographique. Considérons par exemple la FIGURE 5.8 qui représente les emplacements géographiques des « Freeboxes » dans le centre-ville de Grenoble — d'autres FAI couvrent également cette zone — ainsi que trois stations de tramway présentes dans le secteur. Cet exemple met en évidence le fait que même à une échelle limitée, la densité des points d'accès semble suffisamment élevée pour couvrir une zone à proximité des points d'intérêt. De même, si la source de données est géographiquement proche de la zone de destination et est située dans un AS qui dessert cette zone, nous pouvons éviter d'envoyer le message à travers tout l'arbre de distribution et simplement le redistribuer localement.

Conclusion

Ce chapitre décrit *IP Geocast*, un protocole géographique *multicast* adapté à l'architecture d'Internet. Il permet aux appareils de dériver une adresse IPV6 *multicast* en fonction de leurs coordonnées GPS. Ces adresses ont la propriété de permettre aux équipements géographiquement proches les uns des autres d'avoir une forte probabilité de partager un même *géopréfixe*. Cette propriété permet de représenter les zones géographiques de manière hiérarchique, ce qui convient pour : 1. acheminer les messages vers leurs destinations géographiques en utilisant une politique de routage de type *Longest Prefix Match* et 2. adresser tous les dispositifs à l'intérieur d'une zone étendue en utilisant le *géopréfixe* incluant toutes les zones sous-jacentes.

Cependant, les *géopréfixes* suivent une division récursive de l'espace selon un *quadtree*, qui les limite aux carrés résultants de cette division. Envoyer des messages à une zone chevauchant de tels carrés ne peut pas se faire avec un *géopréfixe* commun et par conséquent, plusieurs messages doivent être envoyés à chaque partie de la zone ciblée. Dans le cadre de nos travaux futurs, nous prévoyons de proposer une solution pour résoudre ce problème. Nous aimerions aussi proposer une implémentation de ce schéma d'adressage géographique afin de pouvoir en évaluer les performances.

Conclusion générale

Les travaux présentés s'inscrivent dans le cadre du projet *DataTweet*, au sein duquel il était question de définir un protocole de communication pour l'IoT. Il avait notamment pour but d'utiliser l'ensemble des moyens de communications disponible dans le voisinage d'un équipement afin que celui-ci puisse envoyer ses messages. Ces messages devaient ensuite être acheminés non plus par le biais d'adresses de destinations définies, mais par la validité spatio-temporelle du contenu des données, permettant ainsi à la fois à l'émetteur de ne pas avoir à se soucier de connaitre l'adresse de destination d'un message, et de permettre de relayer une donnée à un ensemble de machines ayant utilité de ce type de données.

Lors de la généralisation de ce concept, il est possible d'envisager que les données envoyées par différents équipements connectés soient alors relayées vers différents centres de traitement capables de stocker et analyser ces données. Ces centres sont alors en mesure de générer de nouvelles données « enrichies » pouvant à leur tour être acheminées vers des équipements et utilisateurs ayant un besoin de données. Typiquement, les données de déplacement de véhicules peuvent être transmises vers des centres de contrôle routiers, qui pourraient alors redistribuer des données concernant les zones congestionnées aux utilisateurs en approche de celles-ci.

Dans ce contexte, nous nous sommes intéressés à l'utilisation des *points d'accès* 802.11 privés au sein des villes car ceux-ci sont omniprésents. Or, 802.11 n'étant pas prévu pour gérer la mobilité, nous avons estimé le taux de connectivité qu'un utilisateur pouvait espérer obtenir en fonction de sa vitesse, de la durée de *handover* et de la densité de *points d'accès* présents. Ainsi, comme montré dans le chapitre 3 traitant de l'exploitation des *points d'accès* 802.11 de la ville, les piétons peuvent espérer être connectés pendant plus de 80% de la durée de leurs trajets avec des durées de *handover* valant $5s$. A l'opposé, les utilisateurs ayant des vitesses de déplacement élevées ne sont plus du tout en mesure d'avoir une connectivité avec de telles durées de *handovers*. De ce constat nous pouvons dire plusieurs choses. Tout d'abord, pour des utilisateurs se déplaçant à des vitesses faibles, il est tout à fait envisageable de pouvoir utiliser une telle infrastructure pour des applications ayant une forte utilisation en terme de débit car ces utilisateurs restent à la fois longtemps sur un même *point d'accès* et restent déconnectés uniquement pendant leurs périodes de *handovers* dans la majorité des cas. Ensuite les utilisateurs ayant des vitesses modérément élevées ne restent plus assez longtemps sur un *point d'accès* afin de pouvoir efficacement les utiliser pour d'importants transferts de données. Cependant ils peuvent espérer se connecter assez fréquemment pendant de courtes durées le long de leurs trajets, permettant alors d'envisager d'utiliser cette infrastructure afin d'effectuer de la collecte de données. Enfin, si l'on souhaite à la fois pouvoir utiliser ces *points d'accès* à des vitesses plus élevées, ou pour faire passer du trafic ayant des contraintes fortes concernant les latences, tel que la VoIP, alors il faut grandement travailler à la réduction de la durée de *handover*.

Concernant celle-ci, nous nous sommes penchés dans une seconde contribution aux comportements de différents équipements 802.11 lorsqu'ils perdent leur connectivité, notamment leurs comportements concernant leurs retransmissions. Nous constatons que, d'une part les cartes 802.11 étudiées ne respectent pas pour la plupart les contraintes du standard, et d'autres part qu'elles ne détectent pas rapidement la perte de connectivité malgré un nombre très important de retransmissions, lesquelles sont faites en utilisant des modulations de plus en plus robustes. Nous avons d'ailleurs observé que lors de la contention, les algorithmes de contrôle de débit changent aussi les modulations car ils observent des pertes liées au nombre croissant de collisions. Or dans le cas présent, un tel changement de modulation pénalise à la fois la *station* ayant baissé sa modulation inutilement mais aussi le débit global du réseau car le canal sera occupé plus longtemps. Lorsque l'ensemble des *stations* adoptent ce comportement, les débits globaux deviennent catastrophiques comparés à l'utilisation d'une unique modulation. Enfin nous avons mis en évidence des erreurs dans les implémentations de 802.11 sur les cartes que nous avons testées concernant les fenêtres de contention et les files de priorité de trafic.

Concernant la dissémination de données, nous nous sommes penchés sur l'adressage de données ayant une forte validité spatiale. Nous avons proposé un schéma d'adressage géographique hiérarchique pouvant être utilisé dans l'infrastructure existante d'Internet. Il repose sur un découpage du monde en *quadtrees* à l'aide des codes de Morton. Ces même codes ayant la caractéristique de faire apparaître des préfixes partagés entre des zones géographiquement proches — les sous-zones d'une même zone englobante partagent un préfixe commun —, nous pouvons les utiliser de la même manière que les adresses IP et les masques de

sous-réseaux afin d'adresser des zones géographiques de tailles différentes. Nous proposons alors d'insérer ces *géopréfixes* dans des adresses *multicast* afin de pouvoir les utiliser sur l'infrastructure existante.

Enfin, nous nous sommes intéressés de manière transversale aux problèmes de reproductibilité et de répétabilité des expériences dans les réseaux sans fil. Pour cela nous avons développé une plateforme reproductible permettant la répétabilité et la reproductibilité des expérimentations : *WalT*. Cette plateforme utilise du matériel grand public peu onéreux et permet de redéployer à l'identique l'environnement logiciel d'une expérience grâce à *Docker*. Dans ce contexte seul l'environnement radio et la topologie déployée varient d'une plateforme *WalT* à une autre, mais nous considérons que ces variations permettent de conclure sur la validité d'un résultat.

Travaux futurs

Plusieurs pistes d'exploration sont envisagées pour les différents travaux menés.

Tout d'abord concernant l'utilisation des *points d'accès* dans les villes, nous avons vu qu'il était envisageable d'utiliser cette infrastructure pour des applications ayant des besoins importants en bande passante tel que le *streaming*, tant que les utilisateurs ne se déplacent pas trop vite. Pour ce type d'application, il serait intéressant d'utiliser les résultats que nous obtenons concernant les durées de connectivité, afin d'optimiser les politiques de *caching* en fonction des différents types d'utilisateurs.

Concernant le comportement des cartes 802.11, nous envisageons d'étudier plus en détail les retransmissions lors de la contention afin d'essayer de voir s'il est possible pour une carte de savoir si elle se trouve en contention ou en présence d'un lien de mauvaise qualité lorsqu'elle observe des pertes. En effet, actuellement, baisser la modulation lors de l'augmentation du nombre de pertes de paquet n'est pas optimal, et des solutions comme Minstrel mettent du temps à converger lorsque le signal est rompu. De plus nous pensons qu'il existe deux cas bien précis : soit nous sommes en présence de contention et dans ce cas là il ne faut absolument pas réduire le débit des cartes, mais augmenter les fenêtres de contention, soit nous sommes en présence d'un canal dégradé et il faut adapter le débit afin d'utiliser une modulation plus robuste. Nous envisageons d'effectuer des mesures supplémentaires afin d'observer le comportement d'une *station* lorsque celle-ci s'éloigne d'un *point d'accès*. Dans les travaux futurs nous analyserons les pertes, retransmissions et intervalles inter-trames, afin de définir si une *station* est en situation de contention ou utilise un canal bruité dans le but d'adapter de manière optimale son comportement afin de limiter ses pertes et son impact sur les autres *stations* de son voisinage.

Concernant *IP Geocast*, nous aimerions proposer une implémentation de celui-ci afin de pouvoir évaluer ses performances lors d'un déploiement dans une infrastructure de réseau. Nous aimerions aussi résoudre de manière correcte les problèmes liés aux bordures que crée un découpage en *quadtree*. Une approche naïve consisterait à définir plusieurs *quadtree* légèrement décalés, mais des frontières continueraient d'exister. Par ailleurs il serait intéressant de proposer un découpage non homogène. Actuellement *IP Geocast* découpe l'intégralité du monde en zones de plus ou moins un mètre carré en fonction de la latitude. Or, certaines zones n'ont peut-être pas besoin d'une telle précision, comme les océans par exemple. Ceci permettrait alors d'alléger la taille de l'encodage.

Bibliographie

[1] Cisco Visual Networking Index : Forecast and Trends, 2017–2022. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html, November 2018. (White Paper). 23, 80

[2] Elodie MORIN, Mickael MAMAN, Roberto GUIZZETTI et Andrzej DUDA : Comparison of the device lifetime in wireless networks for the internet of things. *IEEE Access*, 5:7097–7114, 2017. 23

[3] Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 1 : Radio Resource Measurement of Wireless LANs, 2008. 24, 36

[4] Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 8 : IEEE 802.11 Wireless Network Management, 2011. 24, 35

[5] Jerome HENRY : Finding Your Cell Edge. https://vimeo.com/118976638. 24

[6] Cisco Visual Networking Index : Global Mobile Data Traffic Forecast Update, 2017–2022. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.html, February 2019. (White Paper). 32

[7] M Isabel SANCHEZ, Antonio de la OLIVA et Carlos J BERNARDOS : Experimental analysis of connectivity management in mobile operating systems. *Computer Networks*, 94:41–61, 2016. 32

[8] Arunesh MISHRA, Minho SHIN et William ARBAUGH : An empirical analysis of the IEEE 802.11 MAC layer handoff process. *ACM SIGCOMM Computer Communication Review*, 33(2):93–102, 2003. 32, 33

[9] Vivek MHATRE et Konstantina PAPAGIANNAKI : Using smart triggers for improved user performance in 802.11 wireless networks. *In Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 246–259. ACM, 2006. 32

[10] Héctor VELAYOS et Gunnar KARLSSON : Techniques to reduce the IEEE 802.11b handoff time. *In Communications, 2004 IEEE International Conference on*, volume 7, pages 3844–3848. IEEE, 2004. 32, 33, 35, 39, 60

[11] Ramya RAGHAVENDRA, Elizabeth M BELDING, Konstantina PAPAGIANNAKI et Kevin C ALMEROTH : Understanding handoffs in large ieee 802.11 wireless networks. *In Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 333–338. ACM, 2007. 32

[12] Giuseppe BIANCHI : Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on selected areas in communications*, 18(3):535–547, 2000. 33, 35

[13] Haitao WU, Kun TAN, Yongguang ZHANG et Qian ZHANG : Proactive scan : Fast handoff with smart triggers for 802.11 wireless LAN. *In INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 749–757. IEEE, 2007. 33, 36

[14] Chien-Chao TSENG, Kuang-Hui CHI, Ming-Deng HSIEH et Hung-Hsing CHANG : Location-based fast handoff for 802.11 networks. *IEEE Communications letters*, 9(4):304–306, 2005. 33, 37

[15] Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 2016. 33

[16] Sudarshan VASUDEVAN, Konstantina PAPAGIANNAKI, Christophe DIOT, Jim KUROSE et Don TOWSLEY : Facilitating access point selection in IEEE 802.11 wireless networks. *In Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, pages 26–26. Usenix Association, 2005. 35, 40

[17] Nicolas MONTAVONT, Alberto BLANC, Renzo NAVAS, Tanguy KERDONCUFF et German CASTIGNANI : Handover triggering in IEEE 802.11 Networks. *In 2015 IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–9. IEEE, 2015. 35

[18] Lukasz WISNIEWSKI, Henning TRSEK, Ivan DOMINGUEZ-JAIMES, Anetta NAGY, Reinhard EXEL et Nikolaus KERÖ : Location-based handover in cellular IEEE 802.11 networks for Factory Automation. *In Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on*, pages 1–8. IEEE, 2010. 36

[19] Architectural Design and Specification. http://flexware.at/. 36

[20] Julien MONTAVONT et Thomas NOEL : IEEE 802.11 handovers assisted by GPS information. *In Wireless and Mobile Computing, Networking and Communications, 2006.(WiMob'2006). IEEE International Conference on*, pages 166–172. IEEE, 2006. 36

[21] Minho SHIN, Arunesh MISHRA et William A ARBAUGH : Improving the latency of 802.11 hand-offs using neighbor graphs. *In Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 70–83. ACM, 2004. 37

[22] Sang-Hee PARK, Hye-Soo KIM, Chun-Su PARK, Jae-Won KIM et Sung-Jea KO : Selective channel scanning for fast handoff in wireless LAN using neighbor graph. *In IFIP International Conference on Personal Wireless Communications*, pages 194–203. Springer, 2004. 37

[23] Yuh-Shyan CHEN, Ming-Chin CHUANG et Chung-Kai CHEN : DeuceScan : deuce-based fast handoff scheme in IEEE 802.11 wireless networks. *IEEE Transactions on Vehicular Technology*, 57(2):1126–1141, 2008. 37

[24] Ishwar RAMANI et Stefan SAVAGE : SyncScan : Practical fast handoff for 802.11 infrastructure networks. *Proceedings - IEEE INFOCOM*, 1:675–684, 2005. ISSN 0743166X. 38

[25] Sunggeun JIN, Munhwan CHOI, Lei WANG et Sunghyun CHOI : Fast scanning schemes for IEEE 802.11 WLANs in virtual AP environments. *Computer Networks*, 55(10):2520–2533, 2011. 38

[26] Mahnsuk YOON, Keuchul CHO, Jilong LI, Jeongbae YUN, Minyoung YOO, Youngil KIM, Qin SHU, Jang-Kyu YUN et Kijun HAN : AdaptiveScan : The fast layer-2 handoff for WLAN. *In Information Technology : New Generations (ITNG), 2011 Eighth International Conference on*, pages 106–111. IEEE, 2011. 38

[27] Sangho SHIN, Andrea G FORTE, Anshuman Singh RAWAT et Henning SCHULZRINNE : Reducing MAC layer handoff latency in IEEE 802.11 wireless LANs. *In Proceedings of the second international workshop on Mobility management & wireless access protocols*, pages 19–26. ACM, 2004. 39

[28] Vladimir BRIK, Arunesh MISHRA et Suman BANERJEE : Eliminating handoff latencies in 802.11 WLANs using Multiple Radios : Applications , Experience, and Evaluation. *IMC '05 Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, pages 10–19, 2005. 39

[29] Kishore RAMACHANDRAN, Sampath RANGARAJAN et John C LIN : Make-before-break mac layer handoff in 802.11 wireless networks. *In Communications, 2006. ICC'06. IEEE International Conference on*, volume 10, pages 4818–4823. IEEE, 2006. 39

[30] Sunggeun JIN, Munhwan CHOI et Sunghyun CHOI : Multiple WNIC-based handoff in IEEE 802.11 WLANs. *IEEE Communications Letters*, 13(10), 2009. 39

[31] Sonia WAHARTE, Kevin RITZENTHALER et Raouf BOUTABA : Selective active scanning for fast handoff in WLAN using sensor networks. *Mobile and Wireless Communication Networks*, pages 59–70, 2005. 40

[32] Ranveer CHANDRA et Paramvir BAHL : MultiNet : Connecting to multiple IEEE 802.11 networks using a single wireless card. *In INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 882–893. IEEE, 2004. 40

[33] Manpreet Singh DANG, Amol PRAKASH, Dinesh K ANVEKAR, D KAPOOR et Rajeev SHOREY : Fuzzy logic based handoff in wireless networks. *In Vehicular Technology Conference Proceedings, 2000. VTC 2000-Spring Tokyo. 2000 IEEE 51st*, volume 3, pages 2375–2379. IEEE, 2000. 40

[34] Wenhui ZHANG : Handover decision using fuzzy MADM in heterogeneous networks. *In Wireless communications and networking conference, 2004. WCNC. 2004 IEEE*, volume 2, pages 653–658. IEEE, 2004. 40

[35] Srinivas SHAKKOTTAI, Eitan ALTMAN et Anurag KUMAR : Multihoming of users to access points in WLANs : A population game perspective. *IEEE Journal on Selected Areas in Communications*, 25(6), 2007. 41

[36] Ben-Alexander CASSELL, Timur ALPEROVICH, Michael P WELLMAN et Brian NOBLE : Access point selection under emerging wireless technologies. *In Proc. of 6th workshop on the Economics of Networks, Systems, and Computation (NetEcon), San Jose, CA, USA*, 2011. 41

[37] Ouldooz Baghban KARIMI, Jiangchuan LIU et Jennifer REXFORD : Optimal collaborative access point association in wireless networks. *In INFOCOM, 2014 Proceedings IEEE*, pages 1141–1149. IEEE, 2014. 41

[38] Navrati SAXENA et Abhishek ROY : Novel framework for proactive handover with seamless multimedia over WLANs. *IET communications*, 5(9):1204–1212, 2011. 41

[39] Sangheon PACK et Yanghee CHOI : Fast Inter-Ap Handoff Using Predictive Authentication Scheme in a Public Wireless LAN. *In In Proceedings of IEEE Networks Conference (confunction of IEEE ICN and IEEE ICWLHN*, 2002. 41

[40] Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 6 : Medium Access Control (MAC) Security Enhancements, 2004. 42

[41] Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 2 : Fast Basic Service Set (BSS) Transition, 2008. 43

[42] M Isabel SANCHEZ et Azzedine BOUKERCHE : On ieee 802.11 k/r/v amendments : Do they have a real impact ? *IEEE Wireless Communications*, 23(1):48–55, 2016. 43

[43] Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 5 : Enhancements for Higher Throughput, 2009. 44

[44] Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 4 : Enhancements for Very High Throughput for Operation in Bands below 6 GHz, 2013. 44

[45] U-LTE : Unlicensed Spectrum Utilization of LTE. http://www.huawei.com/ilink/en/download/HW_327803, 2014. (White Paper). 44

[46] Extending LTE Advanced to unlicensed spectrum. https://www.qualcomm.com/documents/extending-benefits-lte-advanced-unlicensed-spectrum, December 2013. (White Paper). 44

[47] Real-time LTE/Wi-Fi Coexistence Testbed. http://www.ni.com/white-paper/53044/en/, February 2016. (White Paper). 44

[48] Hwan-Joon KWON, Jeongho JEON, Abhijeet BHORKAR, Qiaoyang YE, Hiroki HARADA, Yu JIANG, Liu LIU, Satoshi NAGATA, Boon Loong NG, Thomas NOVLAN *et al.* : Licensed-Assisted Access to Unlicensed Spectrum in LTE Release 13. *IEEE Communications Magazine*, 55(2):201–207, 2017. 44

[49] Extending LTE Advanced to unlicensed spectrum. https://www.qualcomm.com/documents/lte-unlicensed-coexistence-whitepaper, June 2014. (White Paper). 44

[50] Amitav MUKHERJEE, Jung-Fu CHENG, Sorour FALAHATI, Havish KOORAPATY, Reem KARAKI, Laetitia FALCONETTI, Daniel LARSSON *et al.* : Licensed-assisted access LTE : Coexistence with IEEE 802.11 and the evolution toward 5G. *IEEE Communications Magazine*, 54(6):50–57, 2016. 45, 46

[51] Erika ALMEIDA, André M CAVALCANTE, Rafael CD PAIVA, Fabiano S CHAVES, Fuad M ABINADER, Robson D VIEIRA, Sayantan CHOUDHURY, Esa TUOMAALA et Klaus DOPPLER : Enabling LTE/WiFi coexistence by LTE blank subframe allocation. *In Communications (ICC), 2013 IEEE International Conference on*, pages 5083–5088. IEEE, 2013. 45

[52] ALCATEL-LUCENT AND ERICSSON AND LG ELECTRONICS AND QUALCOMM TECHNOLOGIES INC. AND SAMSUNG ELECTRONICS AND VERIZON : Lte-u csat procedure ts v1.0, October 2015. 45

[53] Eugene CHAI, Karthik SUNDARESAN, Mohammad A KHOJASTEPOUR et Sampath RANGARAJAN : LTE in unlicensed spectrum : are we there yet ? *In Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pages 135–148. ACM, 2016. 45, 46

[54] C CAPRETTI, Francesco GRINGOLI, Nicolò FACCHI et Paul PATRAS : LTE/Wi-Fi co-existence under scrutiny : an empirical study. *In WiNTECH@ MobiCom*, pages 33–40, 2016. 46

[55] Cheng CHEN, Rapeepat RATASUK et Amitava GHOSH : Downlink performance analysis of LTE and WiFi coexistence in unlicensed bands with a simple listen-before-talk scheme. *In Vehicular Technology Conference (VTC Spring), 2015 IEEE 81st*, pages 1–5. IEEE, 2015. 46

[56] Sasha SIROTKIN : LTE-WLAN Aggregation (LWA) : Benefits and Deployment Considerations. https://www.intel.com/content/www/us/en/wireless-network/lte-wlan-aggregation-deployment-white-paper.html. (White Paper). 46

[57] 3GPP : LTE ; Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN) ; Overall description ; Stage 2. Technical Specification (TS) 36.300, 3rd Generation Partnership Project (3GPP), 04 2016. URL https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2430. Version 13.3.0. 46

[58] 3GPP : Evolved Universal Terrestrial Radio Access (E-UTRA) ; Packet Data Convergence Protocol (PDCP) specification. Technical Specification (TS) 36.323, 3rd Generation Partnership Project (3GPP), 09 2017. URL https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2439. Version 14.4.0. 46

[59] 3GPP : Evolved Universal Terrestrial Radio Access (E-UTRA) ; LTE-WLAN Aggregation Adaptation Protocol (LWAAP) specification. Technical Specification (TS) 36.360, 3rd Generation Partnership Project (3GPP), 04 2017. URL https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3035. Version 14.0.0. 46

[60] 3GPP : Evolved Universal Terrestrial Radio Access Network (E-UTRAN) and Wireless LAN (WLAN) ; Xw interface user plane protocol. Technical Specification (TS) 36.465, 3rd Generation Partnership Project (3GPP), 04 2016. URL https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2949. Version 13.0.0. 46

[61] Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 6 : Wireless Access in Vehicular Environments, 2010. 46

[62] Daniel JIANG et Luca DELGROSSI : IEEE 802.11 p : Towards an international standard for wireless access in vehicular environments. *In Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, pages 2036–2040. IEEE, 2008. 46

[63] David ECKHOFF, Nikoletta SOFRA et Reinhard GERMAN : A performance study of cooperative awareness in ETSI ITS G5 and IEEE WAVE. *In Wireless On-demand Network Systems and Services (WONS), 2013 10th Annual Conference on*, pages 196–200. IEEE, 2013. 46

[64] Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 8 : Medium Access Control (MAC) Quality of Service Enhancements, 2005. 47

[65] Jungwook CHOI et Hyukjoon LEE : Supporting handover in an IEEE 802.11 p-based wireless access system. *In Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking*, pages 75–80. ACM, 2010. 47

[66] Wi-Fi Direct. https://www.wi-fi.org/discover-wi-fi/wi-fi-direct. 48

[67] Daniel CAMPS-MUR, Andres GARCIA-SAAVEDRA et Pablo SERRANO : Device-to-device communications with Wi-Fi Direct : overview and experimentation. *IEEE wireless communications*, 20(3):96–104, 2013. 48

[68] Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 9 : Interworking with External Networks, 2011. 48

[69] How Interworking Works : A Detailed Look at 802.11u and Hotspot 2.0 Mechanisms. https://ruckus-www.s3.amazonaws.com/pdf/wp/wp-how-interworking-works.pdf, July 2013. (White Paper). 48

[70] Hotspot 2.0 : Accelerating the Move to Automatic and Secure Roaming. https://ruckus-www.s3.amazonaws.com/pdf/wp/wp-hotspot-2.0.pdf. (White Paper). 48

[71] Hotspot 2.0 : Passing Go. https://wia.org/wp-content/uploads/Research/iGR_PCIA_HotSpot2_Whitepaper.pdf, 2012. (White Paper). 48

[72] Wi-Fi Certified Passpoint Architecture for Public Access. https://www.arubanetworks.com/assets/wp/WP_Passpoint_Wi-Fi.pdf. (White Paper). 48

[73] Vladimir LAVRUKHIN : An overhead analysis of Access Network Query Protocol (ANQP) in hotspot 2.0 Wi-Fi networks. *In ITS Telecommunications (ITST), 2013 13th International Conference on*, pages 266–271. IEEE, 2013. 49

[74] Wenchao XU, Haibo ZHOU, Yuanguo BI, Nan CHENG, Xuemin SHEN, Lakshmi THANAYANKIZIL et Fan BAI : Exploiting Hotspot-2.0 for Traffic Offloading in Mobile Networks. *IEEE Network*, (99):1–7, 2018. 49

[75] Sahar HOTEIT, Stefano SECCI, Guy PUJOLLE, Sven WIELTHOLTER, Adam WOLISZ, Cezary ZIEMLICKI et Zbigniew SMOREDA : Quantifying the achievable cellular traffic offloading gain with passpoint hotspots. *In Proceedings of the 2014 ACM international workshop on Wireless and mobile technologies for smart cities*, pages 19–28. ACM, 2014. 50

[76] Ubiquiti UniFi - Fast Roaming. https://help.ubnt.com/hc/en-us/articles/115004662107. 50

[77] Yan GRUNENBERGER et Franck ROUSSEAU : Virtual access points for transparent mobility in wireless LANs. *In Wireless Communications and Networking Conference (WCNC), 2010 IEEE*, pages 1–6. IEEE, 2010. 50

[78] Maria Eugenia BEREZIN, Franck ROUSSEAU et Andrzej DUDA : Multichannel virtual access points for seamless handoffs in IEEE 802.11 wireless networks. *In Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, pages 1–5. IEEE, 2011. 50

[79] Anatolij ZUBOW, Sven ZEHL et Adam WOLISZ : BIGAP—Seamless handover in high performance enterprise IEEE 802.11 networks. *In Network Operations and Management Symposium (NOMS), 2016 IEEE/IFIP*, pages 445–453. IEEE, 2016. 50

[80] Yi-Cheng CHAN et Dai-Jiong LIN : The design of an ap-based handoff scheme for ieee 802.11 wlans. *International Journal of e-Education, e-Business, e-Management and e-Learning*, 4(1):72, 2014. 51

[81] C. PERKINS : IP Mobility Support for IPv4, Revised. RFC 5944, RFC Editor, November 2010. URL http://www.rfc-editor.org/rfc/rfc5944.txt. http://www.rfc-editor.org/rfc/rfc5944.txt. 53

[82] G. MONTENEGRO : Reverse Tunneling for Mobile IP, revised. RFC 3024, RFC Editor, January 2001. 53

[83] C. PERKINS, D. JOHNSON et J. ARKKO : Mobility Support in IPv6. RFC 6275, RFC Editor, July 2011. URL http://www.rfc-editor.org/rfc/rfc6275.txt. http://www.rfc-editor.org/rfc/rfc6275.txt. 53

[84] Claude CASTELLUCCIA : HMIPv6 : A hierarchical mobile IPv6 proposal. *ACM SIGMOBILE Mobile Computing and Communications Review*, 4(1):48–59, 2000. 53

[85] H. SOLIMAN, C. CASTELLUCCIA, K. ELMALKI et L. BELLIER : Hierarchical Mobile IPv6 (HMIPv6) Mobility Management. RFC 5380, RFC Editor, October 2008. URL http://www.rfc-editor.org/rfc/rfc5380.txt. http://www.rfc-editor.org/rfc/rfc5380.txt. 53

[86] R. KOODLI : Mobile IPv6 Fast Handovers. RFC 5568, RFC Editor, July 2009. 53

[87] S. GUNDAVELLI, K. LEUNG, V. DEVARAPALLI, K. CHOWDHURY et B. PATIL : Proxy Mobile IPv6. RFC 5213, RFC Editor, August 2008. URL http://www.rfc-editor.org/rfc/rfc5213.txt. http://www.rfc-editor.org/rfc/rfc5213.txt. 53

[88] Xavier PÉREZ-COSTA, Marc TORRENT-MORENO et Hannes HARTENSTEIN : A performance comparison of Mobile IPv6, Hierarchical Mobile IPv6, fast handovers for Mobile IPv6 and their combination. *ACM SIGMOBILE Mobile Computing and Communications Review*, 7(4):5–19, 2003. 54

[89] Mohamed ALNAS, Irfan AWAN et RDW HOLTON : Performance evaluation of fast handover in mobile IPv6 based on link-layer information. *Journal of Systems and Software*, 83(10):1644–1650, 2010. 54

[90] Ki-Sik KONG, Wonjun LEE, Youn-Hee HAN, Myung-Ki SHIN et HeungRyeol YOU : Mobility management for all-IP mobile networks : mobile IPv6 vs. proxy mobile IPv6. *IEEE Wireless communications*, 15(2), 2008. 54

[91] R. MOSKOWITZ et P. NIKANDER : Host Identity Protocol (HIP) Architecture. RFC 4423, RFC Editor, May 2006. 54

[92] R. MOSKOWITZ, T. HEER, P. JOKELA et T. HENDERSON : Host Identity Protocol Version 2 (HIPv2). RFC 7401, RFC Editor, April 2015. URL http://www.rfc-editor.org/rfc/rfc7401.txt. http://www.rfc-editor.org/rfc/rfc7401.txt. 54

[93] Petri JOKELA, Teemu RINTA-AHO, Tony JOKIKYYNY, Jorma WALL, Martti KUPARINEN, Heikki MAHKONEN, Jan MELÉN, Tero KAUPPINEN et Jouni KORHONEN : Handover performance with HIP and MIPv6. *In 1st International Symposium on Wireless Communication Systems*, volume 3, pages 324–328, 2004. 54

[94] A. FORD, C. RAICIU, M. HANDLEY et O. BONAVENTURE : TCP Extensions for Multipath Operation with Multiple Addresses. RFC 6824, RFC Editor, January 2013. URL http://www.rfc-editor.org/rfc/rfc6824.txt. http://www.rfc-editor.org/rfc/rfc6824.txt. 55

[95] Costin RAICIU, Christoph PAASCH, Sebastien BARRE, Alan FORD, Michio HONDA, Fabien DUCHENE, Olivier BONAVENTURE et Mark HANDLEY : How hard can it be ? designing and implementing a deployable multipath TCP. *In Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, pages 29–29. USENIX Association, 2012. 55

[96] Christoph PAASCH, Gregory DETAL, Fabien DUCHENE, Costin RAICIU et Olivier BONAVENTURE : Exploring mobile/WiFi handover with multipath TCP. *In Proceedings of the 2012 ACM SIGCOMM workshop on Cellular networks : operations, challenges, and future design*, pages 31–36. ACM, 2012. 55

[97] Quentin DE CONINCK et Olivier BONAVENTURE : Observing Network Handovers with Multipath TCP. *In Proceedings of the ACM SIGCOMM 2018 Conference on Posters and Demos*, pages 54–56. ACM, 2018. 56

[98] Quentin DE CONINCK, Matthieu BAERTS, Benjamin HESMANS et Olivier BONAVENTURE : A first analysis of multipath tcp on smartphones. *In International Conference on Passive and Active Network Measurement*, pages 57–69. Springer, 2016. 56

[99] Q DE CONICK et Olivier BONAVENTURE : Every millisecond counts : Tuning Multipath TCP for interactive applications on smartphones. Rapport technique, Technical report. Available at http ://hdl. handle. net/2078.1/185717, 2017. 56

[100] L. ONG et J. YOAKUM : An Introduction to the Stream Control Transmission Protocol (SCTP). RFC 3286, RFC Editor, May 2002. 56

[101] R. STEWART : Stream Control Transmission Protocol. RFC 4960, RFC Editor, September 2007. URL http://www.rfc-editor.org/rfc/rfc4960.txt. http://www.rfc-editor.org/rfc/rfc4960.txt. 56

[102] R. STEWART, Q. XIE, M. TUEXEN, S. MARUYAMA et M. KOZUKA : Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration. RFC 5061, RFC Editor, September 2007. 56

[103] Seok Joo KOH, Moon Jeong CHANG et Meejeong LEE : mSCTP for soft handover in transport layer. *IEEE communications letters*, 8(3):189–191, 2004. 56

[104] Li MA, Fei YU, Victor CM LEUNG et Tejinder RANDHAWA : A new method to support UMTS/WLAN vertical handover using SCTP. *IEEE Wireless communications*, 11(4):44–51, 2004. 56

[105] Andrew KELLY, Gabriel MUNTEAN, Philip PERRY et John MURPHY : Delay-centric handover in SCTP over WLAN. *Transactions on Automatic Control and Computer Science*, 49(63):1–6, 2004. 56

[106] Jana IYENGAR et Martin THOMSON : QUIC : A UDP-Based Multiplexed and Secure Transport. Internet-Draft draft-ietf-quic-transport-15, Internet Engineering Task Force, octobre 2018. URL https://datatracker.ietf.org/doc/html/draft-ietf-quic-transport-15. Work in Progress. 56

[107] Adam LANGLEY, Alistair RIDDOCH, Alyssa WILK, Antonio VICENTE, Charles KRASIC, Dan ZHANG, Fan YANG, Fedor KOURANOV, Ian SWETT, Janardhan IYENGAR *et al.* : The QUIC transport protocol : Design and Internet-scale deployment. *In Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 183–196. ACM, 2017. 56

[108] Apple Developers Guidelines : About Networking. https://developer.apple.com/library/archive/documentation/NetworkingInternetWeb/Conceptual/NetworkingOverview/WhyNetworkingIsHard/WhyNetworkingIsHard.html#//apple_ref/doc/uid/TP40010220-CH13-SW2, . 57

[109] Apple Push Notification service. https://developer.apple.com/library/archive/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/APNSOverview.html#//apple_ref/doc/uid/TP40008194-CH8-SW1, . 57

[110] Firebase Cloud Messaging. https://firebase.google.com/docs/cloud-messaging/. 57

[111] Amazon Simple Notification Service. https://docs.aws.amazon.com/sns/latest/dg/welcome.html/. 57

[112] Azure Notification Hubs. https://docs.microsoft.com/fr-fr/azure/notification-hubs/. 57

[113] Karl-Johan PERSSON, Fredric VINNÅ, Stig Gustav Viktor SÖDERSTRÖM et Pär BOHRARPER : Method and a media device for pre-buffering media content streamed to the media device from a server system, octobre 17 2017. US Patent 9,794,309. 57

[114] Mikael OLENFALK : System and method for early media buffering using prediction of user behavior, novembre 30 2017. US Patent App. 15/681,191. 57

[115] John FUNGE et Greg PETERS : Pre-buffering audio streams, avril 28 2015. US Patent 9,021,537. 58

[116] Kevin FALL et Stephen FARRELL : DTN : an architectural retrospective. *IEEE Journal on Selected areas in communications*, 26(5), 2008. 58

[117] Alex MCMAHON et Stephen FARRELL : Delay-and disruption-tolerant networking. *IEEE Internet Computing*, 13(6), 2009. 58

[118] V. CERF, S. BURLEIGH, A. HOOKE, L. TORGERSON, R. DURST, K. SCOTT, K. FALL et H. WEISS : Delay-Tolerant Networking Architecture. RFC 4838, RFC Editor, April 2007. URL http://www.rfc-editor.org/rfc/rfc4838.txt. http://www.rfc-editor.org/rfc/rfc4838.txt. 58

[119] K. SCOTT et S. BURLEIGH : Bundle Protocol Specification. RFC 5050, RFC Editor, November 2007. URL http://www.rfc-editor.org/rfc/rfc5050.txt. http://www.rfc-editor.org/rfc/rfc5050.txt. 58

[120] CC SOBIN, Vaskar RAYCHOUDHURY, Gustavo MARFIA et Ankita SINGLA : A survey of routing and data dissemination in delay tolerant networks. *Journal of Network and Computer Applications*, 67:128–146, 2016. 58

[121] Elizabeth M DALY et Mads HAAHR : Social network analysis for information flow in disconnected delay-tolerant MANETs. *IEEE Transactions on Mobile Computing*, (5):606–621, 2008. 58

[122] Khaled A HARRAS, Kevin C ALMEROTH et Elizabeth M BELDING-ROYER : Delay tolerant mobile networks (dtmns) : Controlled flooding in sparse mobile networks. *In International Conference on Research in Networking*, pages 1180–1192. Springer, 2005. 58

[123] Laurent FRANCK et Felipe GIL-CASTINEIRA : Using delay tolerant networks for car2car communications. *In Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on*, pages 2573–2578. IEEE, 2007. 58

[124] Arkko JARI : Analysing IP Mobility Protocol Deployment Difficulties. IETF meeting #83, March 2012. URL https://datatracker.ietf.org/meeting/83/materials/slides-83-mobopts-0. 58

[125] Kishore RAMACHANDRAN : Mobile IP-deployment after a decade. *Rutgers University*, 2006. 58

[126] T. HENDERSON et A. GURTOV : The Host Identity Protocol (HIP) Experiment Report. RFC 6538, RFC Editor, March 2012. 58

[127] Olivier BONAVENTURE et SungHoon SEO : Multipath TCP Deployments. IETF Journal, November 2016. URL https://www.ietfjournal.org/multipath-tcp-deployments/. 58

[128] Créer des connexions de secours pour iOS via le protocole Multipath TCP. https://support.apple.com/fr-fr/HT201373, . 58

[129] SungHoon SEO : KT's GiGA LTE - Mobile MPTCP Proxy Deployment. IETF meeting #97, November 2016. URL https://www.ietf.org/proceedings/97/slides/slides-97-banana-kt-giga-lte-mobile-mptcp-proxy-development-01.pdf. 58

[130] M. BELSHE, R. PEON et M. THOMSON : Hypertext Transfer Protocol Version 2 (HTTP/2). RFC 7540, RFC Editor, May 2015. URL http://www.rfc-editor.org/rfc/rfc7540.txt. http://www.rfc-editor.org/rfc/rfc7540.txt. 58

[131] Volley overview. https://developer.android.com/training/volley/, . 58

[132] Syed Faraz HASAN, Nazmul SIDDIQUE et Shyam CHAKRABORTY : *Intelligent transport systems : 802.11-based roadside-to-vehicle communications*. Springer Science & Business Media, 2012. 60

[133] Przemyslaw MACHAN et Józef WOZNIAK : Performance evaluation of IEEE 802.11 fast BSS transition algorithms. *In WMNC2010*, pages 1–5. IEEE, 2010. 60

[134] Nicolas MONTAVONT et Thomas NOËL : Anticipated handover over IEEE 802.11 networks. *In WiMob'2005), IEEE International Conference on Wireless And Mobile Computing, Networking And Communications, 2005.*, volume 2, pages 64–71. IEEE, 2005. 60

[135] David KOTZ, Calvin C. NEWPORT, Robert S. GRAY, Jason LIU, Yougu YUAN et Chip ELLIOTT : Experimental Evaluation of Wireless Simulation Assumptions. *In Proc. of MSWiM*, 2004. 64

[136] Kefeng TAN, Daniel WU, An (Jack) CHAN et Prasant MOHAPATRA : Comparing Simulation Tools and Experimental Testbeds for Wireless Mesh Networks. *Pervasive and Mobile Computing*, 7(4), 2011. 64

[137] Barry N. TAYLOR et Chris E. KUYATT : Guidelines for Evaluating and Expressing the Uncertainty of NIST Measurement Results. NIST Technical Note 1297, National Institute of Standards and Technology, 1994. 64

[138] Dipankar RAYCHAUDHURI, Maximilian OTT et Ivan SESKAR : ORBIT Radio Grid Tested for Evaluation of Next-Generation Wireless Network Protocols. *In Proc. of TRIDENTCOM*, 2005. 64

[139] iMinds RESEARCH INSTITUTE : w-iLab.t Generic Wireless Testbeds. URL http://www.iminds.be/en/develop-test/ilab-t. 64

[140] Cedric ADJIH, Emmanuel BACCELLI, Eric FLEURY, Gaetan HARTER, Nathalie MITTON, Thomas NOEL, Roger PISSARD-GIBOLLET, Frederic SAINT-MARCEL, Guillaume SCHREINER, Julien VANDAELE et Thomas WATTEYNE : FIT IoT-LAB : A Large Scale Open Experimental IoT Testbed. *In Proc. of IEEE World Forum on Internet of Things*, 2015. 64

[141] Anne-Sophie TONNEAU, Nathalie MITTON et Julien VANDAELE : How to choose an experimentation platform for wireless sensor networks ? A survey on static and mobile wireless sensor network experimentation facilities. *Ad Hoc Networks*, 30:115–127, 2015. 64

[142] WalT resources. URL http://walt.forge.imag.fr, https://github.com/drakkar-lig and https://hub.docker.com/u/waltplatform/. 64

[143] C. BORMANN, M. ERSUE et A. KERANEN : Terminology for Constrained-Node Networks. RFC 7228 (Informational), mai 2014. 66

[144] Das U-Boot. URL https://www.denx.de/wiki/U-Boot. 68

[145] E. DUBLÉ : Debootstick. URL https://github.com/drakkar-lig/debootstick. 70

[146] Pierre BRUNISHOLZ, Etienne DUBLÉ, Franck ROUSSEAU et Andrzej DUDA : WalT : A reproducible testbed for reproducible network experiments. *In Computer Communications Workshops (INFOCOM WKSHPS), 2016 IEEE Conference on*, pages 146–151. IEEE, 2016. 70

[147] Olivier MESNARD et Lorena A BARBA : Reproducible and replicable CFD : it's harder than you think. *arXiv preprint arXiv :1605.04339*, 2016. 73, 74

[148] Connectivity : Broadband market developments in the EU. http://ec.europa.eu/newsroom/document.cfm?doc_id=44389, 2017. (Report). 80

[149] Arsham FARSHAD, Mahesh K MARINA et Francisco GARCIA : Urban WiFi characterization via mobile crowdsensing. *In 2014 IEEE Network Operations and Management Symposium (NOMS)*, pages 1–9. IEEE, 2014. 80

[150] German CASTIGNANI, Lucien LOISEAU et Nicolas MONTAVONT : An evaluation of IEEE 802.11 community networks deployments. *In The International Conference on Information Networking 2011 (ICOIN2011)*, pages 498–503. IEEE, 2011. 80

[151] Vladimir BYCHKOVSKY, Bret HULL, Allen MIU, Hari BALAKRISHNAN et Samuel MADDEN : A measurement study of vehicular internet access using in situ Wi-Fi networks. *In Proceedings of the 12th annual international conference on Mobile computing and networking*, pages 50–61. ACM, 2006. 80

[152] German CASTIGNANI, Alejandro LAMPROPULOS, Alberto BLANC et Nicolas MONTAVONT : Wi2me : a mobile sensing platform for wireless heterogeneous networks. *In Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on*, pages 108–113. IEEE, 2012. 81

[153] Pierre ROUVEYROL, Patrice RAVENEAU et Mathieu CUNCHE : Large scale Wi-Fi tracking using a botnet of wireless routers. *In SAT 2015-Workshop on Surveillance & Technology*, 2015. 82

[154] QGIS. URL https://www.qgis.org/fr/site/. 82

[155] Maria Eugenia BEREZIN, Franck ROUSSEAU et Andrzej DUDA : Citywide mobile internet access using dense urban WiFi coverage. *In Proceedings of the first workshop on Urban networking*, pages 31–36. ACM, 2012. 83

[156] Google Maps API, . URL https://developers.google.com/maps/documentation/directions/intro. 83

[157] ITUT ITU-T : Recommendation G. 114. *One-Way Transmission Time, Standard G*, 114, 2003. 84

[158] German CASTIGNANI, Juan MONETTI, Nicolas MONTAVONT, Andrés ARCIA-MORET, Raphaël FRANK et Thomas ENGEL : A study of urban IEEE 802.11 hotspot networks : towards a community access network. *In 2013 IFIP Wireless Days (WD)*, pages 1–8. IEEE, 2013. 92

[159] IEEE. Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Std. 802.11, 2012. 96

[160] Internet Protocols Measurement Tools. URL https://gricad-gitlab.univ-grenoble-alpes.fr/Drakkar-LIG/ipmt. 97, 101

[161] R.K. GANTI, Fan YE et Hui LEI : Mobile Crowdsensing : Current State and Future Challenges. *Communications Magazine, IEEE*, 49(11):32–39, Nov 2011. ISSN 0163-6804. 114

[162] Ethan KATZ-BASSETT, John P JOHN, Arvind KRISHNAMURTHY, David WETHERALL, Thomas ANDERSON et Yatin CHAWATHE : Towards IP Geolocation Using Delay and Topology Measurements. *In Proc. of ACM IMC*, 2006. 116

[163] Mahesh BALAKRISHNAN, Iqbal MOHOMED et Venugopalan RAMASUBRAMANIAN : Where's That Phone ? : Geolocating IP Addresses on 3G Networks. *In Proc. 9th ACM SIGCOMM IMC Conference*, pages 294–300, 2009. 116

[164] Georgios KARAGIANNIS, Geert HEIJENK, Andreas FESTAG, Alexandru PETRESCU et Alison CHAIKEN : Internet-wide Geo-networking Problem Statement. Internet-Draft draft-karagiannis-problem-statement-geonetworking-01, Internet Engineering Task Force, mai 2014. URL https://tools.ietf.org/html/draft-karagiannis-problem-statement-geonetworking-01. Work in Progress. 116

[165] Thomas STRANG, Armin DAMMANN, Matthias RÖCKL et Simon PLASS : Using Gray Codes as Location Identifiers. *In Proc. of 6. GI/ITG KuVS Fachgesprach Ortsbezogene Anwendungen und Dienste*, 2009. URL http://elib.dlr.de/60489. 118

[166] Davide PESAVENTO, Giulio GRASSI, Claudio E. PALAZZI et Giovanni PAU : A Naming Scheme to Represent Geographic Areas in NDN. *In Wireless Days*, pages 1–3. IEEE, 2013. 118, 120

[167] Guy M MORTON : *A Computer Oriented Geodetic Data Base and a New Technique in File Sequencing*. IBM, 1966. 118

[168] Mohamed BOUCADAIR et Stig VENAAS : Updates to the IPv6 Multicast Addressing Architecture. RFC 7371, 2014. 119

[169] Thomas O'DANIEL, Mohammed Nazar HUSSEIN et Raed ABDULLA : Localization using GPS Coordinates in IPv6 Addresses of Wireless Sensor Network Nodes. *Indian Journal of Science and Technology*, 2016. 120

[170] B. FENNER, M. HANDLEY, H. HOLBROOK et I. KOUVELAS : Protocol Independent Multicast - Sparse Mode (PIM-SM) : Protocol Specification (Revised). RFC 4601, RFC Editor, August 2006. 121

[171] Sylvia RATNASAMY, Andrey ERMOLINSKIY et Scott SHENKER : Revisiting IP Multicast. *In Proc. of ACM SIGCOMM*, 2006. 121