Communication unicast dans les réseaux mobiles dynamiques

Farah El Ali

Université de Technologie de Compiègne
Laboratoire Heudiasyc – UMR CNRS 7253

Communication unicast dans les réseaux mobiles dynamiques

Thèse

pour obtenir le grade de DOCTEUR de l'Université de Technologie de Compiègne
préparée au laboratoire Heudiasyc,
dans le cadre de l'Ecole Doctorale Technologies de l'Information et de Systèmes
Spécialité : Génie Informatique

Farah EL ALI

04 décembre 2012

JURY :

M. Marcelo DIAS DE AMORIM	Directeur de recherche CNRS LIP6, Paris	Rapporteur
M. Vincent VILLAIN	Professeur, UPJV, Amiens	Rapporteur
M. Abdelmadjid BOUABDALLAH	Professeur, UTC, Compiègne	Examinateur
M. Vania CONAN	Directeur de recherche, Thalès, Colombes	Examinateur
M. Javier IBANEZ-GUZMAN	Ingénieur de recherche, Renault	Examinateur
Mme. Eva CRÜCK	Responsable-adjoint, DGA	Examinateur
M. Bertrand DUCOURTHIAL	Professeur, UTC, Compiègne	Directeur

Remerciements

Trois années ont pris fin avec ce manuscrit.
Je tiens en premier lieu à remercier Monsieur Bertrand Ducourthial, professeur à l'Université de Technologie de Compiègne, pour son encadrement lors de ces trois années, pour m'avoir confié ce travail de recherche, ainsi que pour son aide et ses remarques et conseils au cours de ces plusieurs années. Je remercie Monsieur Ali Charara, directeur du laboratoire, qui m'a ouvert les portes du laboratoire Heudiasyc.

Je tiens à remercier Monsieur Marcelo Dias de Amorim, Directeur de recherche au LIP6, et Monsieur Vincent Villain, professeur à l'UPJV, d'avoir accepté d'être les rapporteurs de ma thèse.

Je remercie de même les membres du jury, Messieurs Abdelmadjid Bouabdallah, Vania Conan, Javier Ibanez-Guzman, et Madame Eva Crück, d'avoir accepté de participer à ma soutenance de thèse.

Je remercie tous les membres de Heudiasyc d'avoir fait partie de mon parcourt durant ces 3 années.

Je remercie mes collègues de bureau pour les discussions intéressantes, les pauses cafés, les pauses desserts, donc merci Ada, Adam, Bin, Clément, Felipe, Gilles, Hoda, Ji, Julien, Luis, Marek, Nicole, Oussama, Sawsan, Vincent... Et merci à plein d'autres personnes que j'ai oublié de citer.

Mes pensées les plus fortes vont à ma famille, Neemat, Kamal, Maha, Aida, Hala, Roula, Frederic, Camille, Constance, Samuel, Emmanuel, David, Khalida.

Un grand merci à ma deuxième famille, Ali, Farah, Jessy, Joseph, Momo, Nicole, Oussama et Zahra.
Et surtout à Bassam qui a du supporter tout le stress de la thèse avec moi.

Résumé

Les communications sont difficiles à maintenir dans les réseaux informatiques dès lors qu'ils sont dynamiques. Les réseaux de véhicules sont un exemple direct de ces réseaux ad hoc dynamiques. Ils font l'objet des développements et des recherches pour les systèmes intelligents pour le transport (ITS). Dans ce manuscrit, nous nous intéressons aux communications unicast dans les réseaux ad hoc dynamiques. Les travaux réalisés à ce sujet sont appliqués dans les réseaux véhiculaires.

Pour mieux comprendre ces réseaux, nous avons effectué des tests sur route afin d'analyser les performances et les capacités de ces réseaux. Les résultats de ces tests nous ont permis de proposer des améliorations aux communications véhicules à infrastructure (V2I) et véhicules à véhicules (V2V). Vue l'étude des performances, nous avons opté pour une architecture opportuniste pour les communications V2I. Elle permet la remontée des données du réseau véhiculaire vers l'infrastructure via une passerelle.

Pour l'autre type de communications dans un réseau mobile, les communications V2V, la source et la destination sont mobiles. La communication est alors menacée d'être interrompue. Nous proposons un algorithme de maintien de chemin qui garantit l'acheminement des messages entre les deux entités en mouvement dans le réseau. Cet algorithme utilise les échanges locaux pour ajuster le chemin, et s'affranchit ainsi de la dynamique du réseau. L'algorithme proposé est implémenté, testé et validé pour les réseaux de véhicules. Cependant, il pourrait être appliqué à d'autres types de réseaux, comme les réseaux de drones ou de piétons.

Pour mieux comprendre les limites du routage en général et de notre algorithme de maintien de chemin en particulier, nous utilisons l'approche « best effort » qui formalise un compromis entre la dynamique du réseau et les propriétés d'un algorithme. Nous introduisons les graphes p-dynamiques pour caractériser la dynamique. Ils permettent alors d'exprimer une propriété dite topologique, qui est nécessaire pour garantir une propriété dite de continuité du service offert par l'algorithme. Nous proposons ainsi, grâce aux graphes p-dynamiques, une nouvelle approche pour modéliser la dynamique des réseaux, et pour exprimer les caractéristiques attendues d'un algorithme en fonction de cette dynamique. Cette approche constitue un premier pas vers une métrique algorithmique de la dynamique des réseaux. Elle permet de vérifier si un algorithme donné peut satisfaire ses spécifications dans un réseau donné.

Par ailleurs, si les protocoles développés pour les communications *unicast* ont été implémentés et testés sur route, ils restent suffisamment généraux pour s'appliquer à d'autres réseaux.

Table des matières

Table des figures

Chapitre 1

Contexte global : les réseaux mobiles

Les réseaux mobiles sont formés à partir d'entités qui se déplacent dans l'espace. Leur mobilité engendre des caractéristiques différentes de celles connues pour les réseaux traditionnels. Ces caractéristiques représentent un vrai challenge pour les études des problèmes traités dans les réseaux comme la communication et le routage.

Les réseaux véhiculaires sont un exemple des réseaux mobiles. Les applications des systèmes intelligents pour le transport reposent très largement sur ces réseaux.

Dans ce chapitre nous présentons les réseaux mobiles et les travaux ITS, qui représentent le contexte global du travail, ainsi que le contexte local représenté par les travaux dans le laboratoire.

1.1 Les réseaux mobiles

Les réseaux mobiles, dits MANETs (*Mobile Ad hoc NETworks*), sont des réseaux dont la topologie change avec la mobilité de leurs entités.

Les réseaux ad hoc sont des réseaux communicants sans connexion filaire. Nous décrivons dans la suite ces réseaux.

1.1.1 Réseaux mobiles sans fil

Un réseau sans fil (*wireless network*) est un réseau informatique ou numérisé qui connecte différents postes ou systèmes entre eux par ondes radio. Il peut-être associé à un réseau de télécommunications pour réaliser des interconnexions entre nœuds.

La norme la plus utilisée actuellement pour les réseaux sans fil est la norme IEEE 802.11, mieux connue sous le nom de Wi-Fi. Leur rayonnement géographique, sans amplification ou l'utilisation de relais, permet une zone de couverture relativement limitée.

Même si ces réseaux offrent une alternative aux réseaux câblés, qui sont parfois plus coûteux (fibre optique par exemple), la mobilité de leurs nœuds et leur nature exigent quelques difficultés ; l'acheminement des données devient problématique, ainsi que la centralisation des informations en absence éventuelle d'infrastructure.

Nous rappelons les diverses architectures possibles des réseaux sans fil dans la sous-section suivante.

1.1.2 Architectures des réseaux sans fil

Nous pouvons classer les architectures des réseaux sans fil comme suit :
- les architectures à infrastructure et centralisées ;
- les architectures Mesh ;
- les architectures sans infrastructure et reparties.

Les architectures à infrastructure et centralisées

En ce mode, les nœuds mobiles communiquent entre eux en se connectant sur des sites fixes. A chaque station fixe correspond une zone géographique limitée à partir de laquelle des nœuds mobiles peuvent émettre et recevoir des messages. Pour communiquer entre elles, les stations fixes sont reliées par un réseau de communication filaire, généralement fiable et d'un débit élevé. Par ailleurs, la bande passante des liaisons sans fils étant limitée, le volume des informations échangées est sévèrement réduit.

Les architectures *Mesh*

L'architecture Mesh définit une architecture où tous les hôtes sont connectés de proche en proche sans hiérarchie centrale. Les nœuds doivent être capables de relayer les données d'autres nœuds. Ce type d'architecture permet d'éviter la coupure de la connexion entre les différentes unités en raison d'absence de points sensibles qui existent dans l'architecture avec infrastructure, et qui, en cas de panne, entraînent la coupure de

connexion sur une partie du réseau. Un réseau Mesh peut contenir à la fois des stations mobiles et des stations fixes. Son architecture offre un déploiement rapide et simplifié, une grande évolution de la couverture et, en raison du maillage, une forte tolérance aux pannes et aux interférences. Cette caractéristique permet de réduire significativement les coûts d'installation et d'exploitation des réseaux.

Les architectures sans infrastructure et réparties

Le réseau mobile sans infrastructure, appelé réseau ad hoc, ne comporte aucune station fixe. Toutes les unités sont mobiles et donc il n'existe pas d'infrastructure. L'absence de cette dernière oblige les unités mobiles à être autonomes. Elles sont obligées à se comporter comme des routeurs qui découvrent et maintiennent des chemins pour les autres hôtes du réseau. Ces réseaux doivent donc s'organiser automatiquement afin d'être opérationnels dès qu'ils sont mis en place [11]. Ces réseaux ne sont pas limités par leur taille en termes d'étendue ou de nombre d'unités contrairement aux réseaux centralisés qui dépendent de la cellule couverte par la station de base.

Les intérêts des architectures *ad hoc*

Une caractéristique particulière du réseau ad hoc est l'absence de toute installation fixe. Ceci lui permet d'être rapide et facile à déployer. Pour cela, le réseau ad hoc est utilisé pour les applications tactiques comme les opérations de secours, militaires ou d'explorations. Cette technologie a donné lieu à plusieurs applications civiles. Parmi les applications bien connues du réseau ad hoc, nous en citons la gestion des crises par exemple. Quand l'infrastructure des réseaux de communication est touchée, restaurer la communication devient essentiel. En utilisant le réseau ad hoc, une infrastructure (formée à partir des nœuds du réseau) peut être construite en quelques heures au lieu de plusieurs jours ou semaines pour récupérer la communication filaire. Les réseaux PAN (*Personal Area Networking*) ou Bluetooth sont d'autres exemples des architectures ad hoc.

Nous pouvons alors citer quelques unes de leurs caractéristiques :
- une topologie dynamique ;
- une bande passante limitée ;
- des contraintes d'énergie ;
- une sécurité physique limitée ;
- l'absence d'infrastructure.

1.1.3 Réseaux véhiculaires

Les réseaux *ad hoc* de véhicules (*Vehicular Ad-Hoc Network* ou VANET) sont une application des réseaux ad hoc mobiles (MANET) sur un réseau véhiculaire, pour des communications au sein d'un groupe de véhicules à portée les uns des autres et entre les véhicules et les équipements fixes à portée, usuellement appelés équipements de la route.

Les réseaux de véhicules ont des caractéristiques semblables à ceux des réseaux MANETs. La communication se fait souvent en multi-sauts. Des changements de topologie du réseau arrivent fréquemment en raison de la haute mobilité des nœuds. Tous les nœuds partagent le même canal menant à la congestion dans des réseaux très denses. La nature décentralisée de ces réseaux mène au besoin de nouveaux systèmes et protocoles de diffusion de l'information. De plus, la nature fragile des liens de communication et le changement continu de voisinage exige de nouvelles approches pour la sécurité de la communication ; les nœuds doivent avoir confiance en leurs voisins (surtout les nouveaux arrivants) et dans les données échangées avec ces voisins ou relayés par ces derniers.

Les réseaux véhiculaires sont un domaine d'application des systèmes de transports intelligents (Intelligent Transportation Systems - ITS). Les véhicules communiquent les uns avec les autres par l'intermédiaire de la communication inter-véhicule (Vehicle To Vehicle communication - V2V) aussi bien avec les équipements de la route par l'intermédiaire de la communication de véhicule à équipement (Vehicle to Infrastructure communication - V2I).

Il est attendu que les réseaux véhiculaires contribueront à des routes plus sûres et plus efficaces à l'avenir en fournissant des informations opportunes aux conducteurs, aux passagers intéressés et à l'infrastructure.

Les communications Véhicule à Véhicule - V2V

Les services et les applications qui sont basées sur la simple communication inter-véhicule et n'impliquant pas d'infrastructure fonctionnent seulement dans le cas où le nombre de voitures est suffisant pour que les communications soient efficaces.

Ces communications se font directement si les véhicules communiquant sont à portées l'un de l'autre, et par multi-sauts au cas où les véhicules communiquant sont distants de plusieurs sauts. Dans ce cas, les véhicules du réseau servent de relais à la communication.

Les communications Véhicule à Infrastructure - V2I

Dans cette thèse nous prenons aussi en compte des applications qui utilisent des points d'infrastructure (*road side units* ou RSU).

Des services à base d'infrastructure (accès à internet, échange de données par exemple de voiture-à-domicile, communications de voiture-à-garage pour le diagnostic distant...) sont offerts aux clients et peuvent motiver des conducteurs à investir dans l'équipement sans fil supplémentaire pour leurs véhicules.

Les communications Hybrides

La combinaison de ces deux types de communications, V2V et V2I, permet d'obtenir une communication hybride très intéressante. En effet, les portées des infrastructures étant limitées, l'utilisation de véhicules comme relai permet d'étendre cette distance. Dans un but économique en évitant de multiplier les bornes à chaque coin de rue, l'utilisation de sauts par véhicules intermédiaires prend toute son importance.

1.1.4 Applications dans les réseaux véhiculaires

Dans le cadre des travaux dans le domaine des *Systèmes Intelligents pour le Transport* (en anglais *ITS*, pour Intelligent Transportation Systems), de nombreuses applications sont envisagées et étudiées. Elles concernent principalement des améliorations au niveau de la sécurité routière, de la régulation du trafic, de la remontée des données capteurs, de l'intégration de nouveaux services à bord (*chat*, détection de piétons...), et bien d'autres apports qui peuvent servir à améliorer les services à bord d'un véhicule et le contrôle de la sécurité routière.

Les types d'applications

Les travaux de recherche effectués dans le domaine des réseaux mobiles concernent en général tous les types de réseaux, que ce soit des réseaux de drones, de passagers, ou de véhicules. Les applications résultant de ces recherches sont alors réparties sur quatre catégories [29] :

- applications orientées véhicule ;
- applications orientées conducteur ;
- applications orientées passager ;
- applications orientées infrastructure.

La première catégorie fournit des informations aux véhicules pour adapter leurs automatismes, comme par exemple le freinage automatique ou la régulation de vitesse, entre autres applications. Ce type d'applications concerne principalement la sécurité routière. La seconde catégorie concerne les applications orientées conducteur. Il recevra alors des informations qui lui seront utiles pour son trajet, sa conduite ou la prévision des situations critiques lors du trajet. Ces applications communiqueront des informations sur le trajet, le trafic, les bouchons ou les accidents... La troisième famille comprend les applications orientées passager. Ces applications offrent des services de divertissement ou d'infotainment, en montrant que les réseaux de communications initialement dédiés à la sureté de conduite seront aussi utilisés pour assurer aux passagers des accès à Internet entre autres services de divertissement [14]. Enfin, la quatrième catégorie comprend les applications orientées infrastructure. Ces applications offriront un moyen pour mieux utiliser les ressources partagées (infrastructure routière, infrastructure de base de données). Cet usage se manifestera quand les opérateurs publics ou privés auront une information précise de leur état et de leur situation actuelle. Ils seront informés du temps de parcours, de la vitesse moyenne, des dépassements de vitesse ou des taux d'émissions de CO_2, un facteur qui peut être réduit à l'aide de conseils pertinents aux automobilistes.

Nous remarquons qu'un nombre important d'applications nécessitent de remonter ou d'obtenir des informations de l'infrastructure et des bases de données centralisées communicantes accessibles via des requêtes vers l'infrastructure. Les applications nécessitant une remontée d'informations vers l'infrastructure sont résumées par les appels d'urgence, les accès Internet, les remontées des informations capteurs depuis les véhicules vers l'infrastructure pour renseigner les opérateurs sur l'état de la route, etc. Les travaux futurs mêleront des communications véhicules vers infrastructure (*V2I*) à des communications

véhicules à véhicules ($V2V$), pour aboutir à des applications aussi performantes que possible. Les ressources de l'infrastructure seront largement sollicitées par les réseaux de véhicules. Par contre, cet échange entre l'infrastructure et le réseau véhiculaire s'avère coûteux. La sollicitation par un grand nombre de véhicules peut encombrer les ressources de l'infrastructure, d'autant plus que si le réseau véhiculaire est dynamique les requêtes changent de point d'accès et de type. Le déploiement devient donc difficile et coûteux. D'autre part, les équipements embarqués envisagés pour les véhicules doivent garder un coût limité, offrant la possibilité d'un déploiement à grande échelle, pour un service d'accès offert équitablement. Ce point rend de plus en plus complexe la tâche.

Exemples d'applications pour les réseaux de véhicules

Ayant défini les différents types d'applications dans la section précédente, nous citons quelques exemples d'applications liées au confort et à la sécurité routière. Les contraintes de ces applications sont différentes comme par exemple la vitesse de propagation de l'information. Dans le cas d'un accident, il faut prévenir les usagers dans un temps borné alors que la diffusion de publicités n'a pas cette contrainte de temps mais elle sera par contre plus consommatrice de bande passante.

Pour une alerte en cas d'accidents, les véhicules se dirigeant vers le lieu de l'accident seront averties des conditions de circulation. Pour un cas de ralentissement anomal, les automobilistes seront avertis des situations de circulation particulières et de la nécessité de ralentir. Le message d'alerte est émis par un véhicule détectant les difficultés de circulation (un freinage important, déclenchement de feux de détresse, pluie...) ou par un véhicule banalisé effectuant des travaux par exemple. La transmission de l'information aux autres voitures doit se faire d'une façon efficace et rapide.

D'autre part, nous pouvons imaginer des services d'infotainment comme l'accès à Internet ou le téléchargement de musique depuis les stations de transport (gare, métro...), les stations d'essence, ou même en plein autoroute (en passant d'une voiture à une autre jusqu'au point d'accès le plus proche). Un service de gestion d'espaces libres dans les parking est aussi envisageable afin de guider les automobilistes.

Propriétés et problématiques des réseaux de véhicules

Les réseaux véhiculaires sont une application dédiée et spécifique des réseaux ad hoc mobiles. Cependant, les travaux de recherche étudiés et réalisés dans le domaine des MANETs ne peuvent pas être directement appliqués dans le contexte des réseaux de véhicules à cause de la forte mobilité de ces derniers. Le déplacement des voitures sur des routes qui se croisent engendre un changement continu de voisinage. Même au sein d'un convoi, à la mobilité relative réduite, le déplacement des véhicules renouvelle leur environnement, conduisant à des caractéristiques des liens de communication instables.

Les réseaux de véhicules se distinguent par les propriétés suivantes :

– leur capacité de traitement et leur énergie sont avantageuses en comparaison avec d'autres types de réseaux ad hoc comme les réseaux de capteurs par exemple. Nous pouvons de même embarquer plusieurs interfaces de communication à la fois.

- leur mobilité et leur déplacement sont liés aux infrastructures routières. Leurs trajectoires peuvent être prédites. Cependant, les conditions environnementales affectent leur modèle de mobilité et la qualité de leurs transmissions radio.
- les applications fournies par ces réseaux et la nature des informations qui circulent forment aussi une particularité des réseaux véhiculaires car l'une des application clé est la prévention et la sécurité routière. L'information diffusée est alors destinée à un ensemble de véhicules limités dans une zone géographique donnée.
- le changement de topologie et de connectivité dans ces réseaux est du à leur forte mobilité. Cette mobilité est liée à la vitesse des véhicules et à leur changement de trajectoire. Un nœud peut rejoindre un réseau et le quitter en un temps très court, ce qui rend les changements topologiques très fréquents. Cette mobilité partitionne et divise le réseau.
- la sécurité et l'anonymat dans un réseau est aussi un problème de sécurité de communication très important. Un message provenant d'un nœud malveillant doit être ignoré par les autres nœuds. D'autre part, le respect de la vie privée doit être assuré par un mécanisme rendant les messages anonymes.

Il ne faut pas oublier que les challenges socio-économiques restent un facteur important qui retarde le développement et le déploiement de ces réseaux. En fait, il faut obliger d'abord les automobilistes à équiper leur voiture par le matériel correspondant pour les services de communications V2I et V2V, ce qui n'est pas réalisable économiquement. De plus, il faut pouvoir offrir le service correspondant ainsi que des avantages afin de motiver les conducteurs à investir et devenir premiers clients.

Les réseaux véhiculaires représentent un domaine de déploiement important au niveau des développements ITS. Leurs spécifications les rendent plus intéressants pour les études des protocoles et des performances, surtout en terme de routage et d'échange d'information. Pour ces raisons, plusieurs laboratoires de recherche se sont intéressés à ce type de réseaux, et se sont équipés afin d'approfondir leurs études dans ce domaine. Le laboratoire HEUDIASYC (HEUristique et DIAgnostique de SYstèmes Complexes), dans lequel s'est déroulée la thèse, en fait partie.

1.2 Le laboratoire HEUDIASYC

Le laboratoire Heudiasyc est une unité mixte de recherche entre l'Université de Technologie de Compiègne et le CNRS (rattachement à l'IN2SI). Son activité est basée sur la synergie entre recherche amont et recherche finalisée, afin de répondre aux besoins dans les domaines de la sécurité, la mobilité et le transport, l'environnement, et la santé. Ceci se fait en collaboration avec des partenaires métiers, en particulier industriels.

Plusieurs plateformes et démonstrateurs ont été développés au sein du laboratoire, afin de mettre en œuvre les travaux des équipes. Ceci montre la volonté de confronter la recherche fondamentale à la complexité des applications.

L'activité scientifique du laboratoire est organisée autour de quatre équipes :

- ASER : Automatique, Systèmes Embarqués, Robotique ; elle se consacre à l'étude et au développement de méthodes de commande, de supervision et d'observation

de systèmes dynamiques complexes en interaction avec un opérateur humain. Les applications s'effectuent autour des drones et des véhicules intelligents.
- DI : Décision, Image ; cette équipe travaille autour de la décision et de l'image (raisonnement incertain, fusion d'information, vision par ordinateur, etc).
- ICI : Information, Connaissance, Interaction ; les travaux permettent une interaction cognitive entre les systèmes informatiques et leurs utilisateurs.
- RO : Réseau, Optimisation ; cette équipe s'intéresse à l'optimisation dans les systèmes logistiques et informatiques et à la sécurité et mobilité dans les réseaux.

Nous remarquons que les différentes équipes du laboratoire adoptent des thèmes de recherche transversaux, ce qui facilite la montée de projets communs entre les différentes disciplines.

Des projets pour la communication collaborative ont été mis en place (projet Percoive, projet Sedvac). Ils impliquent les communications inter-véhicules pour la collecte et la retransmission de données capteurs. Ces données sont retransmises vers un serveur de l'infrastructure.

Un projet sur la fusion distribuée de données est aussi entamé entre les équipes ASER, RO et DI, pour la fusion distribuée des données capteurs sur les véhicules, ainsi que l'étude de la confiance de ces données avant la fusion.

L'orientation de la recherche dans le laboratoire encourage à aborder les problèmes applicatifs des systèmes intelligents pour le transport d'une façon inter-disciplinaire et appliquée. D'où les expériences sur route pour les preuves de concepts, et la diversité dans les études d'un problème.

Bien que les travaux de cette thèse se situent autour des thématiques de l'équipe RO, leurs applications et déploiement se font en parallèle à des projets inter-disciplinaires du laboratoire comme le Labex [7] ou Equipex [5].

De nombreuses plateformes ont été développées au sein du laboratoire afin de mettre en œuvre les travaux et les recherches effectués par les différentes équipes. Parmi ces plateformes, PACPUS pour la perception et l'assistance à une conduite plus sûre, et DYNET/AIRPLUG pour les communications inter-véhicules. Ces deux plateformes sont dédiées aux expériences avec les véhicules intelligents.

Les travaux de cette thèse ont été implémentés et validés à l'aide de la plateforme Airplug que nous décrivons dans la suite.

1.3 La plateforme expérimentale : *Airplug*

Bien que d'autres implémentations sont envisageables pour les travaux de cette thèse, la réalisation a été basée sur la plateforme *Airplug*. *Airplug* est un intergiciel de communication léger pour les réseaux dynamiques [32]. Il se caractérise par sa robustesse et sa simplicité dans l'organisation des échanges de messages intra- et inter-véhicules, ce qui le rend adapté aux réseaux dynamiques.

1.3.1 Principe d'*Airplug*

L'architecture de *Airplug* repose sur les facilités apportées par les systèmes d'exploitation standards : allocations de ressources, ordonnancement des processus, gestion du "temps réel", etc. Airplug étant développé en "espace utilisateur", aucune modification n'est faite au niveau de l'OS, ce qui accroît sa portabilité. Le programme-cœur *airplug* (abrégé APG sur la figure 3.1) s'intercale entre les interfaces réseaux et les applications.

Toutes les communications via *Airplug* se font par passage de messages. Un message provenant d'une application peut être envoyé à plusieurs autres applications, locales ou distantes. Cependant, une application ne peut pas recevoir les messages adressés à toutes les autres applications locales, ni émis par une application non locale sans s'être abonnée auprès d'*Airplug* au flux de données provenant de cette application émettrice. Ce principe d'abonnement permet à une application de contrôler ses réceptions. Les messages utilisent un format d'adressage spécifique.

La réalisation des communications utilisées par *Airplug* se fait de la manière la plus simple et robuste possible, et ceci en utilisant les entrées et sorties standard. Cela garantit une indépendance complète du langage de programmation utilisé pour développer les applications. Pour chaque processus lancé par *Airplug*, l'entrée et la sortie standard sont redirigées depuis et vers *Airplug* via un *pipe*. Ainsi, chaque fois qu'un processus écrit sur sa sortie standard, *Airplug* reçoit les données via ce lien, et chaque fois que *Airplug* écrit sur un de ses liens, le processus correspondant pourra lire les données sur son entrée standard. Comme les interfaces réseaux sont aussi gérées par *Airplug*, les applications accèdent au réseau de la même manière qu'elles le feraient pour communiquer avec d'autres applications locales, simplement en écrivant sur leur sortie standard. Dans ce cas, *Airplug* reçoit les données envoyées par les processus et les envoie vers l'interface désirée.

L'architecture a été conçue de manière à garder une ouverture pour les solutions futures. De ce fait, il y a peu de règles imposées à la création d'une application, qui elle, peut être développée dans n'importe quel langage de programmation. Les applications n'ont pas besoin d'inclure des opérations sur le réseau, ni de gérer les communications inter-processus. Ces applications restent indépendantes de la plateforme. La plateforme est basée sur un seul programme cœur, nommé *airplug*. Ce programme est installé sur chaque nœud mobile et tourne comme processus standard au dessus du système d'exploitation. L'architecture *Airplug* gère les applications locales et distribuées. Une application locale n'a pas d'interaction directe avec les applications distantes. Une application distribuée est composée de plusieurs instances du même programme, tournant sur différents nœuds et échangeant des messages.

La plateforme et le cœur *airplug* sont compatibles avec tout langage de programmation. Cependant, le langage Tcl/Tk a été choisi pour les applications *Airplug* par les développeurs de la plateforme. Ce choix vient du fait que Tcl/Tk est un langage simple et admet des facilités au niveau de la gestion des interfaces graphiques. De plus, Tcl/Tk offre une compatibilité avec NS (*Network Simulator*), ce qui a permis le développement d'un outil Airplug-Ns gérant la simulation pour une application *Airplug* avec NS. Les travaux de cette thèse sont développés en Tcl/Tk en conformité avec la plateforme de

FIGURE 1.1 – Photos des tests sur route effectués pour la validation des applications.

FIGURE 1.2 – La plateforme formée de Mini-dell, GPS et antennes WiFi, ainsi qu'une carte 3G.

l'équipe.

1.3.2 Composants

La plateforme *Airplug* (présentée dans la figure 1.2) est formée de plusieurs composants permettant le travail et les tests avec les mêmes programmes et les mêmes codes sur route, sur table, sur ordinateur, sur NS et à distance :

- sur route : airplug-road, pour effectuer des tests sur route (figure 1.1) ;
- dans le laboratoire : airplug-lab, utilisant la même plateforme matérielle que sur route ;
- sur un ordinateur : airplug-emu, permettant de rejouer les tests sur routes avec les traces GPS (détaillé dans la sous-section 1.3.3) ;
- à distance, en mode *remote* : airplug-rmt, pour pouvoir lancer des applications sur différentes machines ;
- avec Network Simulator : airplug-ns.

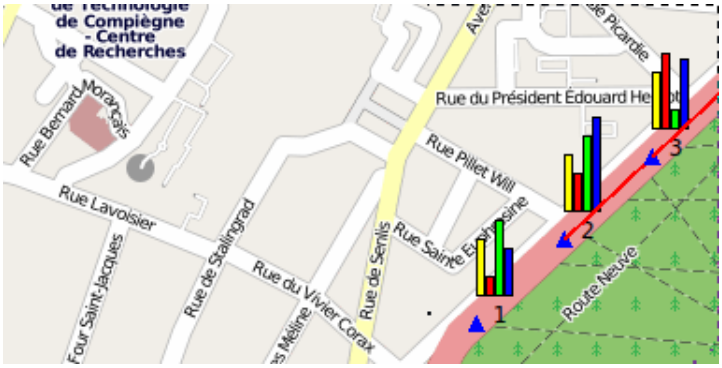


FIGURE 1.3 – Une capture d'écran de l'émulateur de la plateforme *Airplug*, les nœuds sont présentés par des triangles et positionnés sur la carte grâce à leur position géographique.

Nous détaillons maintenant l'émulateur de la plateforme qui a servi à plusieurs validations et expériences.

1.3.3 Airplug EMU : l'émulateur

Les simulateurs gèrent les scénarios routiers complexes générés par des générateurs de trafic donnés. Cependant, la simulation nécessite quelques spécifications sur le modèle de propagation, ou sur les protocoles étudiés. Ceux-ci sont généralement très différents des versions utilisées réellement sur route [23]. De ce fait, les résultats obtenus par simulation sont souvent différents des vraies mesures. Les simulations sont cependant utilisées pour la mise à l'échelle et les comparaisons de performances entre plusieurs protocoles.

Les tests sur route donnent des mesures de performances précises pour des situations réelles. Ils permettent aussi de faire la preuve de concept des applications et protocoles. Cependant, ces tests ne sont pas nombreux et impliquent généralement peu de véhicules. En effet, ces tests sont difficiles à mettre en place, vu qu'ils nécessitent l'implication de plusieurs personnes, plusieurs véhicules et du matériel. De plus, ces expériences sont non répétables, vu que le scénario d'exécution dépend des autres véhicules présents sur les routes. D'autre part, les distances inter-véhicules sont difficiles à contrôler, et les performances des liens sans fils dépendent des conditions environnementales.

Vu que la dynamique du réseau influe les performances des simulations et des tests expérimentaux, nous montrons dans [23] que l'émulation est un moyen adapté à l'étude de tels réseaux. Durant les émulations, quelques parties sont réelles, d'autres sont reproduites artificiellement. *Airplug-EMU* permet un prototypage rapide et des mesures de performances précises. Cet émulateur (dont une capture d'écran est illustrée dans la figure 1.3) est basé sur la plateforme *Airplug*.

EMU émule des réseaux fixes ou dynamiques, et utilise les facilités shell pour gérer les communications. Le scénario à utiliser pour l'émulation est un fichier *xml* qui décrit le nombre de nœuds, leur déplacement, les applications à lancer sur chaque nœud... Cet

émulateur permet de mieux comprendre l'exécution des applications en permettant de changer les couleurs, tailles, étiquettes et formes des nœuds.

1.3.4 Matériel utilisé pour les tests sur route

Pour les tests de validation, des PCs (Dell mini-9 Modèle DP118) (figure 1.2), sous Ubuntu (v8.04 Hardy Heron), ont été utilisés. Ces ordinateurs ont été équipés d'un GPS (nécessaire à l'évaluation des conditions fournies à HOP) et d'une carte WiFi externe à connectique USB (Alfa AWUS036EH) permettant de connecter une antenne sur le toit des véhicules (D-link). Le périphérique WiFi inclus d'origine dans les PCs aurait pu être utilisé pour la découverte des points d'accès WiFi. Par contre, une autre carte WiFi externe a été utilisée afin de permettre une plus grande couverture. Certains PCs sont équipés d'une carte 3G (HUAWEI E510). Un serveur web Apache du laboratoire a été conçu pour recevoir les requêtes. Une page web spécifique en PHP a été ajoutée pour stocker les données dans un fichier.

1.4 La problématique de communication dans les réseaux dynamiques

Le développement des applications de sécurité routière dans le cadre des application ITS privilégie les travaux de recherche sur les réseaux véhiculaires. Ces réseaux n'ont pas les caractéristiques des réseaux traditionnels, et leurs particularités les rend plus complexes à l'étude.

Comme pour un réseau ad hoc, la nature répartie de ces réseaux exige à chaque nœud de se comporter à la fois comme terminal et comme routeur. Les nœuds doivent alors pouvoir gérer la retransmission des messages quand cela est nécessaire (à travers des communications V2V). Par contre, pour les réseaux véhiculaires, cette retransmission se fait vers d'autres nœuds du réseau ainsi que vers des bornes de l'infrastructure, d'où le besoin de communications V2I. Les communications V2V sont utilisées pour les échanges de messages d'alerte ou de données entre les différents nœuds du réseau. Les communications V2I sont utilisées principalement pour la remontée des données vers des serveurs en vue d'une centralisation pour des analyses ou de stockage pour une diffusion plus large.

L'utilisation des réseaux ad hoc sans fil pour déployer des applications avec des communications V2V et V2I exige une étude des performances de ces réseaux. Si les exigences de l'application ne sont pas cohérentes avec les performances que peut offrir le réseau, nous pouvons alors dire et prévoir que l'application ne fonctionnera pas d'une façon efficace. Nous pouvons d'ailleurs avoir une idée sur les performances de ses réseaux d'après nos connaissances sur la nature de leurs liens sans fil qui sont sujettes à des pertes importantes dues aux collisions à la réception et au manque de fiabilité. Ces études sont détaillées dans le chapitre 2.

D'après cette étude de performances et des différentes expériences ayant montré la fragilité des liens et les pertes sur ces liens, nous déduisons que nous pouvons avoir

des meilleurs résultats et de meilleurs réceptions si nous adoptons des communications V2V et V2I opportunistes. Le fait de ne pas chercher à recevoir un acquittement d'un message envoyé ou à avoir une communication en mode connecté, permet d'économiser un délai dû aux échanges de messages et aux retransmissions suite aux pertes sur les liens. Ces réflexions nous ont conduit à proposer une architecture opportuniste pour l'accès à l'infrastructure depuis un réseau véhiculaire. Cette architecture est décrite dans le chapitre 3.

Nous montrons dans ce chapitre que les communications V2I opportunistes que nous proposons se montrent assez efficaces pour la remontée des données puisqu'elles permettent d'éviter les délais de configuration d'adresses, les délais du *hand over*. En outre, l'utilisation des communications V2V pour rejoindre une borne fixe de l'infrastructure étend la portée de ces bornes et permet un déploiement moins dense. Il est avantageux d'acheminer le message par communications V2V jusqu'à la rencontre d'une passerelle vers l'infrastructure.

Après avoir étudié les communications V2I, nous nous sommes intéressés aux communications V2V.

Les communications V2V dans les réseaux mobiles sont divisées en 3 catégories : les communications *one to all*, les communications *one to many* et les communications *one to one*. Pour les communications *one to all* ou le *broadcast*, la cible est tous les nœuds du réseau. Les communications *one to many* correspondent au *multicast*, la cible est un sous-ensemble des nœuds du réseau. Pour les communications *one to one* ou *unicast*, la cible est un nœud bien défini du réseau.

Une communication *unicast* est utilisée pour échanger des messages entre deux entités qui se connaissent. Une entité source possédant des données à transmettre à une destination bien définie entame une communication *unicast* en envoyant ces données dans des messages adressés à la destination.

Cette communication peut être directe, autrement dit la destination est à la portée de la source, et les messages sont alors envoyés directement à cette destination, ou bien indirecte et réalisée en multi-sauts. La communication *unicast* vers une destination éloignée implique généralement la recherche d'un chemin de la source vers la destination en passant par d'autres nœuds du réseau. Une route est alors construite entre ces deux entités. Plusieurs protocoles de routage existent pour la construction de chemin pour les réseaux traditionnels.

Par contre, la route construite est stable tant que le réseau est stable. Tout changement dans la topologie peut entrainer une reconfiguration des données de routage (route, table...). Ces mises à jour sont aussi fréquentes que les changements de topologie. Nous en déduisons que les protocoles qui se basent sur une connaissance de la topologie deviennent obsolètes quand les réseaux sont fortement dynamiques, car ceci implique une certaine stabilité de la topologie. Comme nous le verrons, cela est vrai encore si la source a besoin de connaitre la position de la destination dans le réseau.

Les protocoles de routage pour les réseaux dynamiques, bien qu'ils soient conçus pour ce type de réseaux, n'affrontent pas tous les problèmes de la mobilité surtout quand la dynamique du réseau est très grande. En fait ces protocoles de routage reposent tout

d'abord sur une recherche de la destination (dans le cas des routages réactifs), ou sur le maintien de structures de données (dans le cas des routages proactifs), ou même sur des services de localisation (dans le cas des routages géographiques). Ils reposent donc plus ou moins sur un *broadcast*. La requête pour la recherche d'une adresse ou d'une route vers une destination se fait par *broadcast*, ce qui peut écrouler le réseau. D'autre part, le maintien de structures implique des échanges périodiques entre toutes les entités du réseau, ce qui peut de même écrouler le réseau. Enfin, le service de localisation, par ses requêtes et ses réponses de localisation, engendre beaucoup de messages de contrôle, surtout quand la dynamique est forte. Nous en concluons qu'il faut essayer d'éviter le *broadcast* pour économiser les ressources. Il est alors intéressant de proposer un *unicast* sans adresse, qui sera mieux adapté aux réseaux dynamiques.

L'information concernant cette destination (adresse ou route) est vite obsolète. Au moment où la source reçoit ces informations, il est très probable que la destination ait changé de position (et cela est dû à la mobilité du réseau). Plus la dynamique est forte, et moins ces informations sont valides. Une communication *unicast* distante est sujet à plusieurs ruptures de connexion et de recherche de nouvelles routes. Il semble alors plus intéressant de chercher à maintenir les communications déjà établies entre les voisins dans un réseau, que de chercher à établir une nouvelle communication.

Nous proposons dans ce manuscrit un algorithme de maintien de chemin dédié à maintenir les communications *unicast* entamées entre deux voisins dans le réseau. Cet algorithme n'utilise pas de *broadcast*. La réparation du chemin se fait par intervention des nœuds voisins pour relayer la communication. Toutes les mises à jour du chemin se font localement. L'algorithme est décrit dans le chapitre 4.

L'étude de cet algorithme nous a mené à une modélisation de la dynamique d'un réseau, et à une validation formelle de l'algorithme. La validation et la modélisation sont détaillées dans le chapitre 5.

Cet algorithme a été validé expérimentalement par des preuves de concept sur route. Ses performances ont été à la suite validées par des études par émulations (grâce à l'émulateur de la plateforme *Airplug* décrit dans la section précédente). Nous présenterons au fil de ce manuscrit les différentes étapes d'implémentation, de validations formelle et expérimentale, ainsi que les résultats. Le passage au protocole, la preuve de concept sur route et les études par émulation sont décrits dans le chapitre 6.

1.5 Conclusion

Le développement des ITS motive de nombreuses études au niveau des communications V2I et V2V dans les réseaux véhiculaires. Ayant parcouru les différentes aspects des communications *unicast* dans ce chapitre, nous nous intéressons aux mécanismes de communications ne reposant pas sur des adresses, qui sont problématiques dans les réseaux dynamiques. Nous avons détaillé dans ce chapitre les motivations des travaux décrits dans ce manuscrit. Dans la suite, nous développons les études de performances effectuées, l'architecture opportuniste V2I proposée, et l'algorithme de maintien de chemin, ainsi que le passage vers le protocole implémenté.

Chapitre 2

Étude des difficultés des réseaux véhiculaires

Les réseaux véhiculaires sont des réseaux dont les performances restent encore incertaines et inconnues. Les liens ont des fiabilités variables suivant les conditions environnementales et les entourages des zones de déploiement. D'autre part, la dynamique des voitures accentue les incertitudes et affecte les performances.

Comme expliqué dans le chapitre précédent, l'évaluation des capacités d'un réseau mobile en général et véhiculaire en particulier, est importante afin de comprendre les performances des algorithmes déployés dans ces réseaux. Dans ce chapitre, nous présentons les études réalisées pour les évaluations de performances d'un réseau dynamique sans fil. Les expérimentations ont été possibles grâce à la plateforme *Airplug* décrite dans le chapitre précédent.

Les travaux de ce chapitre ont été publiés dans un article intitulé *On the capacity of a linear vehicular network* à VTC 2011.

2.1 L'étude des performances d'un réseau véhiculaire

L'expansion des nouvelles applications ITS nécessite de revoir les protocoles déjà existants pour les réseaux, qui ont changé de nature avec les nouvelles technologies (sans fil, mobile...), et l'analyse et l'étude des performances de ces réseaux pour garantir le minimum des propriétés requises par les algorithmes déployés.

La plupart des applications innovantes dans ce domaine sont des applications qui nécessitent des communications véhicule-à-véhicule. De ce fait, l'étude du réseau VANET s'avère un outil nécessaire pour les études de performances, puis l'implémentation et le déploiement des applications. Il s'agit en fait d'étudier les capacités d'un réseau sans fil mobile en terme de débit, délai et pertes dans des conditions environnementales réelles (convoi sur autoroute, interaction avec un flot de véhicules...). Les applications à implémenter seront à la suite ajustées selon leur nécessité en débit ou délai, et des capacités du réseau qui les accueillera. La connaissance des capacités du milieu où l'application à implémenter sera déployée, permet de garantir des résultats lors de l'exécution semblables ou identiques à ceux attendus et souhaités.

Comme nous l'avons introduit dans le chapitre précédent, les applications de sécurité routière, comme les applications d'alerte par exemple, sont plus concernées par les délais de transfert d'un paquet que par le débit que peut assurer une connexion. Elles sont concernées par le délai de première réception d'un message. D'autre part, les applications de transfert de flux sont plus concernées par les pertes sur les liens. Elles nécessitent alors un réseau où les pertes sont relativement faibles. Les applications diffusant des données relatives aux véhicules sont sensibles aux deux métriques (perte et délai).

Plusieurs études de capacité d'un réseau sans fil ont été évaluée par des simulations (exemple [46]). Mais les simulations manquent de réalisme. En fait, de telles études diffèrent par rapport aux vrais tests effectués sur route, et cela au niveau des représentations des couches basses surtout. Les conditions de trafic et de l'environnement ne sont pas représentées d'une façon complète.

Les simulations ne sont pas l'outil idéal pour les évaluations de performances d'un réseau qui admet de grandes incohérences entre les valeurs estimées et les valeurs réelles. Les réseaux sans fil sont très vulnérables aux environnements dans lesquels ils sont déployés. Un changement de voisinage, une variation de la densité peuvent les perturber. Autrement dit, un réseau VANET admet des performances différentes suivant qu'il se trouve déployé dans une zone urbaine ou dans une zone rurale où il y a moins de multi-chemins. De même, lorsque ce réseau se trouve dans un environnement dense (en nombre de voitures ou en bornes sur les routes), les pertes sur les liens vont être accentuées.

2.1.1 Travaux antécédents

Des tests sur route ont été mentionnés dans des travaux précédents [40, 38, 24, 56, 41, 62, 50]. Dans [40], les communications entre deux véhicules ont été étudiées. Cet article montre que le protocole IEEE 802.11 offre des performances suffisantes pour des applications de type P2P ou FTP. Les performances de UDP et de TCP ont été évaluées dans [38] à travers le transfert de données entre une voiture et un point d'accès 802.11b.

Les performances d'un réseau ont été examinées dans [24] en utilisant des voitures liées à des points d'accès WiFi dans des conditions normales de conduite. En revanche, le cache a été utilisé pour optimiser les délais de configuration et d'acquisition des adresses IP. Les auteurs montrent que ce type de réseau est adéquat pour une large variété d'applications, en particulier celles qui tolèrent des connectivités intermittentes.

D'autres travaux traitent des performances des réseaux véhiculaires. Dans [56], les performances d'un réseau en UDP et TCP ont été mesurées avec des véhicules se déplaçant à des vitesses différentes qui dépassent des points d'accès IEEE 802.11 sur les bords de route. Les auteurs discutent de l'intérêt d'avoir un WLAN dispersé à travers des équipements placés dans des entités mobiles. Une analyse des performances attendues est ensuite exposée.

Dans [41], des expérimentations avec plusieurs véhicules roulant sur une autoroute et communiquant avec des points d'accès sont développées. Les auteurs montrent que les protocoles courants et existants assurent 50% du débit total maximal. Les conditions du voisinage des points d'accès sont connues et peuvent être exploitées dans le but d'améliorer les accès opportunistes des réseaux véhiculaires.

D'autre part, les performances des équipements IEEE 802.11a, 802.11b et 802.11g ont été comparées pour des communications véhicule-à-véhicule et véhicule-à-infrastructure [62]. Les résultats montrent que les facteurs majeurs dont la variation affecte les performances sont la distance inter-véhiculaire, la distance à vol d'oiseau entre l'émetteur et le récepteur, et l'algorithme choisi pour gérer l'adaptation de débit.

D'autres expérimentations sur le terrain *(Field Operational Test)*, comme dans des projets tels que Pre-Drive C2X et SIM-TD préparent de grands essais pour évaluer les technologies des communications véhiculaires [8, 12].

Pour toutes les raisons citées ci-dessus, des tests sur route ont été utilisés pour évaluer les capacités d'un réseau véhiculaire mobile et sans fil. Des évaluations de performances et des analyses ont été effectuées à la suite pour mieux comprendre le comportement d'un tel réseau en terme de débit, délai et perte.

2.1.2 Contribution

Le travail décrit dans ce chapitre étudie les capacités des réseaux de véhicules pour un scénario en convoi. Les performances du réseau sont alors observées, étudiées et analysées. Des améliorations sont ensuite proposées pour garantir de meilleurs performances.

Ces études reposent sur des tests sur route effectués dans des conditions réelles de conduite et de circulation. Sept voitures ont été utilisées. La technologie IEEE 802.11 standard a été utilisée pour les communications sans fil. Toutes les communications entre les différentes entités ont été effectuées par des *broadcast* (locaux) à 2Mb/s au niveau de la couche physique. Le *broadcast* local est assez efficace pour les échanges dans les réseaux dynamiques vu qu'il évite la découverte du voisinage, les configurations et les échanges d'adresses quand le voisinage est instable et changeant. Les études faites et les résultats obtenus peuvent être appliqués à d'autres protocoles et standards (exemple : 802.11a ou 802.11p...).

En parcourant tous les travaux dans la sous-section précédente, nous pouvons noter que les technologies comme le 802.11a, b ou g sont encore utilisées d'une façon assez répandue, même si de nouveaux protocoles, comme le 802.11p, sont en cours de développement. Les équipements sur le marché sont compatibles 802.11a, b ou g. Ceux compatibles 802.11p restent relativement chers et difficilement accessibles. Pour ces raisons, les technologies a, b et g de 802.11 sont utilisées actuellement. Au contraire des communications *unicast*, le *broadcast* permet d'éviter les délais nécessaires à la configuration d'adresses durant les communications, en évitant la construction des tables de routage ou l'identification entre les entités. Cet avantage est d'autant plus important quand le protocole utilisé pour le routage ne repose pas sur des adresses IP. Dans le cas de ces tests, les transmissions conditionnelles sont utilisées.

Toutes les mesures (pertes, délais et débits) ont été faites dans la couche applicative, et ceci pour mesurer le temps total de traitement depuis l'envoi jusqu'au traitement (en comptant le temps occupé par le traitement au niveau du système d'exploitation). Pour cette raison, une application simple a été mise en place pour générer des messages périodiques et les envoyer de proche en proche et sur plusieurs sauts. Pour les contraintes des tests avec la plateforme *Airplug*, les communications sont en UPD sur IP fixe. Les mesures permettent de former une idée sur les performances globales de la communication, de bout-en-bout. Les spécifications de l'application peuvent être de cette façon comparées à ses performances réelles une fois déployée dans un environnement donné.

Dans ce chapitre, les résultats des tests sur route sont mis en évidence. Les pertes dans les réseaux mobiles sans fil se montrent importantes. Par contre, les pertes de données peuvent être évitées par des émissions multiples. Ces émissions peuvent consommer de la bande passante. Or, pour des applications d'alerte et de sécurité routière, et au contraire d'applications de transfert de fichiers par exemple, l'envoi d'un ou de quelques messages n'affecte pas la bande passante.

Les répétitions engendrent des délais, mais le délai qui compte est celui de la première réception, surtout quand le message est transmis à plusieurs sauts. Cependant, il reste possible de relayer des informations de saut en saut pour les applications temps réel qui se basent sur des données produites par les capteurs intégrés.

Cette étude s'intéresse surtout aux analyses des pertes, délais et débits. Le délai inter-paquet choisi pour l'envoi des messages est de 100 ms, et cela est dû au fait que la valeur de 100 ms a été prouvée la plus efficace en terme de pertes (elle engendre le moins de pertes par rapport à des valeurs inférieures ou supérieures). Cette valeur a été mesurée par des expériences sur table dans le laboratoire.

2.2 La plateforme expérimentale, les scénarios et les mesures des métriques

Les analyses des performances ont été faites durant la preuve de concept (figure 2.1) de plusieurs applications *Airplug*. Le matériel utilisé est décrit dans le chapitre précédent.

FIGURE 2.1 – Photos des expérimentations sur route.

2.2.1 Logiciel

Les tests reposent sur la plateforme *Airplug*, détaillée dans le chapitre correspondant (chapitre 1). Pour ces tests, l'intergiciel Airplug a été utilisé avec son outil de transmissions conditionnelles (application HOP) [30], et un programme de test qui génère des paquets et mesure les performances à la réception des messages (application dite TST).

Afin de mesurer les performances du réseau véhiculaire, une application spécifique (TST) compatible *Airplug* a été mise en place. Le premier véhicule du convoi a été désigné pour être la source des messages. L'application TST locale de ce véhicule génère des paquets d'une taille donnée, avec un délai inter-paquet prédéfini. Les instances TST sur les autres véhicules reçoivent les messages et calculent des métriques d'après les données reçues. Ces métriques sont détaillées dans la suite. De ce fait, l'application TST génère des messages comprenant toutes les informations utiles. L'architecture complète utilisée pour les tests est montrée dans la figure 2.2.

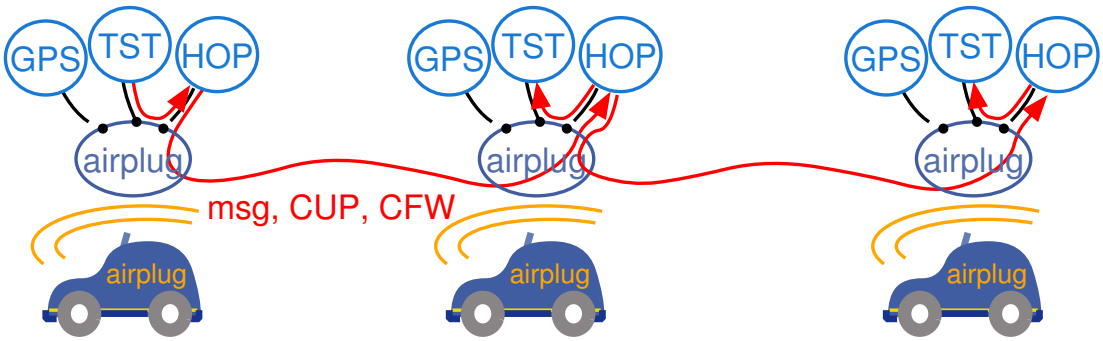


FIGURE 2.2 – L'architecture proposée pour les tests.

2.2.2 Scénario

L'architecture utilisée pour ces tests est composée de sept véhicules. Les mesures ont été faites pour des messages transmis du premier véhicule, source de la communication, au dernier du convoi, destinataire de la communication. La distance moyenne entre les véhicules varie de 325 m à 400 m. La vitesse moyenne était de 76km/h, pour une conduite

sur une route double voie dans un environnement normal de circulation.

Les portées des antennes étaient un facteur incontrôlable durant les tests. Pour cette raison, les conditions de retransmissions ont été forcées de sorte que le message suive un ordre bien défini dans le convoi. L'application HOP, à travers les conditions envoyées avec le message, permet d'avoir une topologie en chaine linéaire, et par la suite, d'avoir des résultats en nombre de sauts (cas de la figure 2.4). Sans cette fonctionnalité, des messages émis par la voiture i peuvent atteindre la voiture $i + 2$ ou $i + 3$ sans avoir traversé les voitures intermédiaires. Dans ce cas, les résultats dépendent des distances entre les véhicules et non du nombre de sauts (cas de la figure 2.5). Seules les mesures relatives au nombre de sauts ont été analysées. Le choix d'analyser et d'étudier les résultats dépendants du nombre de sauts vient du fait que la mesure de performances suivant la distance reste sujette au matériel utilisé et à l'environnement plus que les mesures suivant le nombre de sauts.

Parmi les différents tests sur route effectués, six ont été retenus pour analyse. Le tableau 2.3 les résume. Le sigle IPG désigne le délai inter paquet (*Inter Packet Gap*). Les tests sont choisis avec des délais inter paquets variables (4 tests avec un IPG de 100 ms, 1 test avec un IPG de 250 ms et 1 test avec un IPG de 500 ms), de façon à pouvoir déduire si ce délai inter paquets influence les résultats. Les analyses ont montré qu'au delà d'une valeur de 100 ms, le délai inter-paquets n'influence pas les résultats. Cependant, des analyses sur table au laboratoire on montré que pour des valeurs inférieures à 100 ms, les pertes sont grandes sur les liens.

Test	1	2	3	4	5	6
IPG (ms)	100	100	100	100	250	500
# paquets	994	611	255	294	170	137
Durée (s)	113	63	25	37	43	68

FIGURE 2.3 – Table de tests, avec les délais inter-paquets et le nombre de paquets et les durées correspondant

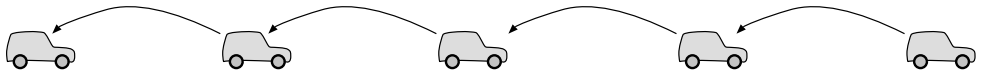


FIGURE 2.4 – Le scénario choisi avec le forçage des sauts. Le message traverse tous les véhicules entre la source et le puits.

2.2.3 Métriques mesurées

Parmi les métriques les plus signifiantes, nous mesurons les pertes, les délais et les débits. Pour mieux comprendre les capacités d'un réseau véhiculaire, et pour pouvoir étudier sa capacité à satisfaire les besoins des applications ITS à déployer, il a fallut

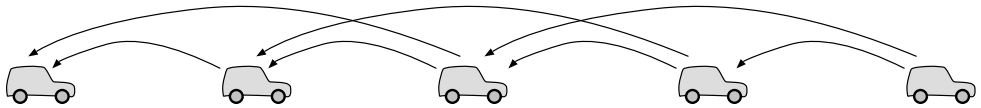


FIGURE 2.5 – Le scénario au cas où les sauts ne sont pas forcés. Dans ce cas un message peut être retransmis par deux relais différents.

ajouter une métrique importante, qui est les évènements de pertes. Nous détaillons les différentes métriques :

- Le taux de pertes est calculé par le pourcentage de paquets qu'un véhicule perd par rapport au nombre de paquets qui lui sont adressés.
- Les délais sont mesurés depuis le départ du paquet de la voiture source à son arrivée sur la voiture destination.
- Le débit est calculé par le nombre de paquets qu'un nœud peut envoyer par seconde.
- Les évènements de pertes fournissent des connaissances concernant l'environnement dans lequel se déroulent les expériences. Ils sont calculés en repérant les pertes de messages successifs.

Les résultats obtenus sont détaillés dans la section suivante.

2.3 Les pertes dans un convoi de véhicules

Cette section décrit les résultats des pertes dans un convoi de véhicules. Comme prévu, les pertes dans un réseau sans fil sont importantes. Ces résultats affectent les performances des applications déployées dans ce type de réseau. Cependant, les résultats obtenus par les tests sur route diffèrent largement de ceux obtenus par une simulation ou étude théorique. Pour compenser ces pertes, un dispositif de retransmission est proposé, permettant de garantir un taux de réception supérieur à celui obtenu dans un réseau sans fil. Ces différentes analyses sont expliquées à la suite.

2.3.1 Résultats

Les pertes dans un réseau donné sont étudiées à travers le pourcentage de réception sur chaque entité. La figure 2.6 montre la variation du pourcentage de réception en fonction du nombre de sauts. Une ligne brisée présente la valeur moyenne des pourcentages de bonnes réceptions de l'ensemble des tests (se référer au tableau 2.3 pour plus d'information sur les tests). Comme on peut l'observer, les résultats varient suivant les expérimentations. Comme prévu, le pourcentage de bonne réception décroit avec le nombre de sauts. Une diminution de 10% est remarquée au niveau du premier saut.

La variation du délai inter-paquet n'influence pas les résultats. Les variations des conditions environnementales sont d'autant plus importantes, et entrainent des variations brusques des performances. Les résultats obtenus sont très différents des résultats

obtenus par simulation dans [46]. Lors de ces simulations, les délais inter-paquets imposés ont montré un impact important sur les résultats. Le simulateur utilisé (ns-2) ne représente pas parfaitement les conditions environnementales.

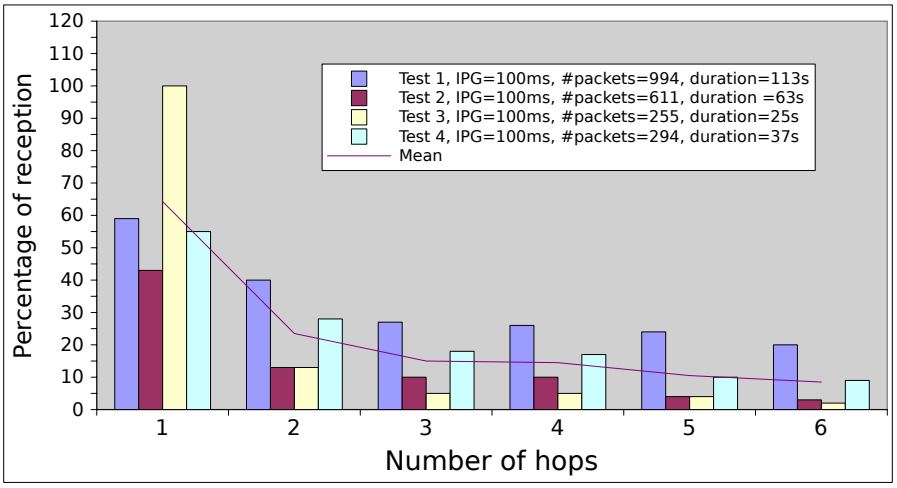


FIGURE 2.6 – Pourcentage de réception sur chaque véhicule du convoi, pour 4 expériences avec des délais inter-paquets de 100 ms

2.3.2 Analyse des résultats

La remarque au niveau des pertes est que dans 5 sauts, le pourcentage de bonne réception tombe à 10%. Il est important de noter qu'il n'y a pas de retransmission durant les tests pour compenser les pertes. L'envoi des messages se fait sans attente d'acquittement de la part du récepteur. Ces informations permettent d'avoir une idée sur les valeurs réelles des pertes sur une liaison WiFi dans un réseau mobile en *broadcast*. Ces résultats accentuent en plus le besoin et l'importance des retransmissions qui se font soit en unicast WiFi (couche 2) soit en TCP (couche 4). Ces retransmissions peuvent être gérées par le protocole de communication utilisé, ou par l'application elle-même. À noter que ces retransmissions perdent de leur importance et de leur nécessité quand les données échangées sont des données produites régulièrement par des capteurs. Une bonne réception d'un message émis peut arriver quand les données portées par ce message ne sont plus valides.

Cependant, la répétition des messages (dont l'absence influence les résultats de pertes) augmente le taux de bonne réception. Un message émis réussit alors à atteindre plus de voitures avec un meilleur pourcentage de réception. Dans la figure 2.7, les influences des transmissions multiples sont étudiées.

2.3.3 Répétitions des messages

L'envoi d'un message à plusieurs reprises sert à augmenter le taux de bonne réception de ce message. Dans la figure 2.7, les pourcentages de bonnes réceptions sont montrés en fonction du nombre de retransmissions et du nombre de sauts traversés par le message. Pour rappel, les tests sont faits avec un délai inter-paquets de 100 ms. Pour illustrer ces résultats, considérons que nous disposons d'un message à envoyer, et que ce message doit avoir une réception garantie à 80%. Il faut alors l'envoyer 2 à 3 fois afin qu'il atteigne un véhicule à distance 1 (un saut), 4 à 5 fois afin qu'il atteigne un véhicule à distance 2 (2 sauts), et 6 fois au moins afin qu'il atteigne un véhicule à distance 3 (3 sauts). Pour un parcours supérieur à 3 sauts, le paquet peut être émis 5 ou 6 fois, mais le pourcentage de bonne réception sera de l'ordre de 70%.

Ces résultats peuvent sembler inacceptables en un premier abord. Cependant, résultants de la nature opportuniste du réseau, ils arrivent à satisfaire les besoins de certaines des applications à déployer dans les réseaux véhiculaires, en particulier celles qui reposent sur des envois de données générées périodiquement par des capteurs.

Dans un premier temps, les fréquences de génération de messages permettant la traversée de quelques voitures sont étudiées, et cela pour des pourcentages de réceptions donnés. Considérons par exemple une application générant des messages toutes les secondes (période de 1 s), et une émission avec un délai inter-paquets de 100 ms ; l'application considérée transmet alors des messages générés par un seul capteur. Dans ce cas, un même message peut être envoyé 10 fois afin de parcourir le plus de véhicules possibles. La figure 2.7 permet d'illustrer les résultats d'une telle expérience.

Cependant, les messages émis ont une taille bornée par le LLC (*Logical Link Control*) alors que les messages générés par les capteurs sont d'une taille réduite. Plusieurs messages provenant de plusieurs capteurs peuvent être concaténés pour former un message de taille correspondante à celle des messages échangés. Concaténer plusieurs données et les envoyer à plusieurs reprises sur le réseau, permet de garantir une traversée plus étendue des messages et des taux de réceptions plus importants. La figure 2.8 permet d'illustrer le contenu d'un message potentiel. Ce message est alors une concaténation de plusieurs données capteurs. La dernière information provenant d'un capteur sera prise en compte dans le message. Dans cette figure, nous illustrons le contenu d'un message envoyé en fonction du temps et de la sortie des capteurs A, B et C. Prenons l'exemple du message émis avec un délai inter-paquet de 500 ms ; il contient les informations a2 de A, b2 de B et c1 de C. Ces informations représentent les dernières sorties des capteurs respectifs.

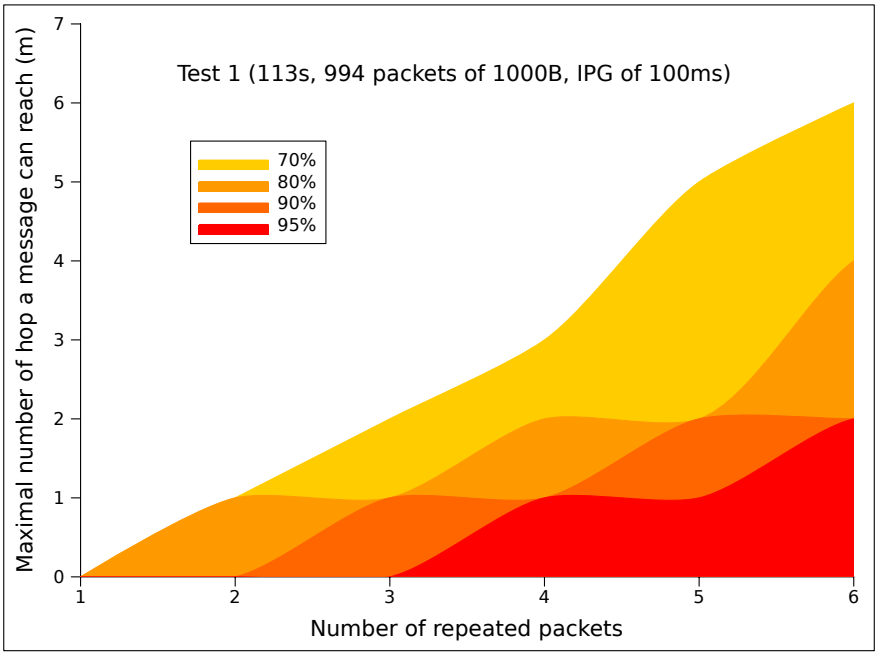


FIGURE 2.7 – Nombre maximum de sauts qu'un message peut traverser, en fonction du nombre de retransmissions effectuées.

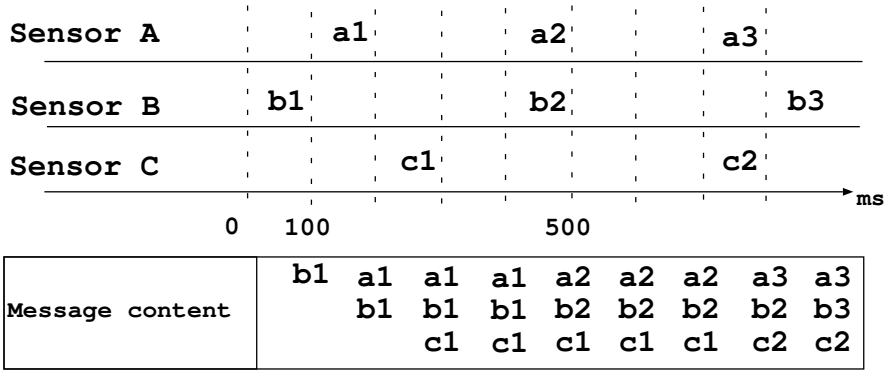


FIGURE 2.8 – Exemple d'un message concaténant plusieurs données provenant d'applications différentes. Le contenu du message à chaque instant dépend des dernières sorties des capteurs. Le message est alors une concaténation des dernières informations collectées des capteurs respectifs.

2.3.4 Evènements de perte

Les études précédentes montrent que les pertes dans un convoi de véhicules sont très importantes. Par contre, il reste possible d'avoir une bonne réception d'un message si ce dernier est envoyé à plusieurs reprises. Mais cette technique échoue si plusieurs paquets successifs sont perdus. À ce point, il est important d'étudier les événements de pertes.

Les informations sur les évènements de pertes permettent d'avoir une idée sur la distribution des pertes des paquets et le nombre de paquets successifs perdus. L'ensemble de paquets successifs perdus est alors noté comme évènement de pertes. Il est remarqué après une séquence de paquets bien reçus.

Il est remarquable que les évènements de pertes soient distribués d'une façon uniforme. D'une façon générale, un à deux paquets sont perdus à chaque fois. La perte d'une série de messages à la fois est rare (cf. figure 2.9).

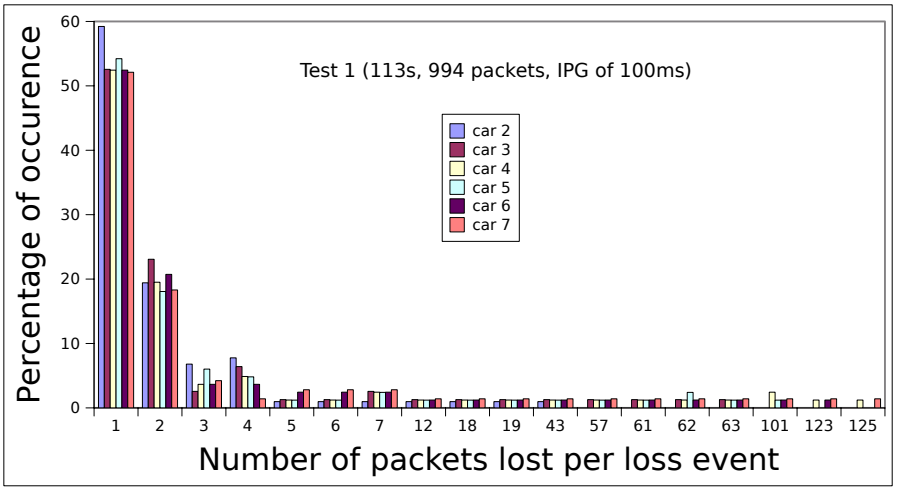


FIGURE 2.9 – Occurrence des évènements de perte en fonction du nombre de paquets perdus durant cet évènement

Les évènements de pertes les plus importants engendrent la perte de 1 à 5 messages successivement. Le pourcentage d'occurrence de ces évènements est en revanche important. La moyenne cumulée de ces pourcentages d'occurrences montrent que dans 80% des évènements de pertes, 1 à 5 messages consécutifs sont perdus. Comme nous pouvons le constater, la plupart des pertes concernent 1 à 2 paquets en général. Nous remarquons que 80% de ces évènements de pertes concernent moins de 5 paquets. Cependant, la distribution des pertes en fonction du temps est aussi un facteur important. La figure 2.10 illustre cette distribution en fonction du temps pour le Test 1 (cf. figure 2.3). L'axe des ordonnées indique le numéro du dernier véhicule détenant (ayant reçu) le paquet en question. Par exemple, pour le paquet numéro 250, il a été reçu en dernier sur le véhicule 1, et ceci car il n'a jamais été reçu sur le véhicule 2.

Une autre remarque peut être faite sur les résultats des évènements de pertes. Les

pertes les plus importantes sont groupées sur des périodes de 5 s, et comprennent 50 paquets environ. Ces pertes résultent surtout des conditions environnementales dans lesquelles les tests ont été faits.

Pour résumer, les évènements de pertes comprennent généralement un à deux paquets, et ils sont groupés sur des périodes de 5 s sur certaines voitures. Ceci dit, les facteurs externes affectent les performances locales des communications. Ces facteurs externes sont en général les distances inter-véhicules, la présence de poids lourds sur les voies, les traversées des rondpoints, etc.

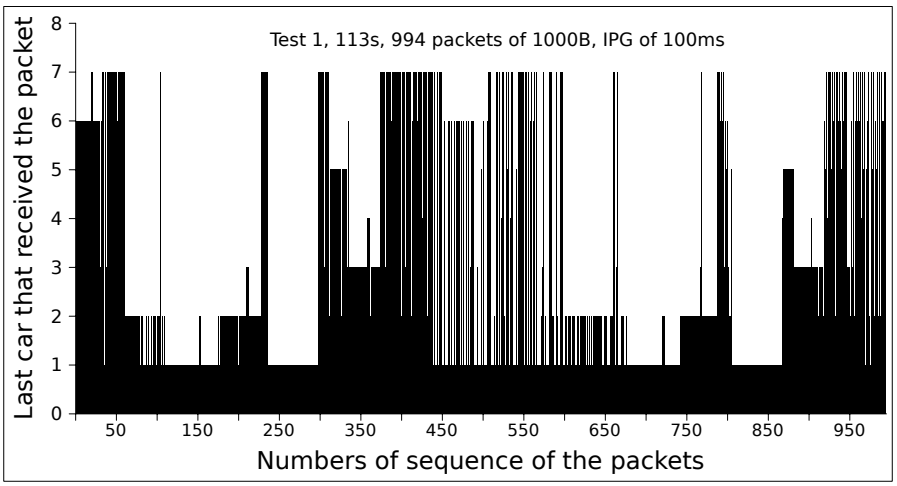


FIGURE 2.10 – Graphe représentant l'identité du dernier véhicule ayant reçu un paquet donné en fonction du numéro de séquence du paquet. Le véhicule 1 est la source des messages.

2.4 Les délais dans un convoi de véhicules

Le délai est le temps mesuré entre l'envoi d'un paquet et sa réception sur un véhicule donné. Ce délai est un facteur important pour comprendre les performances des applications sur un tel réseau, surtout celles des applications d'alerte et d'aide à la conduite. Une application nécessitant un délai de réception minimal, déployée dans un réseau où le délai assuré est supérieur aux exigences de cette application, peut échouer.

Dans cette section, les délais de réception du paquet d'origine et de ses retransmissions sont étudiés.

2.4.1 Résultats

Comme pour les pertes, les conditions environnementales et de trafic dans lesquelles les tests ont été faits ont une grande influence sur les résultats, et par conséquence sur les délais.

Comme le montre la figure 2.11, le délai augmente avec le nombre de sauts parcourus par le message. Les résultats varient de test en test, mais une tendance générale est remarquée (illustrée dans la figure 2.11). Il faut noter que, comme pour les pertes, le délai inter-paquets n'affecte pas les délais.

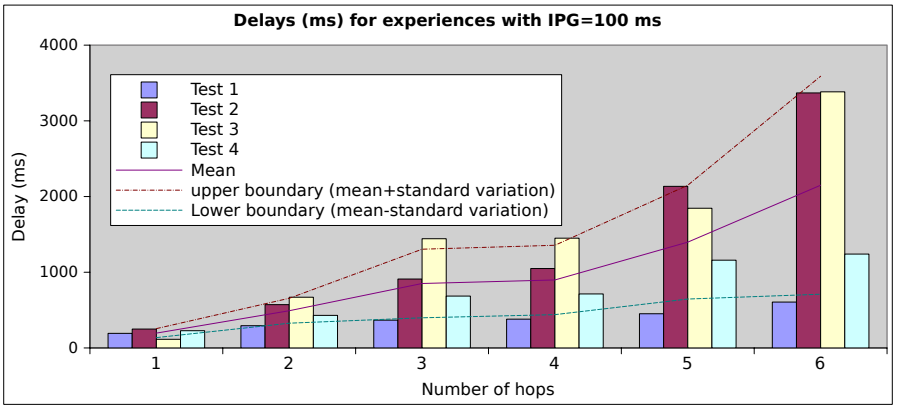


FIGURE 2.11 – Résultats des délais pour quatre expériences avec un délai inter-paquets de 100 ms

2.4.2 Analyse des résultats

Pour les résultats des expériences en terme de délais, seuls ont été traités les messages arrivés sur les voitures, donc les résultats sont largement affectés par les pertes sur les liens.

Comme déjà expliqué, les pertes sur les liens affectent les communications dans le convoi, mais l'envoi multiple des paquets assure un bon taux de réception.

Cependant, ces retransmissions engendrent naturellement des délais entre l'émission du premier paquet et l'arrivée de l'information quelques sauts plus tard, sachant que l'arrivée de l'information nécessite parfois plusieurs envois.

On peut remarquer par exemple, que pour le Test 1 (figure 2.12), le délai peut dépasser le délai critique de sécurité (la distance de sécurité n'est pas respectée) quand le paquet doit être retransmis à plusieurs reprises. Dans le cas de cet exemple, le délai est dépassé quand le message est transmis cinq fois pour atteindre la troisième voiture. La distance de sécurité qu'on mentionne décrit le temps de réaction nécessaire pour qu'un conducteur puisse réagir à une situation critique sur route. Ce temps est de 2 s en France [1].

En d'autres termes, si un taux de réception de 90% est exigé, le paquet doit être envoyé 5 fois, engendrant un délai total supérieur à celui nécessaire pour réagir.

Les résultats sont très dépendants des tests. Par contre, les trois tests effectués indiquent un délai inférieur au délai de sécurité. Ce délai est respecté jusqu'à la sixième

1. Article R412-12, modifié par Décret numéro 2003-293 du 31 mars 2003-art.2 JORF 1er avril 2003.

voiture. Au delà de la sixième voiture, le délai de sécurité peut être dépassé.

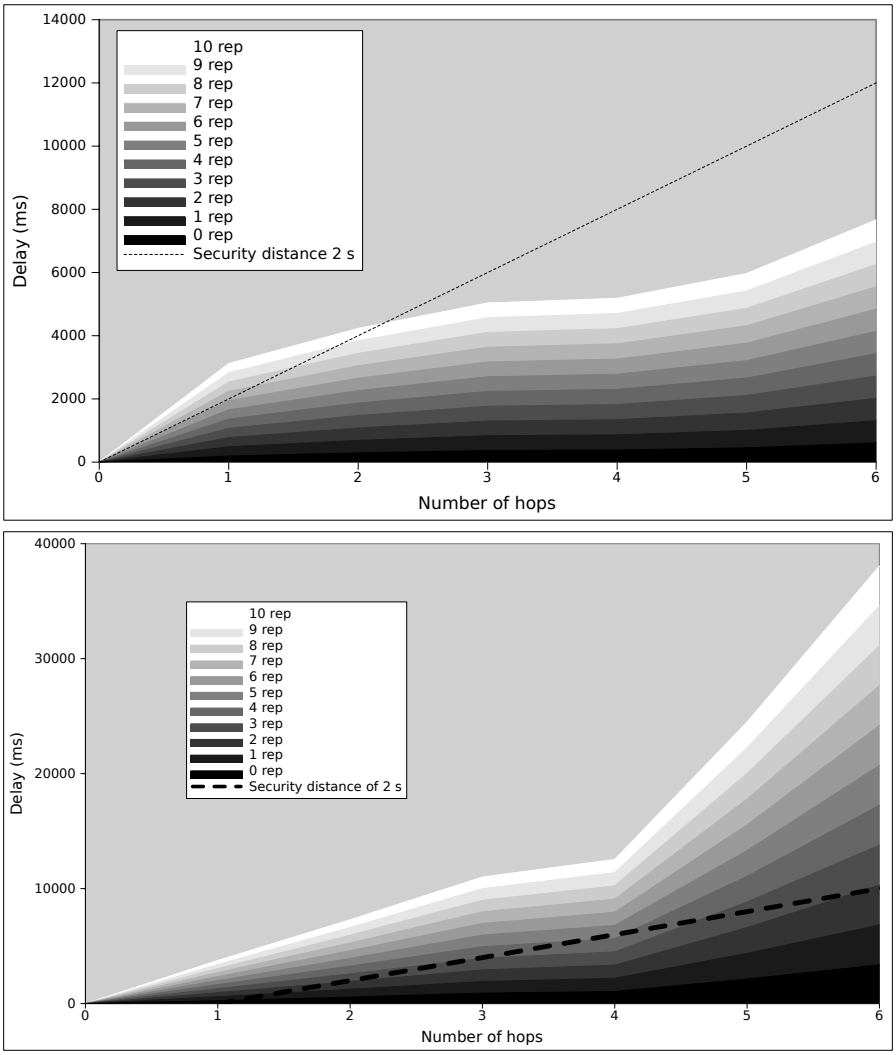


FIGURE 2.12 – Les délais engendrés quand les répétitions sont nécessaires, en fonction du nombre de sauts à parcourir pour traverser le trajet de deux des tests effectués.

2.5 Les débits dans le convoi

2.5.1 Résultats

Les résultats des tests sur routes, se focalisant sur le débit, sont montrés dans la figure 2.13. La variance par rapport à la valeur moyenne est importante pour le premier saut (17 Kb/s), puis varie de 5 à 7 kb/s pour les autres sauts. Cela représente 38% de

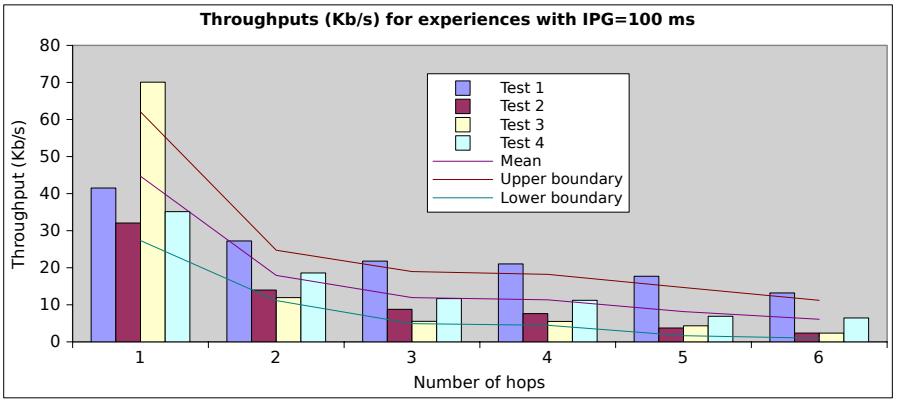


FIGURE 2.13 – Les débits pour quatre expériences avec un délai inter-paquets de 100 ms.

la valeur moyenne du premier saut, et atteint 83% du dernier saut.

2.5.2 Étude des résultats

Il est important de noter que pendant l'envoi d'un message de 1000 bytes chaque 100 ms, le débit théorique est de 80 kb/s. D'où, dans la figure 2.13, le débit moyen est proche de 50% de la valeur théorique maximale pour le premier saut, et proche de 25% dès le second saut. Ces résultats sont acceptables pour les applications de sécurité routière qui ont peu de données à transmettre, mais qui sont en revanche très sensibles au délai et au taux de bonne réception. Ces résultats se montrent acceptables surtout avec les pertes importantes observées sur les liens.

2.6 Conclusion

L'analyse des performances d'un réseau véhiculaire constitue un point clé pour comprendre ses capacités et les performances d'un algorithme dans ce réseau.

Ce chapitre a décrit les résultats expérimentaux des tests sur route effectués avec un convoi véhiculaire et une technologie IEEE 802.11 utilisée en *broadcast*. Plusieurs métriques du réseau ont été étudiées, en particulier les pertes, les évènements de pertes, les délais et les débits.

D'après les résultats obtenus, le réseau véhiculaire en convoi dans les vraies conditions, satisfait largement le besoin des applications de sécurité routière et d'aide à la conduite. Les pertes, assez importantes, peuvent être compensées par des retransmissions des messages. Ces répétitions engendrent un délai supplémentaire à celui engendré par la communication et le transfert par multi-sauts. Mais ce délai total engendré reste tolérable et inférieur à la limite seuil si les répétitions sont limitées. Un taux de réception acceptable est garanti avec ces retransmissions. En outre, l'étude des évènements de pertes indique que cette technique est envisageable.

La concaténation des messages de différentes applications permet d'économiser en terme de bande passante.

Ces résultats peuvent être exploités pour extrapoler un nouveau protocole offrant un meilleur débit pour les communications. Un bon taux de réception est assuré, ainsi qu'un délai tolérable en concaténant les messages, ce qui offre aussi un débit acceptable. Ce protocole est d'autant plus efficace s'il n'est pas influencé par les évènements de pertes. Des évènements de pertes engendrant la perte de x paquets successifs peuvent affecter le bon fonctionnement d'un protocole donné. Une optimisation des protocoles de transfert de flux pour les réseaux véhiculaires peut être considérée suite à ces tests.

Les résultats ont permis de comprendre les performances d'un réseau véhiculaire en convoi. Le fonctionnement des protocoles implémentés dans ces réseaux dépend alors de leur adaptation.

Ces études peuvent servir de base pour améliorer les performances des algorithmes dans les réseaux véhiculaires. Ils nous ont déjà aidé à comprendre la nature de ces réseaux, et de prévoir les pertes de messages sur les liens WiFi. Ces résultats représentent un début d'une étude plus complète et élaborée.

Chapitre 3

Architecture vers l'infrastructure

L'accès vers l'infrastructure à partir d'un réseau véhiculaire devient une nécessité incontournable pour les applications ITS. Nous détaillons dans ce chapitre l'architecture opportuniste et légère pour l'accès vers l'infrastructure.

À noter que ce travail est le fruit d'un stage de Master 2 effectué au laboratoire Heudiasyc, et qui a permis d'avoir des discussions très intéressantes avec Thierry Ernst sur le sujet de l'utilisation de IP dans les communications V2I et V2V.

Les travaux détaillés dans ce chapitre ont été publiés dans deux articles ; le premier, intitulé *A light architecture for opportunistic vehicle-to-infrastrcuture communications* a été publié dans MobiWac 2010. Le deuxième, intitulé *Architecture pour communication véhicule-infrastructure*, a été publié dans CFIP 2009.

3.1 Problématique de l'accès à l'infrastructure depuis un réseau véhiculaire

3.1.1 Travaux précédents

Un certain nombre d'applications pour les systèmes de transport intelligents nécessitent des communications véhicules-infrastructure. On peut citer la remontée des informations sur les vitesses ou la recherche d'un point d'intérêt entre autres applications ayant recours à une base de donnée centralisée dans l'infrastructure. Pour cette raison, il y a une forte tendance aujourd'hui à lier les réseaux de véhicules à l'infrastructure en général et à Internet en particulier. Cette tendance permet d'offrir une gamme de services élargie pour le passager. Ces services sont soit offerts par les constructeurs automobiles, soit par les exploitants des installations infrastructures. Cependant, la configuration d'adresses reste un processus coûteux en terme de messages de contrôle ainsi qu'en terme de temps consacré à cette configuration.

De larges initiatives ont été faites dans le domaine des communications V2I (un bilan de quelques uns de ces projets peut être trouvé dans [1]), dans le but de mettre en valeur la sécurité routière en particulier, et la gestion des infrastructures routières en général (réduction des embouteillages, optimisation des utilisations des routes ...).

Motivés par la sûreté routière et la gestion des infrastructures (réduction des embouteillages, remontée et centralisation d'informations...), de larges initiatives R&D ont été lancées aux USA (VII [13], CICAS, IVBSS...), en Europe (CVIS, SafeSPOT [10], COOPERS, PReVENT, GST, HIGHWAY, FleetNet, SeVeCom [11], GeoNet...), au Japon (SmartWay, VICS), en Inde (ITSIndia), en Allemagne (NoW), en France (PREDIT)... Ainsi l'initiative VII (*Vehicle Infrastructure Integration*) se base sur des communications V2V et V2I pour accroître la sûreté et limiter les embouteillages. Le projet CVIS (*Cooperative Vehicle Infrastructure Systems*) porte également sur la sûreté routière ; il inclut des communications V2V et V2I [4]. Le projet Safespot vise à développer un *Safety Margin Assistant* basé entre autre sur les communications V2V et V2I. Le projet PReVENT vise à aider le conducteur à éviter les accidents ou à limiter leur impact ; le sous-projet WILLWARN fait appel aux communications V2V et V2I. Le projet GST (*Global System for Telematics*) porte sur la création d'un standard ouvert pour les services télématiques à bord [6].

Du point de vue des protocoles réseaux, la mise au point de standards adéquats préoccupe les divers organismes internationaux (IEEE, IETF, ETSI, ISO, SAE, ASTM) ou les consortia d'industriels, comme le C2C-CC par exemple. L'IEEE développe la pile protocolaire WAVE, incluant une extension de la famille de protocoles 802.11 pour les applications ITS. L'ISO développe le standard Calm pour les réseaux de véhicules. L'IETF travaille sur des extensions d'IP (Mobile IP, IPv6, Nemo) et sur l'autoconfiguration dans les réseaux Manet (*Mobile Ad hoc NETwork*) et Vanet (*Vehicular Ad hoc NETwork*) au sein du groupe de travail Autoconf. Le Car-to-Car consortium (C2C-CC) développe et expérimente des protocoles spécifiques aux réseaux de véhicules. Ces initiatives sont en cours de développement et donnent lieu à de nombreux travaux ; leur intégration ou inter-opérabilité fait l'objet d'intenses discussions et recherches.

Plusieurs solutions ont été envisagées pour les connexions des réseaux de véhicules à Internet. Les acteurs principaux dans ce domaine sont l'IEEE (*Institute of Electronic and Electronics Engineers*), l'ISO (*International Organization for Standardization*), l'IETF (*Internet Engineering Task Force*) et le C2C-CC (*Car 2 Car consortium*).

L'IEEE a tout d'abord étendu la famille de protocoles d'accès 802.11 en y ajoutant le protocole 802.11p. Ce protocole s'inspire du standard ASM E2213-03 [1] qui est en lui même basé sur 802.11a. Ce protocole modifie la couche physique et la couche MAC pour s'adapter aux communications dans les réseaux véhiculaires, correspondant à la bande DSRC (*Dedicated Short Range Communication*). Le terme DSRC désignait des concepts différents depuis la gamme de fréquences jusqu'aux types d'applications. En complément, l'IEEE a défini la famille de protocoles 1609, dite *WAVE Wireless Access in Vehicular Environments*. Ce standard structuré en quatre composants (1609.0 à 1609.4) définit l'accès WiFi dans les réseaux véhiculaires [15]. Il définit aussi l'architecture, le modèle de communication, la structure de management, la sûreté et l'accès physique. Au final, 802.11p et WAVE spécifient une pile protocolaire complète. Le standard 1609.3 inclut le protocole WSMP (*Wave Short Messages Protocols*) pour les communications inter-véhicules. Ce protocole est proposé comme une alternative à IPv6 [9]. À noter que le terme WAVE proposé par l'IEEE devrait clarifier les usages du terme DSRC en les limitant [15]. Actuellement, la bande DSRC ne désigne pas les mêmes gammes de fréquences d'un continent à l'autre.

Les développements de l'IEEE se sont fait en lien avec l'ISO, plus particulièrement le " *Technical Comittee* 204 *Intelligent Transport Systems*, *Working Group* 16, *Wide Area Communications*" en charge des communications moyennes et longues portées, qui travaille sur le standard Calm [2] *Continuous Air-Interface for Long and Medium range telecommunication* [3]. Il s'agit en fait plus d'un référentiel que d'un protocole, dont le but est de standardiser les communications par commutation de paquets dans les réseaux hétérogènes en environnement mobile. Le but de Calm est d'offrir des communications en continu de manière transparente à l'utilisateur à travers des réseaux et des interfaces de communication variées, telles que 802.11, 802.11p, 802.15, 802.16e, 802.20, réseaux cellulaires 2G/3G/4G et systèmes ITS nationaux. Calm intègre à la fois les travaux de l'IEEE et de l'IETF.

Nous présentons dans ce manuscrit une architecture opportuniste qui permet la communication véhicule-infrastructure. Elle présente un VANET sans IP, avec une passerelle vers l'infrastructure. Cette architecture permet d'établir une connexion IPv4 ou IPv6 depuis un véhicule vers l'infrastructure, via la 3G ou les points d'accès WiFi. Elle permet également d'utiliser les protocoles de routage spécifiques aux réseaux de véhicules. L'étude a été faite avec les transmissions conditionnelles comme outil de routage, afin d'éviter les affectations d'adresses [30]. L'architecture admet une découverte automatique des services afin de déterminer la route vers l'infrastructure.

1. (Standard Specification for Telecommunications and Information Exchange Between Roadside and Vehicle Systems - 5 GHz Band Dedicated Short Range Communications (DSRC) Medium Access Control (MAC) and Physical Layer (PHY) Specifications)

2. Communication Architecture for Land Mobile depuis 2007

3.1.2 Contribution

Les travaux cités dans la section précédente permettent de remarquer que les projets de recherche ITS actuels encouragent la révision des protocoles existant pour les réseaux et la mise au point de nouveaux protocoles pour les communications véhicules à véhicules et véhicules à infrastructure. La nature dynamique de ces réseaux exige la considération de la mobilité des nœuds comme facteur essentiel dans la conception des nouveaux protocoles, même ceux concernant l'accès vers l'infrastructure avec les communications V2I.

L'accès à l'infrastructure peut subir les règles d'adressage IP pour l'affectation d'adresse et la transmission de requêtes, sachant que d'autres possibilités sont envisageable, mais pas encore déployées (surtout pour une communication *unicast* entre le véhicule et l'infrastructure) ; les autres solutions sont semblables au protocole WAVE pour les communications à courte portée, qui éventuellement fait usage de IP dans les communications. Les communications V2V par contre, peuvent surmonter l'adressage dans le réseau de véhicule pour utiliser à la place les communications WAVE ou les communications à transmissions conditionnelles par exemple. L'utilisation des adresses IPs dans le réseau V2V pose de multiples sujets de débats. En fait, la configuration de l'adresse, les requêtes IP et les *handovers* sont très coûteux dans des réseaux à forte mobilité, tels que les réseaux véhiculaires. Une telle uniformisation des réseaux a des avantages mais aussi des inconvénients en forme de contrôle ajouté (*overhead* du message ou messages supplémentaires), en plus des problèmes d'auto-configuration des adresses déjà cités.

Plusieurs travaux de recherche ont traité l'accès à l'infrastructure et la mobilité des réseaux. L'IETF travaille depuis quelques années sur les réseaux mobiles, les réseaux ad hoc, et plus récemment les réseaux de véhicules. La problématique adressée est celle d'un déploiement complet d'IP, en donnant à chaque véhicule une adresse. Ces travaux portent essentiellement sur IPv6.

Le protocole Mobile IPv6 repose sur la mise à jour d'une adresse temporaire dite *care_of address*. L'entité mobile dispose alors d'une adresse permanente et d'une adresse temporaire. L'adresse permanente lie le mobile à son réseau d'origine. L'adresse temporaire est liée au réseau visité. Pour chaque réseau visité, le mobile met à jour son adresse temporaire et l'envoie à une entité appelée *home agent* de son réseau d'origine afin que cette dernière l'enregistre. Tous les messages arrivant au *home agent* et destinés au nœud mobile lui seront alors redirigés. Le nœud informera ensuite la source des messages de sa nouvelle adresse. La communication peut se poursuivre alors sans l'intervention du *home agent*. En comparant à IPv4, on remarque que IPv6 est optimisé du point de vue routage, car le recours au *home agent* n'est plus nécessaire pour la redirection des paquets.

Par contre, même si le protocole IPv6 supporte la mobilité des nœuds, la mobilité des réseaux reste un problème. Ce problème vient du fait que les réseaux sont composés de plusieurs adresses IP, ce qui complexifie la gestion. De plus, certains travaux envisagent d'associer à chaque ordinateur embarqué au sein d'un même véhicule, une adresse IP. La gestion s'avère alors complexe car la destination finale des paquets de la source sera un routeur (lui même mobile), et la source n'a pas de connaissance sur l'existence des terminaux mobiles dernière ce routeur. Une communication avec ces terminaux sera alors

impossible.

Le protocole Nemo Basic Support (RFC 3963) règle ce problème tout en se basant sur le protocole Mobile IPv6. D'autre part, le protocole Nemo Extented Support étudie les problématiques d'optimisation de multidomiciliation et d'optimisation de routage, sans se baser par contre sur Mobile IPv6 [28, 34]. En fait, Nemo BS ne change pas les adresses des terminaux *Mobile Network Nodes (MNN)* situés derrière le routeur mobile (*Mobile Router ou MR*). Le déplacement du routeur mobile entraine le changement de son adresse extérieure sans par contre changer le point d'ancrage pour les MNN (ses nœuds ne changent pas d'adresses). Le *home agent* du MR encapsule alors tous les paquets dont le préfixe de l'adresse de destination correspond au préfixe du réseau mobile. Reste à résoudre les problèmes liés à la multidomiciliation d'un réseau au cas où le MR possède plusieurs interfaces (donc plusieurs adresses *care_of* sur les différents liens), alors que le HA (*home agent*) ne peut enregistrer qu'une seule adresse temporaire pour un préfixe de réseau mobile donné.

La problématique de l'assignement d'une adresse IP à un véhicule est adressée dans le *Ad Hoc Network Autoconfiguration (Autoconf) Working Group*. Les réseaux ad hoc dynamiques admettant des communications multi-sauts possèdent une nature qui empêche l'utilisation des protocoles d'auto-configuration d'adresses tels que ceux des RFC 4861 et 4862.

D'autre part, jusqu'à présent, il n'existe pas de standard pour l'assignement des adresses IP dans un réseau de véhicules [25]. Les recherches dans ce domaine restent rares. Dans [35], la topologie des réseaux VANET est supposée être formée de plusieurs petits convois linéaires indépendants. Des nœuds du convoi sont choisis pour être *leader*. Ils agissent alors comme des serveurs DHCP. C'est une solution de type *distributed DHCP* qui garantit l'unicité de l'adresse affectée au sein de chaque convoi élémentaire. Par contre, deux véhicules distants pourraient avoir la même adresse.

Une autre solution est proposée dans [17]. Elle se base sur l'architecture proposée dans le C2C-CC et sur la technique d'autoconfiguration dite SLAAC (*Stateless Address Autoconfiguration*). SLAAC repose sur une signalisation de type NDP (*Neighbour Discovery Protocol*) qui sert à vérifier l'unicité des adresses IPv6 attribuées dans le réseau. Cette technique repose sur le fait que chaque nœud du convoi est capable de communiquer avec tous les autres nœuds du réseau. Le protocole GeoSAC étend SLAAC aux réseaux géographiquement distribués en utilisant le protocole de routage du C2C-CC. Ce dernier permet d'offrir une zone de *broadcast* limitée, facilitant la tâche de configuration des adresses dans le convoi. À noter que le consortium C2C-CC promeut le développement des réseaux de véhicules. Il a développé une pile protocolaire complète spécifique à l'architecture envisagée [14]. Cependant, elle permet aussi l'intégration de IPv6, Mobile IP et Nemo à cette architecture. Les couches physiques envisagées sont 802.11p, le WiFi et la 3G.

L'architecture proposée dans ce chapitre permet la remontée de données vers l'infrastructure avec un choix de la technologie d'accès (WiFi ou 3G) et un choix d'accès IP (IPv4 ou IPv6).

3.2 Scénarios envisagés et architecture proposée

Dans cette section, nous expliquons l'architecture proposée dans ses détails et clarifirons les scénarios d'usage de cette architecture.

3.2.1 Scénario

L'architecture proposée a pour objectif de permettre aux applications embarquées dans les véhicules d'envoyer des données vers un serveur de l'infrastructure. Il s'agit donc d'une remontée des données capteurs et des données recueillies sur le bus du véhicule, vers une base de donnée centralisée de l'infrastructure. Des communications véhicules à infrastructure sont alors envisagées. Dans ce travail, les retours depuis le serveur vers le réseau véhiculaire ne seront pas traités, seules les remontées sont étudiées. Cependant, le retour des informations étant supporté par cette architecture, les extensions de ce travail ont été envisagées pendant des travaux effectués à la suite dans le laboratoire. Les détails de la communication seront expliqués ultérieurement. Les applications concernées par ce chapitre concernent la remontée des données vers un serveur. Ces applications traitent des données de type position, vitesse, adhérence, luminosité, pluie, densité de voisinage, etc. Ces données, en grande partie produites par des capteurs à bord, permettent d'évaluer les états de la route et les conditions de circulation. L'échange de ces données permet d'informer de l'état d'un véhicule, ou de gérer une flotte de véhicules (e.g. société de livraison ou de dépannage...). Les informations individuelles provenant des voitures peuvent être traitées en amont, pour une simple remontée vers la base de données, ou bien traitées en aval après réception et sauvegarde, au niveau de la base de données. Le traitement des données n'est pas abordé dans ce travail. Ce chapitre se concentre surtout sur l'architecture de remontées de données.

La remontée des données sert en fin de communication en transférant les données du réseau véhiculaire à la base de données dans l'infrastructure. À la fin de la communication, les informations sont collectées par un serveur web. La communication utilisera HTTP sur TCP sur IP (IPv4 ou IPv6 selon les disponibilités des réseaux d'infrastructure). Une page PHP dynamique d'un serveur web gère la réception des données. Afin de rester conforme aux standards web utilisés, ces protocoles seront utilisés pour les communications vers un réseau fixe. De cette façon, l'architecture ne nécessitera pas un changement au niveau de l'infrastructure (*hot spot* WiFi, réseau d'opérateur). Cependant, la connexion IP vers le serveur sera attribuée ou bien envisagée pour un seul véhicule du convoi, appelé *véhicule passerelle*. Le véhicule passerelle n'est pas forcément le véhicule ayant produit les données à transférer.

3.2.2 Architecture

Comme mentionné ci-dessus, cette architecture est conçue pour la remontée des données vers un serveur web. Une application embarquée dans le véhicule et désirant envoyer des données (illustration par l'application APP dans la figure 3.1), contactera tout d'abord sa passerelle locale pour la transmission des données. Si le véhicule dis-

pose localement d'un accès WiFi ou 3G, alors il émettra directement ses données vers Internet. Sinon, tout dépend du type des données. Si les données sont de faible urgence, autrement dit, si un petit délai avant l'arrivée au serveur est toléré, la passerelle locale attend un certain temps en espoir de trouver un point d'accès WiFi ou 3G. Par contre si les données sont urgentes (ou si le délai d'attente sur un véhicule dépasse le seuil), la passerelle locale diffusera le message à transmettre aux véhicules voisins. Si parmi ces véhicules voisins se trouve un véhicule portant une passerelle vers l'infrastructure, il se chargera de la transmission des données vers le serveur. Sinon le message sera retransmis de proche en proche jusqu'à atteindre un véhicule passerelle.

Il se peut qu'aucune passerelle ne soit disponible dans un temps ou délai raisonnable. Dans ce cas, le message ne sera plus d'actualité, et il ne sera plus retransmis sur le réseau. Le délai toléré pour un message est un paramètre réglable et considéré à l'arrivée (ou réception) de chaque message. Dans certains cas, un message peut ne jamais atteindre sa destination. Quand une passerelle existe et qu'aucun délai n'est exigé, le message émis sera reçu sur le serveur. Cependant, considérant les applications prévues dans le cadre de la sécurité routière, un message dont le délai dépasse un certain seuil peut être considéré comme un message périmé, et sa réception n'a plus grand intérêt. Il sera alors avantageux d'envoyer un message plus récent, contenant des informations produites récemment par les capteurs et les calculateurs embarqués. Dans les cas de bonne réception au niveau du serveur, il se peut que ce dernier reçoive le même message plusieurs fois. C'est le cas quand un message arrive en même temps sur deux passerelles, ou suit des chemins différents pour atteindre sa destination. Cette situation, si rencontrée, n'a d'inconvénients que la surcharge du réseau et du serveur qu'elle entraîne. La probabilité d'une réception multiple comparée avec la probabilité de non aboutissement dépend des paramètres de l'architecture. Ces paramètres sont, entre autres, le nombre de passerelles dans le convoi et le délai toléré pour un message. Ce paramétrage exige un compromis entre la surcharge des ressources et la perte de données. Cependant, ces données sont produites régulièrement par les capteurs qui mettent à jour périodiquement leurs sorties. De ce fait, il reste à définir l'importance des données à transmettre dans le message. La densité du convoi affecte également le paramétrage. Si le nombre de véhicules est suffisant, les données sont retransmises d'une façon efficace. D'autre part, le taux des bornes WiFi déployées sur les bords de routes, et le nombre de véhicules équipés d'accès Internet ou 3G joue un rôle important dans l'efficacité de la remontée. La densité de ses bornes rend leur remonté vers l'infrastructure plus rapide. À noter que ce compromis entre qualité de service et installation de ressources réseaux est toujours considéré comme une problématique, surtout économique, de déploiement des applications VANET.

Il faut noter qu'un délai supplémentaire vient s'ajouter lorsque la communication se fait en TCP/IP. Ce délai est exigé par la configuration d'adresses et des retransmissions.

L'architecture proposée comprend une communication V2V sans IP. Par contre la communication IP est conservée sur le lien véhicule-infrastructure. Ainsi, tout routage véhiculaire peut être utilisé pour le transfert de données dans le réseau véhiculaire. Cette architecture peut être utilisée avec différents protocoles de routage, comme AODV par exemple, ou DSR. Mais pour simplifier au plus l'architecture, un routage sans adresses

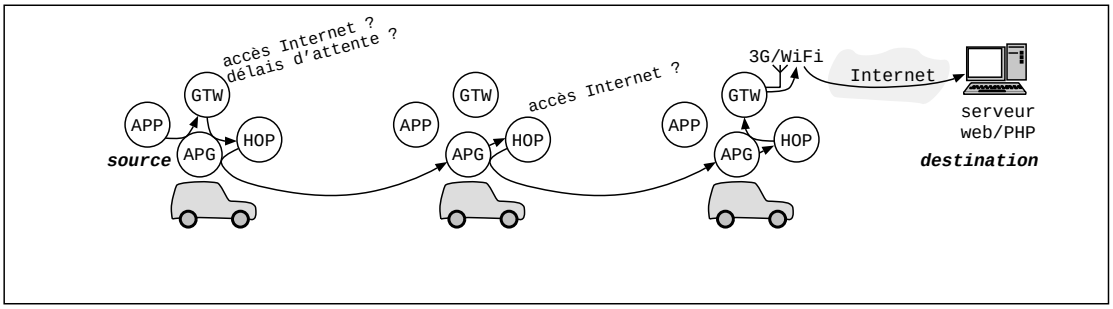


FIGURE 3.1 – Architecture de communication véhicule-infrastructure. APG : airplug (gère les communications intra et inter-véhicules), APP : application générant les données, HOP : multi-sauts VANET (transmissions conditionnelles), GTW : passerelle ayant ou non un accès vers Internet.

IP peut être choisi pour le réseau véhiculaire. Le transfert de données dans le réseau V2V engendrera alors moins de délai. Le temps de configuration du réseau sera aussi évité. Dans le cas de l'étude présentée dans ce chapitre, les *transmissions conditionnelles* [30] sont utilisées pour l'acheminement des données. Le but de ce type de routage est d'éviter l'utilisation des adresses dans les communications V2V. Elles sont remplacées par des conditions déterminant si le message reçu doit être retransmis aux couches applicatives et/ou aux voitures voisines. Avec ce type de routage, les conditions sont évaluées à la réception, évitant ainsi tout type de contrôle pour connaitre le voisinage. Ce contrôle est parfois vain et coûteux dans des réseaux dynamiques tel qu'un réseau de véhicules. Cette architecture limite alors au minimum le contrôle dans le réseau. Au cas où aucun message n'est émis sur le réseau, peu de messages de contrôle sont émis. Seuls sont nécessaires les messages servant à la découverte des *hot-spot* WiFi.

L'architecture garantit le respect de la vie privée (*privacy*) [37], ce qui représente un avantage considérable. L'aspect vie privée des communications représente un facteur important dans le développement des applications ITS. Cet aspect représente un frein pour le déploiement de certaines applications qui peuvent nuire au respect de la vie privée des automobilistes. Ce facteur devient de plus en plus important avec l'intégration des GPS et des cartes 3G ou WiFi à bord des véhicules. Avec la remontée des données, la vie privée devient menacée. Cette remontée comprend les vitesses des véhicules, leurs trajets, leurs dynamiques, etc. Dans cette architecture, les données sont transmises anonymement d'un véhicule à l'autre pour rejoindre l'infrastructure, les données transmises sur la connexion V2I n'émanent ainsi pas forcément du véhicule ayant la passerelle. Ceci protège aussi bien le véhicule émetteur que le véhicule passerelle. Il sera alors impossible de distinguer les données provenant du véhicule passerelle lui-même ou d'un autre véhicule. Par le fait que la connexion vers un serveur passe par un véhicule passerelle, ce dernier agit comme un proxy pour la connexion web : s'il n'interdit pas l'authentification, il permet aussi de l'éviter. Seul le véhicule passerelle lance la requête vers le serveur web, et nécessite alors une adresse IP pour la connexion. Mais puisqu'il est à proximité

de l'infrastructure et qu'une seule requête est envoyée à la fois, il n'y a pas à proprement parler de problématique d'auto-configuration d'adresses ni de gestion de mobilité.

Finalement, il est important de remarquer qu'une architecture IP de bout en bout, tout en posant de réelles difficultés en ce qui concerne l'auto-configuration (cf. section précédente), ne règle ni le problème du routage dans les réseaux de véhicules, ni celui du transport des données. Aucune solution normalisée n'existe pour le premier point et les difficultés de TCP en environnement mobile (ici fortement dynamique) sont bien connues. Si notre architecture n'offre pas une connexion aux standards Internet de bout en bout, elle évite pour le moins ces deux écueils et permet d'ores et déjà de remonter des données produites par les véhicules vers un serveur.

3.2.3 Composants de l'architecture

La réalisation de l'architecture, illustrée sur la figure 3.1, est décrite dans les sections suivantes. Le déploiement de l'architecture se fait à travers la plateforme Airplug (décrite dans le chapitre 1), qui est dédiée aux réseaux dynamiques tels que les réseaux de véhicules. Elle permet aussi le développement de nouveaux protocoles et de nouvelles applications d'une façon simple et robuste. Les transmissions conditionnelles [30] sont utilisées pour les communications V2V. L'application passerelle appelée GTW (pour GaTeWay) effectue l'émission vers le serveur web si un accès WiFi ou 3G est assuré, ou, dans le cas écheant, elle confie le message à HOP pour le transmettre en recherche d'une passerelle vers Internet.

3.3 Utilisation de Airplug et HOP dans l'architecture

Comme expliqué précédemment dans la section 1.3, les communications via Airplug se font par passage de messages. Un message peut être envoyé à plusieurs applications, locales ou distantes. Une application reçoit uniquement les messages des applications auxquelles elle est abonnée. Ce principe permet de contrôler les réceptions au niveau d'une application et augmente la robustesse de l'architecture.

L'application passerelle GTW sera alors une application Airplug, développée dans l'espace utilisateur. Elle recevra les données à émettre via son entrée standard et les transmettra à Airplug via sa sortie standard. Les transmissions conditionnelles, implémentées à travers l'application HOP [30] de *Airplug*, seront utilisées pour la transmission des données en multi-sauts.

La figure 3.2 détaille les relations entre l'application passerelle GTW et le protocole HOP au sein de l'architecture V2I : GTW émet localement vers l'instance locale de HOP (1), qui émet vers l'instance distante de HOP (2), qui transmet à l'instance distante GTW (3).

Pour les besoins de l'architecture, l'application dédiée aux transmissions conditionnelles a été modifiée pour accepter des messages particuliers qui lui précisent certains mots-clés à considérer comme vrais lors de l'évaluation des conditions. Ces messages proviennent des applications locales du véhicule. De ce fait, l'application passerelle lo-

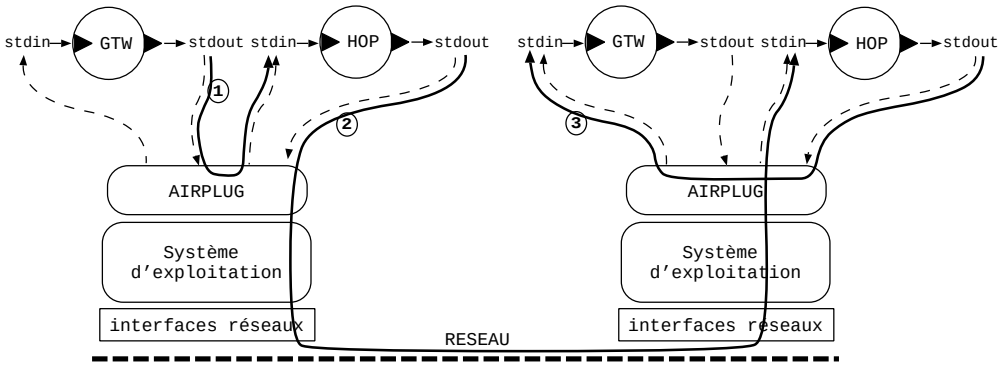


FIGURE 3.2 – Architecture de Airplug, communications intra-véhicules (1) et (3), inter-véhicules (2).

cale GTW de chaque voiture, envoie périodiquement de tels messages à HOP pour lui indiquer la présence ou non d'une passerelle *hot spot* WiFi (mot-clé wifi) ou 3G (mot-clé 3G) vers l'infrastructure.

Une application GTW qui détient un message à envoyer vers l'infrastructure, au cas où elle ne possède pas d'accès vers cette infrastructure (absence de carte 3G et de point d'accès WiFi) et que le message n'est pas urgent, attend qu'elle soit à proximité d'un *hot spot* WiFi, et transmet le message à HOP accompagné de deux conditions (CUP et CFW). L'application HOP locale envoie alors le message de GTW et les deux conditions fournies, en compléments à d'autres informations nécessaires à l'évaluation des conditions (figure 3.3), aux véhicules voisins. Les conditions peuvent avoir des combinaisons de valeurs booléennes, la condition CUP a alors la forme "wifi $\vee$ 3G" et permet de transmettre le message à toute application GTW possédant effectivement un accès vers l'infrastructure. La condition CFW prend alors la valeur "$\neg$wifi $\wedge$ $\neg$3G$\wedge$ dst 2000 $\wedge$ dur 180" et permet de retransmettre le message dans le voisinage si le véhicule ne dispose pas localement d'accès, que la distance parcourue par le message est inférieure à 2km et que le délai du message ne dépasse pas 3 minutes. Dans ces cas, les informations complémentaires sont alors la date de génération du message et la position du véhicule source au moment de l'envoi du message. Ces données peuvent être obtenues à partir des GPS embarqués, ce qui permet à tout véhicule intermédiaire de calculer sa distance à l'émetteur et l'ancienneté du message reçu. Une estampille empêche tout traitement d'un message déjà reçu.

D'autres conditions de retransmission peuvent être envisagées. Celles ci peuvent être plus restrictives, dans le sens que moins de voitures auront le droit de rée-mettre. La diffusion sera alors restreinte. Ces conditions sont en elles mêmes des paramètres à adapter suivant les conditions du réseau (densité, dynamique, ressources réseaux), le taux d'équipement des véhicules en cartes 3G ou WiF, le taux d'équipement en points d'accès vers l'infrastructure et l'importance du message à transmettre.

L'application GTW admet une découverte intuitive de passerelle et de points d'accès. La communication entre l'application GTW locale et l'application HOP locale permet

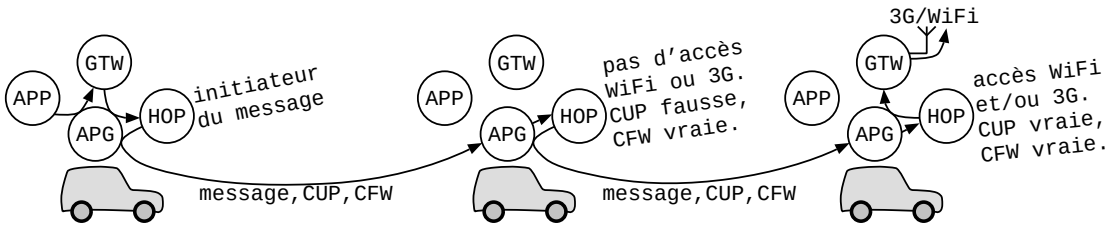


FIGURE 3.3 – Transmissions conditionnelles : un message est accompagné des conditions CUP et CFW.

d'avoir une découverte de service induite sans avoir à ajouter un pré-traitement pour rechercher une passerelle vers Internet.

3.4 GTW : la passerelle pour la découverte de réseaux, l'émission ou la retransmission

L'application dite GTW (pour GaTeWay), mentionnée précédemment, a été conçue pour les besoins de l'architecture V2I proposée. Cette application a pour rôle de faire le lien entre le réseau de véhicules et Internet. GTW vérifie périodiquement la présence d'un accès vers les réseaux externes. Les interfaces utilisées pour ces accès sont alors un sous-ensemble des interfaces détectées sur le véhicule. Le choix de l'interface peut se faire manuellement ou automatiquement. Il est alors possible de restreindre les envois à la connexion 3G par exemple. Il est également possible de choisir entre les différentes options : 3G, WiFi, ou LAN ; IPv4 ou IPv6. GTW communique à HOP périodiquement les résultats de ses diagnostics et lui indique les réseaux disponibles à travers une communication intra-véhiculaire en lui indiquant les mots-clés à évaluer à vrais lors de l'examen des conditions associées aux messages reçus, comme décrit dans la section précédente.

Lorsqu'une application désire émettre un message vers un serveur Internet, elle transmet ce message à l'application GTW locale du véhicule. Ce message est accompagné d'informations indiquant son urgence. Si ce véhicule dispose d'un accès vers Internet, l'instance locale de GTW émet directement le message vers l'infrastructure. Sinon, au cas où le message n'est pas urgent, elle se permet d'attendre qu'elle soit à proximité d'un point d'accès WiFi ou 3G pour émettre le message. Dans le cas où le message est urgent, ou bien que le délai de transfert de ce message expire, elle le transmet à l'instance HOP locale. Celle-ci se chargera de trouver une passerelle vers l'infrastructure grâce à la découverte de service induite par la technique des retransmissions conditionnelles.

La destination, qui est un serveur web de l'infrastructure, peut être définie par défaut par GTW ou bien fournie par l'application émettrice. Dans la pratique, la destination est essentiellement dépendante de l'application émettrice et du type de données transmises. Le choix de l'URL, si déterminé par GTW, économise de la place dans les messages.

Les messages émis doivent respecter une taille limitée par le MTU (*Maximum Transmission Unit*, indiquant la taille maximale d'un paquet en réseau, et exigé par la pla-

teforme dont les paquets sont de taille limitée) du chemin jusqu'au serveur. Cette taille est un paramètre à fournir à GTW. Si la taille des données excède le seuil permis, ces données seront divisées en plusieurs transmissions par GTW. Par défaut, la taille de message utilisée par GTW est de 1100 octets pour le *payload* applicatif.

Cette application GTW a été développée en Tcl/Tk, le langage choisi pour le développement des applications Airplug, comme nous l'avons expliqué dans le chapitre 1.

3.5 Validation expérimentale

Cette section décrit les expériences sur route faites pour la preuve de concept du protocole. Cette validation a permis de mettre en évidence les résultats obtenus lors des tests.

3.5.1 Matériel

Pour les tests de validation, la plateforme *Airplug* a été utilisée en mode road. La liste du matériel utilisé est détaillée dans le chapitre 1.

Nous avons rajouté, pour la centralisation des données collectées, un serveur web Apache du laboratoire qui a été conçu pour recevoir les requêtes. Une page web spécifique en PHP a été ajoutée pour stocker les données dans un fichier.

3.5.2 Scénario

Quatre voitures ont été utilisées pour les tests. Chacune était équipée d'un mini PC, d'une antenne WiFi, et d'un GPS. Une seule voiture était équipée d'une clé 3G ; elle constituait le véhicule-passerelle du convoi. Le test consistait à émettre des messages du premier véhicule du convoi et à les récupérer sur le dernier véhicule par des transmissions conditionnelles. Ce dernier, doté d'une carte 3G, envoyait les messages reçus sur le serveur.

3.5.3 Résultats

Les tests, décrits dans la section précédente, consistaient à envoyer des messages du premier véhicule du convoi au dernier pour rebondir ensuite sur le serveur du laboratoire.

Pour donner une notion des résultats obtenus lors des tests, notons qu'en moyenne, sur 25 messages émis du premier véhicule, 19 messages sont reçus sur le dernier véhicule, donc sur le serveur. Le délai moyen d'un saut inter-véhiculaire est de 31 ms environ. Il faut noter que les GPS en USB engendrent des délais supplémentaires. Nous avons constaté lors des tests que la portée des antennes WiFi utilisées peut atteindre 1km, variable suivant les conditions environnementales. En considérant la grande portée des antennes WiFi, les conditions de retransmission ont été choisies pour forcer les rebonds, afin de garantir la faisabilité du scénario souhaité. Les messages partaient donc du premier véhicule pour atteindre le deuxième, puis le troisième, jusqu'à arriver au dernier

véhicule portant la 3G. Il faut noter que l'application passerelle GTW est présente sur tous les véhicules du convoi.

Ces tests ont permis la vérification du bon fonctionnement des échanges entre les différentes applications impliquées dans la réalisation de cette architecture. Les échanges APP–GTW, GTW–HOP, GTW–Internet ont été validés. Des tests avec IPv6 ont été effectués en LAN privé et avec un serveur web spécifique faute de disposer d'adresses IPv6 publiques. Vue que l'application HOP accepte maintenant de considérer tout mot-clé comme étant vrai dans les expressions, il était facile d'ajouter le mot-clé LAN pour les tests.

L'architecture et sa stratégie d'émission ont prouvé leur efficacité d'après les tests.

D'après les expérimentations, il ressort que l'architecture est opérationnelle. Elle permet ainsi la remontée des données depuis un réseau véhiculaire vers un serveur web en utilisant le réseau WiFi ou 3G. Les échanges V2I se font en IPv4 ou en IPv6.

Un film présentant les tests et les préparations a été réalisé. Il est disponible à l'adresse suivante : `https://www.hds.utc.fr/airplug/doku.php?id=en:doc:movies:start`.

3.6 Conclusion

Ce chapitre a introduit l'architecture opportuniste pour l'accès à l'infrastructure depuis un réseau véhiculaire.

Nous avons étudié la remontée des données. L'architecture proposée permet la recherche d'une passerelle vers l'infrastructure, tout en exploitant un routage adapté aux réseaux VANET et une communication V2I (véhicule passerelle-infrastructure) avec HTTP/TCP/IP. L'avantage de cette architecture est qu'elle évite l'encombrement du réseau avec des messages de contrôle pour la configuration des adresses IP dans le réseau véhiculaire. De plus, la découverte de la passerelle, sans contrôle ajouté, est une propriété importante, de même que le respect de la vie privée facilitée par cette architecture.

Vu les résultats des études de performances des réseaux sans fil mobiles, nous avons opté pour une démarche opportuniste pour la remontée des informations vers l'infrastructure. Cette démarche évite les délais dispensables qu'exigent la configuration d'adresses et la retransmission des données. Un processus de retransmission a été étudié afin d'assurer une meilleure réception des données transmises.

Les détails de cette architecture ont été expliqués. L'application clé GTW a été détaillée, en mentionnant les apports et les ajouts nécessaires à effectuer sur la plateforme pour la rendre opérationnelle et efficace.

Les tests sur route ont été nécessaires pour la preuve de concept de l'architecture. Ces tests ont été détaillés dans ce chapitre. D'autres tests pour les mesures de performances ont été effectués. Ils seront détaillés dans le chapitre suivant. Cette application a servi à la preuve de concept d'une application de collecte de données.

Cependant, des améliorations peuvent être apportées à cette architecture. La descente des données représente un aspect intéressant à étudier. Elle viendra compléter la remontée des données pour une communication bidirectionnelle entre le véhicule et

l'infrastructure. Elle peut être étudiée d'un point de vue opportuniste comme pour la remontée, pour compléter l'architecture.

Chapitre 4

Maintien de chemin, l'algorithme

Les études précédentes sur un réseau ad hoc mobile nous ont montré les capacités et les caractéristiques d'un tel réseau.

Nous proposons dans ce chapitre un algorithme de maintien de chemin, qui permettra d'obtenir des communications *unicast* sans avoir à maintenir d'adresses. Le maintien et la recherche d'adresses sont en effet des processus coûteux dans un réseau dynamique.

Les travaux présentés dans ce chapitre ont été publiés dans plusieurs articles ; le premier, intitulé *Communications dans les réseaux dynamiques*, a été publié dans AlgoTel 2012. Un deuxième article, intitulé *A distributed algorithm for path maintaining in dynamic networks*, a été publié dans DYNAM 2011. Un troisième article est en cours de soumission à VTC 2013.

4.1 La communication *unicast* dans les réseaux mobiles

Comme nous l'avons expliqué dans le chapitre 1, les communications dans les réseaux mobiles sont de trois types :

- les communications *one to all* ou *broadcast*, où l'information détenue par un nœud est diffusée à la totalité du réseau ;
- les communications *one to many* ou *multicast*, où l'information détenue par un nœud est acheminée vers un sous-ensemble des nœuds du réseau. Ce sous-ensemble est défini par une caractéristique géographique (nœuds suivant le nœud présent par exemple) ou qualitative (nœuds ayant un seul voisin par exemple), ou autre.
  Ces communications nécessitent un routage pour pouvoir acheminer les données de la source vers leurs destinations.

4.1.1 Travaux antécédents

Les réseaux mobiles, par leurs caractéristiques, rendent inefficaces la plupart des protocoles conçus pour les réseaux traditionnels (dans [18], l'efficacité du routage géographique a été mise en œuvre avec la mobilité des nœuds). L'évolution de leur topologie a fait l'objet d'un nombre important d'études et de travaux. Plusieurs protocoles de routage ont été étudiés pour les réseaux mobiles en général, et les réseaux véhiculaires en particulier (un bilan des routages pour les réseaux véhiculaires est montré dans [49]).

Comme nous avons déjà vu, un réseau ad hoc est un ensemble de nœuds mobiles qui se déplacent arbitrairement et dynamiquement dans le réseau. La connexion entre les nœuds change à tout moment. Dans la plupart des cas, la destination ne se trouve pas obligatoirement dans la portée de la source, ce qui implique que l'échange des données entre deux nœuds quelconques, doit être effectué à l'aide de nœuds intermédiaires.

Pour cela, le réseau doit donc s'organiser automatiquement et réagir rapidement aux différents mouvements des nœuds. Chaque nœud est alors un relai potentiel, et contribue ainsi au routage des informations. De nombreux protocoles de routage ont été proposés pour les réseaux mobiles. Ces protocoles sont classés en deux catégories principales, les protocoles proactifs et les protocoles réactifs. Les protocoles proactifs établissent les routes à l'avance en se basant sur l'échange périodique de tables de routage alors que les protocoles réactifs recherchent les routes à la demande du réseau. Il existe une troisième approche, dite hybride, qui combine les deux approches précédentes.

Routages proactifs

Les protocoles de routage proactifs essaient de maintenir les meilleurs chemins existants vers toutes les destinations possibles, autrement dit l'ensemble de tous les nœuds du réseau. Ces chemins sont construits et gardés au niveau de chaque nœud du réseau. Les routes sont sauvegardées mêmes si elles ne sont pas utilisées. La construction et la mise à jour permanentes des chemins de routage sont assurées par un échange continu des messages de mise à jour des chemins. Cet échange induit un contrôle excessif surtout dans le cas des réseaux de grande taille. L'avantage de ce type de routage est qu'une

route est disponible pour chaque destination à tout moment. Au moment où les nœuds auront besoin d'une route, elle sera prête et disponible. Mais cela est coûteux en messages échangés régulièrement, ce qui induit une consommation excessive de la bande passante. Ces messages de contrôle ne sont pas tous nécessaires car seules quelques routes sont utilisées dans le réseau généralement.

Parmi les divers protocoles de routage proactifs, nous citons OLSR (*Optimized Link State Routing*), TBRPF (*Topology Broadcast Based on Reverse-Path Forwarding*), FSR (*Fisheye State Routing*), DSDV (*Dynamic Destination Sequenced Distance Vector*), B.A.T.M.A.N (*Better Approach To Mobile Ad hoc Networking*)....

Comme son nom l'indique, OLSR est un protocole à état de lien optimisé ; il obtient aussi des routes de plus court chemin. Alors que dans un protocole à état de lien, chaque nœud déclare ses liens directs avec ses voisins à tout le réseau, dans le cas d'OLSR, les nœuds ne déclarent qu'une sous-partie de leur voisinage grâce à la technique des relais multipoints. Cette technique consiste essentiellement, en un nœud donné, à ignorer un ensemble de liens et de voisins directs, qui sont redondants pour le calcul des routes de plus court chemin : plus précisément, dans l'ensemble des voisins d'un nœud, seul un sous-ensemble des ces voisins est considéré comme pertinent. Il est choisi de façon à pouvoir atteindre tout le voisinage à deux sauts (tous les voisins des voisins), cet ensemble est appelé l'ensemble des relais multipoints. Un algorithme de calcul de relais multipoints est donné dans [61]. Ces relais multipoints sont utilisés de deux façons : pour diminuer le trafic dû à la diffusion des messages de contrôle dans le réseau, et aussi pour diminuer le sous-ensemble des liens diffusés à tout le réseau puisque les routes sont construites à base de relais multipoints. La figure 4.1 donne un exemple de gain en nombre de retransmissions sur un graphe simple. Supposons qu'un nœud émette un message, et que pour diffuser cette information au réseau, ses voisins répètent cette information.

FIGURE 4.1 – Dans le premier graphique, à gauche, où tous les voisins d'un nœud retransmettent, six répétitions (les nœuds en noirs) sont nécessaires. Par contre, en utilisant la retransmission par les relais multipoints seuls (à droite), on économise deux retransmissions.

Pour maintenir à jour toutes les informations nécessaires au choix des relais multipoints et le calcul de la table de routage, les nœuds OLSR ont besoin de s'échanger

des informations périodiquement. Pour s'informer du proche voisinage, les nœuds OLSR envoient périodiquement des messages dits HELLO contenant la liste de leurs voisins. Ces messages permettent à chacun de choisir son ensemble de relais multipoints. Le deuxième type de message de OLSR sont les messages TC (Topology Control). Par ce message les sous-ensembles de voisinage que constituent les relais multipoints sont déclarés périodiquement dans le réseau. Ils sont diffusés en utilisant une diffusion optimisée par relais mulitpoints. Ces informations offrent une carte de réseau contenant tous les nœuds et un ensemble partiel des liens, mais suffisant pour la construction de la table de routage. La table de routage est calculée par chacun des nœuds et le routage des données s'effectue saut par saut sans l'intervention d'OLSR dont son rôle s'arrête à la mise à jour de la table de routage de la pile IP.

Le protocole BATMAN [2] a été conçu pour remplacer OLSR. Cependant, il porte sur une connaissance décentralisée de la route optimale. Chaque nœud sauvegarde la connaissance sur le prochain saut et la provenance du message reçu.

Le protocole TBRPF [20, 55] se base sur l'échange d'information portant sur une partie de la topologie stockée dans la table de topologie de chaque nœud. Chaque nœud construit son arbre de la topologie en utilisant une modification de l'algorithme Djikistra. En effet, comme pour OLSR, chaque nœud envoie seulement une partie de son "arbre source", celle ci fournissant des routes à tous les nœuds accessibles, pour limiter la consommation de la bande passante. Cependant, TBRPF utilise après une combinaison de mises à jour périodiques et différentielles pour informer tous les voisins de cette partie. Pour découvrir le voisinage, le protocole TBRPF emploi les messages différentiels ≪ HELLO ≫ qui contiennent juste les changements de statut des voisins. Ces messages sont beaucoup plus petits en comparaison avec ceux utilisés par d'autres protocoles proactifs. Or un nœud peut, à tout moment, envoyer des informations supplémentaires de topologie, jusqu'à la topologie complète, pour améliorer la fiabilité dans les réseaux fortement mobiles.

Le protocole DSDV [45], de son côté, repose sur des transmissions périodiques de toute la table de routage, et des transmissions des changements importants de routes. Cependant, il est lent et admet beaucoup d'*overhead*.

Le protocole FSR [60] optimise les échanges en regroupant les mises à jour et en les agrégeant en un seul message. Il assure une grande précision sur les voisins proches, et une moins bonne précision sur les voisins lointains. L'importance et la précision se dégradent avec la distance. Sur la base des informations de voisinage et de topologie, FSR construit sa table de routage, le chemin à parcourir se précise en se rapprochant de la destination. FSR n'inonde pas le réseau avec des mises à jour, mais, périodiquement tous les nœuds échangent avec ses voisins sa table de topologie. En effet, l'ensemble des liens se propage saut par saut à chaque envoi.

Le protocole WRP [54] se base sur le principe que chaque nœud connaît le nombre minimum de liens qui le sépare de tous les autres nœuds. Chaque nœud stocke ces informations dans une "table de distances", et possède à côté une table de routage, une table de distances et une table de coûts des liens. Il admet également une vérification de la constance des voisins. A chaque changement d'état, le nœud envoie une mise à jour à

ses voisins qui à leur tour envoient la mise à jour et ainsi de suite. La caractéristique du WRP est que chaque nœud vérifie la consistance des voisins à chaque changement de lien voisin détecté. Ceci aide à éliminer les situations des boucles de routage et à minimiser le temps de convergence du protocole.

Routages réactifs

Les protocoles réactifs, quant à eux, ne gardent que les routes en cours d'utilisation pour le routage. A la demande, le protocole va chercher à travers le réseau une route pour atteindre une nouvelle destination. Ce protocole ne construit une route que sur la demande de la part d'un nœud qui veut communiquer avec une destination distante. Cette technique permet de ne pas inonder le réseau par de paquets de contrôle et de ne conserver que les routes utilisées.

Le protocole AODV est un exemple de protocole réactif. AODV est un protocole basé sur le principe des vecteurs de distance. Lorsqu'un nœud cherche une route vers une destination, il diffuse une demande de route à travers le réseau. Les nœuds qui reçoivent ces messages les diffusent à leur tour jusqu'à atteindre un nœud qui possède une information de routage récente vers la destination recherchée ou jusqu'à atteindre la destination elle-même. Les nœuds qui relaient des informations de routage mettent également à jour leurs informations de routage grâce aux informations reçues dans les messages. AODV construit les routes à l'aide de messages de type "route request" (RREQ) et "route reply" (RREP). Lorsque la source désire établir une route vers une destination, elle diffuse le paquet de demande de route à ses voisins. Les nœuds recevant le paquet mettent à jour leur information relative à la source et établissent des pointeurs de retour vers la source dans les tables de routage. Un nœud recevant un RREQ émettra un RREP s'il est la destination, ou s'il possède une route vers la destination avec un numéro de séquence supérieur ou égal à celui repris dans le RREQ. Si tel est le cas, il envoie en *unicast* un paquet RREP vers la source. Sinon, il rediffuse le RREQ. S'ils reçoivent un RREQ qu'ils ont déjà traité, ils ne le retransmettent pas. Les nœuds établissent des pointeurs de propagation vers la destination, alors que les RREP reviennent vers la source. Une fois que la source a reçu les RREP, elle peut commencer à émettre des paquets de données vers la destination. Si, ultérieurement, la source reçoit un RREP contenant un numéro de séquence supérieur ou bien le même, mais avec un nombre de sauts plus petits, elle mettra à jour son information de routage vers cette destination et commencera à utiliser la meilleure route. Une route est maintenue aussi longtemps qu'elle continue à être active. Une route est considérée active tant que des paquets de données transitent périodiquement de la source à la destination selon ce chemin. Si un lien se rompt alors qu'une route est active, le nœud extrémité du lien cassé émet un paquet "route error" (RERR) vers la source pour lui indiquer que la destination est désormais non atteignable. Après réception de RERR, si la source désire toujours la route, elle peut ré-initier un processus de découverte de route.

DSR se base sur la technique de routage par la source. Ceci signifie que le nœud source doit déterminer le chemin à travers lequel les paquets de données seront envoyés. En effet, le nœud source diffuse un paquet "Route Request". Si ce paquet réussit à trouver

le nœud destinataire, le nœud source reçoit un paquet "Route Response" contenant un enregistrement de la séquence des nœuds visités durant la propagation de la requête dans le réseau. Puisque le chemin est déterminé par le nœud source, les nœuds intermédiaires n'ont pas besoin de garder en mémoire les informations du chemin, puisque les paquets de données contiennent déjà les décisions de routage [39]. DSR utilise 2 mécanismes pour acheminer les paquets d'un nœud source S à nœud destinataire D. Le premier mécanisme cherche une route entre S et D. Si la route n'existe pas dans le cache de S, celui-ci inonde le réseau par une requête de recherche de route. Cette requête est relayée vers tous les nœuds du réseau, jusqu'à aboutir au nœud D ou jusqu'à arriver à un nœud qui contient dans son cache la route vers la destination. La liste des nœuds traversés est alors inversée afin de retourner un message de réponse "route response" au nœud S. Le deuxième mécanisme s'engage à maintenir la route enregistrée dans S vers D mise a jour. En effet quand un lien est cassé et la route déjà définie n'est plus valable, un paquet d'erreur est envoyé au nœud source S qui va essayer de trouver un autre chemin dans son cache. Sinon, il renvoie une requête de recherche de chemin.

Le protocole TORA [58] (*Temporally Ordered Routing Algorithm*)résiste au changement de la topologie en stockant plusieurs chemins vers une même destination. Ceci permet d'utiliser un chemin alternatif vers la destination quand le chemin initial est perdu suite à un changement de topologie, sauf si tous les chemins vers la destination sont rompus. TORA se caractérise par ses messages de contrôle limités à un nombre réduit de nœuds. Avec TORA, l'utilisation des meilleurs chemins est secondaire. Parfois, les longs chemins sont choisis pour éviter le contrôle induit par le processus de découverte de nouveaux chemins. TORA utilise la propriété des graphes acycliques orientés appelée "orientation destination". En effet, un graphe acyclique orienté (DAG) est "orienté destination" s'il existe toujours un chemin possible vers la destination spécifiée. Si un lien (ou plus) se casse, le graphe devient "non orienté destination". TORA assure la transformation du graphe en un graphe "oriente destination" durant un temps infini. Pour arriver a ce but, il utilise le concept d'"inversement de liens" ; il utilise la notion de "taille" de nœud. En fait, chaque nœud possède une taille qu'il échange avec l'ensemble de ses voisins directs. Un lien est toujours orienté du nœud possédant la plus grande taille vers celui ayant une plus petite taille. Quand un nœud intermédiaire détecte une rupture, il lance une requête vers la source pour l'alarmer de la défaillance du chemin. Les nœuds qui reçoivent cette requête d'alarme modifient leur taille afin de garder le DAG "oriente destination".

Routages hybrides

Les protocoles hybrides combinent les deux approches. Ils utilisent en premier lieu un protocole proactif, pour avoir des informations sur le proche voisinage (voisinage à deux ou trois sauts) ; ainsi ils disposent des routes immédiates dans le voisinage. Au-delà de ce voisinage à quelques sauts, le protocole hybride fait appel aux techniques des protocoles réactifs pour chercher des routes vers les destinations à plusieurs sauts. Avec ce découpage, le réseau est partagé en plusieurs zones, et la recherche de route en mode réactif peut être améliorée. A la réception d'une requête de recherche réactive, un nœud

peut informer immédiatement si la destination est dans son voisinage ou pas, et par conséquence, il saura s'il faut aiguiller ladite requête vers les autres zones sans déranger le reste de sa zone. Ce type de protocole s'adapte bien aux grands réseaux. Cependant, il adopte les inconvénients des protocoles réactifs et proactifs. Les messages de contrôle périodique sont coûteux en bande passante, et de plus, la recherche d'une route vers une destination lointaine implique un échange important de messages, surtout si cette recherche implique une destination différente de celle dans la requête précédente.

Le protocole ZRP (*Zone Routing Protocol*) est un exemple de routage hybride. Il met en place, simultanément, un routage proactif et un routage réactif, afin de combiner les avantages des deux approches. Pour ce faire, il passe par un concept de découpage du réseau en différentes zones, appelées "zones de routage". Une zone de routage est alors définie pour chaque nœud, elle inclut les nœuds qui sont à une distance minimale (en terme de nombre de sauts), du nœud en question, inférieure ou égale au rayon δ de la zone [59].

Routages géographiques

L'idée du routage géographique est d'utiliser des informations géographiques pour acheminer les messages de la source vers la destination. Nous supposerons que tous les nœuds connaissent leur position et que ceux-ci connaissent également la position de tous les autres nœuds du réseau. Avec ces hypothèses, il est facile de considérer qu'un nœud peut facilement choisir parmi ses voisins un relais, généralement le plus proche de la destination, pour acheminer un paquet dont il connaît la destination finale et donc aussi sa position. Il est très simple d'envisager des critères de sélection parmi ces voisins ; donnons quelques exemples parmi les heuristiques les plus classiques.

Les protocoles de routage géographiques ont comme caractéristique commune deux étapes distinctes : la localisation d'un nœud destinataire et l'acheminement ou le routage des paquets vers ce nœud. Dans la première étape, il est nécessaire que le nœud source détermine la position géographique du nœud destinataire avant d'envoyer des paquets d'information. Pour cette opération de découverte de la position géographique, un service de localisation doit être utilisé [51]. Dans le contexte des réseaux ad hoc mobiles à large échelle, la sélection d'un service de localisation efficace est un problème difficile vu l'absence d'une infrastructure fixe.

Le routage d'un message par un nœud repose essentiellement sur la position de ses voisins immédiats et la position du nœud destinataire. Le nœud est supposé connaître sa position géographique via les données GPS. Les positions des voisins sont connues grâce aux envois périodiques d'un nœud de sa position à ses voisins immédiats. En d'autres termes, la connaissance du voisinage est périodiquement mise à jour à l'aide de messages échangés entre les nœuds voisins.

Le GPSR (*Greedy Perimeter Stateless Routing*) se base sur l'approche "Greedy Geographic Forwading". Celle ci consiste à choisir le nœud le plus proche de la destination, parmi les nœuds voisins, comme prochain saut. Pour mesurer la proximité des nœuds au nœud destination, il est possible d'utiliser la distance euclidienne comme référence. Mais, il se peut qu'il n'y ait aucun nœud plus proche de la destination que le nœud

actuel. Dans ce cas, la distance entre lui et la destination étant plus grande que la portée de son réseau sans fil, le routage s'arrête chez lui et la communication est perdue. Le GPSR emploie la règle de la main droite pour remédier au problème ci-dessus. En effet lorsque le nœud ne trouve pas un voisin plus proche que lui de la destination, il trace une ligne virtuelle entre lui et la destination et la fait tourner vers la droite, centrée sur lui-même. Le premier nœud rencontre par cette ligne est choisi comme le prochain saut.

Dans le protocole DREAM [19] (*Distance Routing Effect Algorithm for Mobility*), chaque nœud du réseau ad hoc échange des messages de contrôle "Location Path" (LP), pour informer les nœuds voisins de sa localisation. Pour réduire le nombre de paquets de contrôle, la fréquence des messages est inversement proportionnel à la distance entre les nœuds : en grande fréquence pour les nœuds proches, en plus petite fréquence pour les nœuds lointains. Ces fréquences dépendent en plus de la vitesse de déplacement du nœud source S. Lorsque le nœud source souhaite communiquer avec un nœud destination D, il dessine un cercle de centre D et d'un rayon proportionnel à la vitesse de D. Ensuite il construit un cône de sommet S et de côtés tangentes au cercle déjà dessiné. Ensuite, S envoie son paquet de données aux nœuds inclus dans cette zone. Si cette zone ne contient pas de nœuds voisins, S inonde le réseau entier avec son paquet de données. Dans le cas où la zone dessinée par le cône contient des nœuds voisins, une liste qui contient leurs identificateurs sera ajoutée à l'entête du message. Seuls les nœuds dont les identificateurs sont spécifiés dans l'entête traitent le message. Le nœud, traitant le message, détermine les nœuds du prochain saut de la même manière, et envoie le paquet avec une nouvelle liste dans l'entête. Si aucun nœud n'est trouvé, le paquet reçu est ignoré. Ainsi, le paquet de données arrive peu à peu à D. Ce dernier, à la réception du message, renvoie un acquittement à S en suivant le même chemin parcouru par le paquet. Si S ne reçoit pas un acquittement pendant un temps limité, il inonde tout le réseau avec le paquet non acquitté. Si D reçoit le paquet suite à une inondation, il ne renvoie pas d'acquittement.

Routages dédiés VANETs

De nombreux routages ont été conçus pour les réseaux véhiculaires (une étude peut être trouvée dans [49]). Nous pouvons en citer, pour le routage *unicast*, le protocole *min delay*, le *Greedy perimeter coordinator routing protocol*, basé sur la géolocalisation et profitant des rues et des intersections pour réparer le chemin, le VADD (*Vehicule-assited data delivery routing protocol*) qui est un protocole *store and forward* d'un véhicule à un autre jusqu'à la destination statique ou CAR (*Connectivity-aware routing protocol*) qui établit des chemins en marquant comme repères les véhicules qui se trouvent sur des jonctions des routes. DIR (*diagonal-intersection-based routing protocol*) adopte le même principe que CAR sauf que les points de repères sont l'intersection entre les jonctions des routes et la diagonale entre la source et la destination. Avec protocole ROMSGP (*Receive on Most Stable Group-Path*), les véhicules sont divisés en 4 groupes suivant leur vecteur de vitesse. Le routage est dit stable si les deux véhicules appartiennent à un même groupe. Le GVGrid est un protocole qui garantit la qualité de service. Il divise la carte en plusieurs grilles et choisit une suite de grilles selon la direction et la

distance entre les véhicules et les intersections. Le véhicule dont la vitesse est la plus grande est choisi pour transmettre les messages à la grille suivante. Quand un lien est cassé, GVGrid cherche directement un autre véhicule dans la grille. D'autres protocoles existent pour les communications *unicast*, comme le *Delay-Bounded Routing Protocol* ou le *Reliable routing for roadside to vehicle communications*. Cependant, ces protocoles ont leurs limitations ; la construction du chemin et la recherche de la destination sont souvent considérées comme effectuées simultanément. D'autre part, la plupart de ces protocoles sont développés dans des milieux urbains supposant une grande densité du réseau. La mise à l'échelle est aussi un point critique. Des protocoles pour des routages *multicast* ou *broadcast* existent aussi, comme *Distributed Robust GEocast Multicast Routing Protocol* par exemple pour le *multicast*, ou le DV-CAST (*Broadcasting in VANET*) pour le *broadcast*. Cependant, la mise à l'échelle est un problème pour ces protocoles, ainsi que la dissémination dans un réseau non dense (dans le cas des *broadcast*).

Les transmissions conditionnelles [30, 31] représentent un autre exemple de routage dédié aux réseaux véhiculaires. En fait, la destination dans un réseau véhiculaire est souvent indiquée par une sorte de conditions, comme par exemple "les voitures qui suivent l'émetteur dans le convoi", "les voitures qui précèdent l'émetteur dans le convoi", "les voitures situées dans une zone géographique donnée", "les voitures admettant une connexion vers un réseau externe", ou "les voitures ayant une identité donnée", etc.

L'adressage semble être obsolète dans les réseaux à forte dynamique. Les adresses réseaux sont souvent formées à partir de sa position dans le réseau. Cependant, cette position risque de changer souvent avec la mobilité des nœuds. Par conséquence, l'utilisation des adresses réseaux devient coûteuse [44]. De ce fait, dans la plateforme *Airplug*, les nœuds sont plutôt adressés par leur identité ou par des conditions que par des adresses réseaux. L'identité des nœuds doit être unique, formée à partir de l'adresse MAC par exemple, ou bien choisie aléatoirement afin de préserver la vie privée des utilisateurs.

Pour surmonter le problème d'adressage réseau, l'adressage par conditions remplace l'adressage réseau, les transmissions conditionnelles remplacent le *broadcast*, et le maintien de chemin, objet des études et recherches développées dans ce manuscrit, remplace l'adressage *unicast* traditionnel.

La transmission conditionnelle [31] se fait par un ajout de conditions à évaluer au message à envoyer. Ces conditions sont évaluées par les nœuds voisins de l'émetteur à la réception du message envoyé. Seuls les voisins vérifiant les conditions requises transmettent le message à leur couche applicative. Ces conditions peuvent être des positions ou zones géographiques (les conditions géographiques rapprochent les transmissions conditionnelles des algorithmes de routage géographique), ou une identité (ce qui définit une communication *unicast* en spécifiant une destination précise). La décision de retransmission est orientée récepteur, qui lui, va décider en évaluant les conditions s'il transmet le message à d'autres nœuds et/ou le transmet à la couche applicative.

Les conditions à transmettre avec le message sont au nombre de deux, appelées *upward condition* (CUP) et *forward condition* (CFW). Quand CUP est vraie, le nœud passe le message à la couche applicative. Quand CFW est vraie, il relaie le message à ses voisins.

Une implémentation du service de transmissions conditionnelles (appelée HOP) a été développée comme composant essentiel de la plateforme *Airplug*. Une adaptation à NS-2 a été également implémentée [47].

HOP permet un routage "opportuniste" vers une destination qui n'est pas connue d'avance. Il repose sur la transmission de messages vers des destinataires connus par des conditions qu'ils doivent satisfaire. L'apport qu'assure l'algorithme de maintien de chemin par rapport à HOP, qui représente un routage *one to many*, est que le maintien de chemin implique une communication de nature *unicast* seulement avec une destination dont l'identité est bien connue. L'algorithme de maintien de chemin vient alors compléter HOP pour essayer de couvrir les différents aspects de la communication dans un réseau mobile.

4.1.2 Contribution

Comme nous l'avons résumé dans le chapitre 1, la gestion de la communication dans les réseaux mobiles est confrontée à des difficultés résultant de la dynamique des entités communicantes. Cette propriété de dynamique rend difficile le maintien d'une communication dans le réseau. Une communication entre deux entités dans un réseau nécessite un routage. Ce routage construit un réseau logique complet sur un réseau physique incomplet. Le routage peut s'effectuer par contenu, par adresses... Pour les communications *unicast*, le routage est généralement basé sur l'identifiant des nœuds, accompagné de leur adresse ou position dans le réseau. Le problème principal d'un routage basé sur adresses est qu'il repose sur des données à durée de validité limitée. En fait, la position du nœud est variable avec sa mobilité. La fréquence de changement de cette position dépend de la dynamique du réseau. En conséquence, quand la dynamique est forte, les adresses pour le routage sont assez souvent mises à jour. Pour cette raison, une table d'adressage ou de routage est inefficace car les routeurs ou les relais changeront sans cesse d'identité. La connaissance de l'adresse courante du destinataire est aussi un problème quand la dynamique est forte. Cette connaissance est coûteuse en terme de messages. Ce coût s'avère plus important quand la dynamique est forte car la fréquence d'échange de messages augmente pour maintenir la validité des informations collectées. Dans [44], ces problématiques sont détaillées. L'inefficacité d'un routage ou d'une localisation de destination est argumentée. Dans ce chapitre, nous proposons une approche plus simple et moins ambitieuse pour le routage dans les réseaux dynamiques. Cette approche est le maintien de chemin. Les entités considérées sont des entités voisines, donc se connaissent, ce qui évite la recherche de l'adresse de la destination. Nous économisons alors en nombre de messages échangés et en délai de recherche d'adresse.

Les protocoles de routages cités dans la sous-section précédente, sont très coûteux en messages de contrôle. Même s'ils arrivent à construire un chemin vers une destination distante, leur limitation reste liée à la dynamique des réseaux dans lesquels ils sont implémentés. Dans une très forte dynamique, même le protocole de routage le plus efficace n'aboutira pas.

Un besoin de maintenir la route de la source vers la destination est essentiel pour une communication *unicast* dans un réseau mobile. Au niveau du routage, plusieurs

protocoles ont été proposés pour construire et maintenir une telle route dans un réseau mobile [52]. Cependant, quand la dynamique est très forte, aucun algorithme n'arrive à satisfaire ses spécifications. Cette observation résulte de plusieurs travaux antérieurs [30, 18], mais il reste évident de déduire que plus la dynamique est forte et que le voisinage est instable, moins il est possible de maintenir une route entre deux entités.

Les protocoles de routage existants, dans leurs spécifications, assurent le maintien de chemin quand le chemin se casse. Cependant, ce maintien de chemin repose sur les mêmes principes que pour la construction du chemin. Ce qui veut dire un excès de messages de contrôle et une inefficacité quand le réseau est fortement mobile.

Nous proposons dans ce chapitre un algorithme de maintien de chemin basé sur un échange de message limité entre nœuds du chemin. Ce chemin se construit quand les entités voisines s'éloignent.

La solution proposée consiste à maintenir une communication existante dans un réseau dynamique. Elle évite tous les aspects conséquents à la mobilité des nœuds : routage, adressage, recherche de destination... Elle suppose dans un premier temps l'existence d'une communication entre deux entités proches dans le même voisinage. Le maintien de chemin intervient lorsque la communication est interrompue.

L'algorithme de maintien de chemin est un algorithme qui s'affranchit du problème d'adressage vu qu'il n'est pas basé sur des adresses mais sur des identités, et qu'il n'utilise pas de table de routage. Au contraire, il cherche à maintenir un chemin pour pouvoir acheminer les données de la source vers la destination sans avoir à garder une connaissance globale du réseau. Le maintien de chemin se fait localement sur les nœuds du chemin. Ce chemin est mis à jour à chaque changement ajout ou retrait de nœuds relais.

Comme il s'agit d'un algorithme qui se base sur l'intervention des nœuds proches des deux entités pour maintenir la communication et assurer le transfert des informations, il implique uniquement les nœuds voisins du chemin. Ils serviront de relais et entreront en action quand les deux nœuds initialement impliqués dans la communication s'éloigneront l'un de l'autre. Quand les différents nœuds présents dans le voisinage détectent la coupure du lien, ils se proposent comme intermédiaires. L'émetteur choisit un nœud parmi ceux qui se proposent. La communication peut alors se poursuivre normalement.

Les points forts de l'algorithme concernent le fait qu'il gère l'intégration et la suppression des nœuds intermédiaires du chemin.

Cette gestion se fait par un chemin sauvegardé sur chaque nœud impliqué dans la conversation. Ce chemin est stocké sous la forme d'une liste des nœuds prédécesseurs et une autre des nœuds successeurs dans le chemin. Ces listes sont formées à partir des différents messages reçus du voisinage et contenant des listes semblables. Les listes reçues permettent de détecter les arrivées et départs de nœuds intermédiaires du chemin.

Une fois le chemin construit, et la continuité de la communication assurée, on peut alors s'intéresser au contrôle de flux. Ce contrôle peut être assuré par un protocole de transport adapté aux réseaux dynamiques.

Le fonctionnement de l'algorithme de maintien de chemin diffère de ce qui existe dans la littérature par le fait que la connaissance et la réparation du chemin sont réalisées localement car tout *broadcast* global au niveau du réseau est exclu.

4.2 L'algorithme de maintien de chemin

Dans cette section nous présentons l'algorithme de maintien de chemin et ses différents mécanismes.

L'algorithme de maintien de chemin, dit PTH (pour PaTH), repose sur trois principes : la mise à jour périodique, l'extension du chemin et la réduction du chemin.

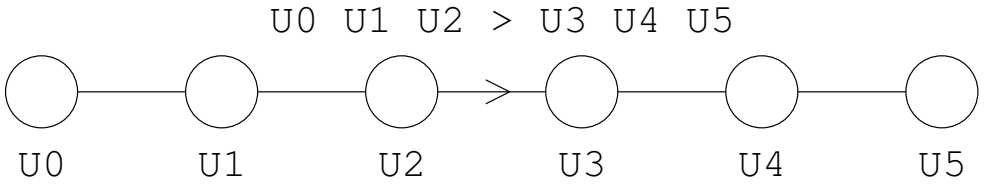


FIGURE 4.2 – Exemple de message échangé lors de la mise à jour du chemin.

Mise à jour périodique

Afin de faire face aux changements de voisinage, PTH repose sur l'émission périodique de messages dans le voisinage des nœuds du chemin. Seuls ces nœuds envoient des messages sur les liens. Un exemple des messages échangés est illustré dans la figure 4.2.

Ces messages contiennent des informations locales sur le chemin. Ces informations sont formées à partir de la liste des nœuds formant le chemin (par exemple $u_0u_1u_2u_3\ldots$), quelques drapeaux permettant de déterminer l'émetteur et le récepteur potentiel du message en cours (par exemple $u_0u_1 > u_2u_3$ où u_1 est l'émetteur et u_2 le récepteur), et un drapeau d'incertitude sur un membre du chemin ($u_0u_1 > u_2?u_3$ où u_1 a une incertitude sur son successeur u_2). Un tel message informe le prédécesseur et le successeur du nœud de l'état local du chemin. Ces voisins peuvent alors mettre à jour leurs informations locales à leur tour. D'autres nœuds du réseau, n'appartenant pas au chemin, peuvent se proposer pour appartenir à ce chemin, afin de le réparer ou de le raccourcir. Ce phénomène est expliqué à la suite. Il faut cependant noter que les informations relatives à des nœuds distants ne sont pas forcément à jour. Cela ne perturbe pas l'utilisation du chemin localement, car cette information n'est utilisée que par le mécanisme de réduction du chemin (et est vérifiée à ce moment-là, voir ci-dessous), et à titre informatif pour les nœuds source et puits.

Extension du chemin

Le mécanisme d'extension de chemin permet de substituer les arêtes cassées. Ceci peut apparaître quand deux voisins s'éloignent l'un de l'autre. Supposons qu'une arête (u_i, u_j) du chemin se casse. Comme le nœud u_i envoie périodiquement des messages contenant $u_i > u_j$, il attend l'acquittement implicite de u_j. Cet acquittement est acheminé par des messages sans drapeau d'incertitude, contenant $u_iu_j >$. Si le nœud u_i ne

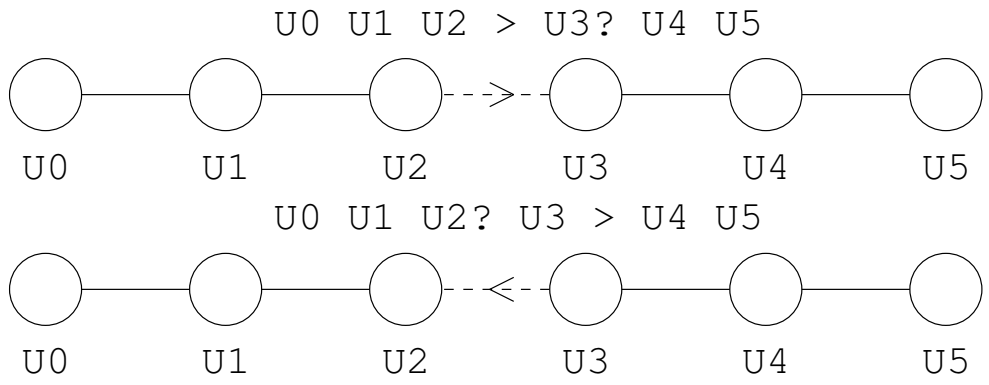


FIGURE 4.3 – Exemple des messages échangés par les deux extrémités lorsqu'un lien se casse.

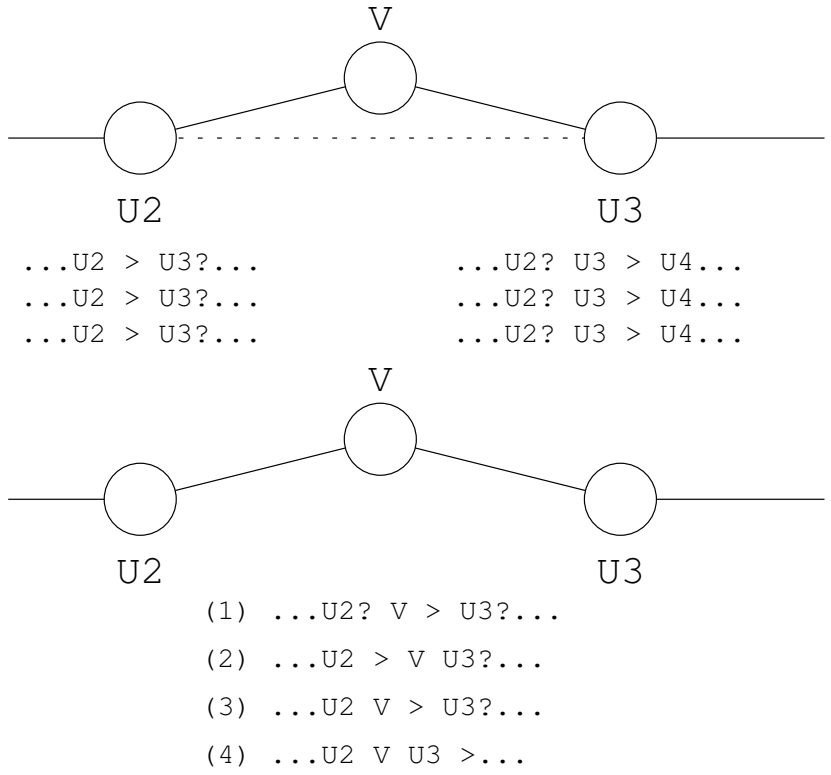


FIGURE 4.4 – Exemple des messages échangés lors de l'extension du chemin.

reçoit pas de tels messages de la part de u_j, il positionne le drapeau d'incertitude sur son nœud successeur u_j dans ses prochains messages, qui contiennent maintenant $u_i > u_j$?.

Un exemple de ces messages est illustré dans la figure 4.3.

Soit u_j reçoit des messages de u_i avec un drapeau d'incertitude sur son identité (échec de la communication $u_i \leftarrow u_j$), soit il ne reçoit aucun message de u_i (échec de la communication $u_i \rightarrow u_j$). Dans les deux cas, il positionne un drapeau d'incertitude sur son prédécesseur u_i dans ses messages. Ces derniers contiendront alors $u_i?u_j >$.

Les drapeaux d'incertitudes permettent aux voisins de u_i et de u_j de proposer la réparation du chemin. Si un nœud voisin v reçoit les deux messages $u_i > u_j?$ et $u_i?u_j >$, il commence à les compter. Quand le compte atteint un seuil donné, la panne de communication entre u_i et u_j est confirmée et v se propose pour devenir relai de la communication en envoyant $u_i?v > u_j?$. Comme plusieurs nœuds du voisinage de u_i et de u_j peuvent se proposer pour devenir relais, le nœud u_i (qui est plus près de la source) choisit le premier voisin qui se propose (disons v) et envoie $u_i > vu_j?$. À la suite, v envoie $u_iv > u_j?$. A la réception de ce message, le nœud u_j envoie $u_ivu_j >$ et le chemin est réparé par insertion de v comme relai de communication entre u_i et u_j.

Ce mécanisme est illustré en exemple dans la figure 4.4.

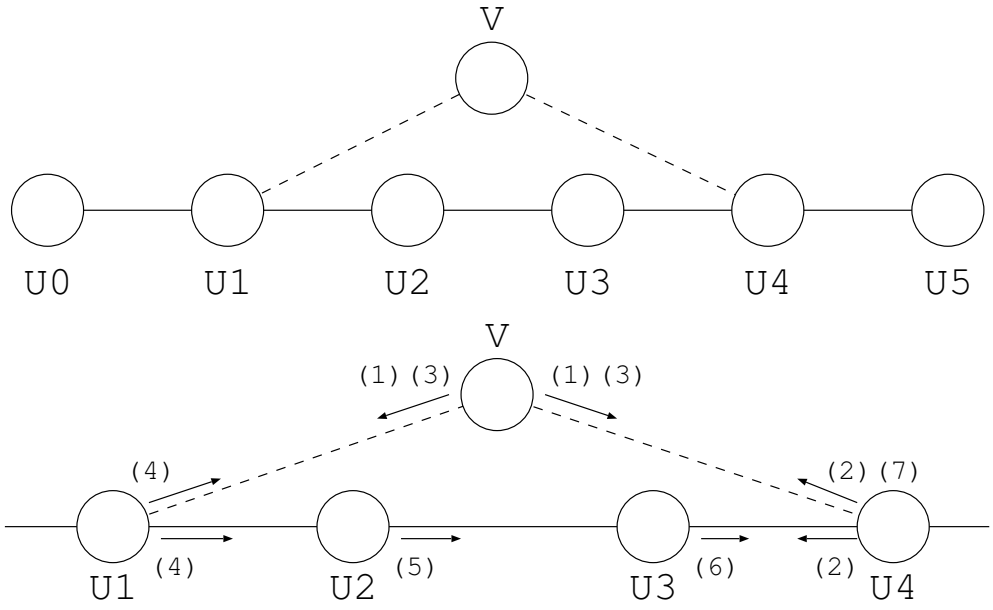


FIGURE 4.5 – Un exemple des messages échangés lors de la réduction d'un chemin.

Réduction du chemin

Le mécanisme de réduction de chemin consiste à réduire le chemin quand c'est possible pour garder le chemin le plus court possible tout en impliquant uniquement les nœuds de la route et leurs voisins. La réduction peut être effectuée par un nœud de la route ou par un voisin qui n'appartient pas au chemin mais est voisin d'au moins deux nœuds du chemin.

Considérons le second cas : deux nœuds u_i et u_j appartiennent à la route avec u_i plus proche de la source que u_j, et soit v un voisin commun de u_i et u_j tel que v n'appartient pas au chemin. Supposons que le nœud v reçoit les messages envoyés par u_i et contenant $u_i > u_{i+1} \dots u_j$, et les messages envoyés par u_j et contenant $u_i \dots u_{j-1} u_j > u_{j+1}$.

Il peut alors en déduire que la réduction est possible et proposera à la fois à u_i et u_j d'être un relai entre eux au lieu d'utiliser les nœuds $u_{i+1} u_{i+2} \dots u_{j-1}$.

A cet effet, le nœud v commence à envoyer des messages contenant des informations qu'il a apprises sur le chemin avec la proposition de réduction, notée $u_i{-}?v{-}?u_j$, avec le drapeau d'incertitude sur les liens (u_i, v) et (v, u_j). Les messages de v contiennent alors $u_i{-}?v{-}?u_j u_{i+1} \dots u_{j-1} v > u_j$. Lorsque u_j reçoit un tel message, il confirme à v la possibilité de réduire en omettant le drapeau d'incertitude sur l'arête (v, u_j) : ses prochains messages contiennent alors $u_i{-}?v - u_j u_{i+1} \dots u_{j-1} u_j > u_{j+1}$. A la réception d'un tel message, le nœud v met à jour le contenu de son propre message en omettant le drapeau incertitude sur l'arête (v, u_j), car elle a été confirmée par u_j. Ses messages suivants contiennent alors $u_i{-}?v - u_j u_{i+1} \dots u_{j-1} v > u_j$.

Pendant ce temps, le nœud u_i reçoit des messages envoyés par v et contenant la proposition de réduction. Cependant, il les ignore jusqu'à ce que u_j confirme le lien (v, u_j). Une fois c'est fait, le nœud u_i confirme à son tour le lien (u_i, v). Ses prochains messages contiennent alors $u_i{-}v{-}u_j > u_{i+1} \dots u_{j-1} u_j$. Ils sont adressés à son successeur actuel dans le chemin. L'information sur la réduction sera ensuite transmise de nœud en nœud tout le long du chemin jusqu'à ce qu'il atteigne u_j. À ce niveau, le nœud u_j envoie des messages contenant $u_i v u_j > u_{j+1}$, confirmant la réduction. v met à jour alors ses autres messages, qui contiennent maintenant $u_i v > u_j$, confirmant à son tour la réduction. Enfin u_i met à jour ses messages à son tour, confirmant la réduction : ils contiennent maintenant $u_i > v u_j$. Le chemin est à la suite réduit entre u_i et u_j.

Ce processus de réduction est nécessaire parce que plusieurs réductions se chevauchant peuvent se produire simultanément. Les nœuds du chemin, entre u_i et u_j doivent valider la réduction proposée par v. Quand ils transmettent la proposition, ils deviennent engagés. Le nœud u_i devient lui-même engagé lors de la transmission de la réduction à u_{i+1}. Au cas où un de ces nœuds est déjà engagé avec une autre réduction, il ignore tout simplement la proposition reçue et ne la propage pas à son successeur dans le chemin. Ainsi u_j ne reçoit jamais la réduction et ne la confirme jamais. Ceci permet d'éviter un fragmentation du chemin en plusieurs petits chemins déconnectés faute de réductions qui se chevauchent simultanément : il existe un *mutex* sur la partie du chemin partagée par les réductions concurrentes. A noter également que, dans le cas où les arêtes (u_i, v) et (v, u_j) n'existent pas assez longtemps, la confirmation par u_j, ensuite relayée par v et enfin par u_i, ne se produira jamais et le chemin reste inchangé. Le chemin n'est pas

alors perturbé par un voisin v temporaire qui peut bien proposer une réduction avant de s'éloigner.

Le même processus est utilisé lorsque la réduction est proposée par le nœud u_j appartenant au chemin au lieu d'un nœud voisin v.

Le mécanisme de réduction par un nœud voisin est illustré dans la figure 4.5.

4.3 Conclusion

Le maintien de chemin est une nouvelle stratégie permettant les communications *unicast* dans les réseaux dynamiques. Elle ne nécessite pas d'adresses réseau mais demande à ce que les deux protagonistes soient proches l'un de l'autre au départ. Cette hypothèse est très réaliste en égard aux applications envisagées dans les réseaux dynamiques.

Dans ce chapitre, nous avons présenté le principe de l'algorithme de maintien de chemin que nous proposons, ainsi que ses mécanismes de fonctionnement.

L'étude de cet algorithme nous a conduit à une modélisation des réseaux dynamiques, et à une nouvelle métrique de cette dynamique, que nous introduisons dans le chapitre suivant.

Chapitre 5

Modélisation des réseaux dynamiques

Pour étudier notre algorithme de maintien de chemin, nous avons été amenés à modéliser les réseaux dynamiques d'une part, et les limites d'un algorithme dans un réseau dynamique d'une autre part. Pour cela, nous reprenons le concept d'algorithme *best effort* [33], qui propose un contrat entre la dynamique d'un réseau et l'algorithme, et nous exprimons des propriétés à l'aide de notre modélisation.

Ce formalisme est utilisé pour établir les limites et les propriétés de notre algorithme de maintien de chemin.

5.1 Modélisation

L'étude formelle de l'algorithme de maintien de chemin nous a menés vers une modélisation des réseaux dynamiques. Il nous a fallu ainsi définir une métrique algorithmique de cette dynamique.

5.1.1 Travaux antécédents

Les réseaux dynamiques ont été étudiés pour leurs propriétés structurelles [43] [21]. Dans [22] [36], la notion de "graphe évolutif" est introduite en introduisant un domaine temporel dans la théorie des graphes, afin de capturer la nature changeante de la dynamique des réseaux.

Entre autres concepts introduits, la journée est un chemin dans des topologies successives (chemin spatio-temporel).

Dans [53], les graphes évolutifs sont utilisés pour concevoir des algorithmes de routage.

Dans [63], les graphes d'accessibilité temporelle sont introduits, où un lien existe quand un voyage existe entre les deux extrémités. Par comparaison, notre modèle repose sur une famille de graphes dynamiques, chacun permettant d'envoyer un certain nombre de messages par lien.

Plusieurs travaux traitent la relation entre la dynamique d'un réseau et la faisabilité d'un algorithme distribué. Dans [57], quelques contraintes algorithmiques sont exigées et étudiées afin qu'un nœud réussisse à acheminer son message vers une destination donnée. Dans [26], les computations locales distribuées par les "relabeling" des graphes et les graphes évoluants sont utilisés pour étudier, comparer et analyser les algorithmes distribués. Dans [27], une synthèse est fournie à travers la plateforme unifiée appelée TVG (*time varying graph*). Plusieurs classes de TVG sont identifiées et l'impact de leurs propriétés sur les algorithmes distribués est aussi étudié.

Dans [48], la relation entre la présence d'un sous-graphe stable connecté durant des périodes successives et la faisabilité et la complexité de plusieurs problèmes d'ordonnancement distribués, est étudiée. Un algorithme rapide de dissémination d'informations est proposé dans [42], utilisant le codage réseau. Dans [16], un nouveau modèle est introduit pour étudier la dissémination d'informations dans les réseaux mobiles. Il repose sur deux paramètres α et β tels que dans α tranches de temps, quelques nœuds retenant l'information sont connectés à d'autres n'ayant pas l'information pour au moins β tranches de temps. Des conditions ont été validées sur α et β, garantissant la convergence.

Par comparaison, nos conditions de faisabilité sont définies utilisant les graphes p-dynamiques. De plus, la relation entre la dynamique du réseau et les propriétés d'un algorithme est fournie par un compromis exprimé par les spécifications *best effort*.

5.1.2 Contribution

Comme nous l'avons déjà dit, quand la dynamique d'un réseau est très forte, les algorithmes ne peuvent plus satisfaire leurs spécifications. Un contrat entre la dynamique

du réseau et les performances d'un algorithme doit être établi. Ce contrat est caractérisé par l'approche *best effort*.

Pour ce faire, nous exprimons des propriétés caractérisant la topologie et le service offert par l'algorithme. Nous utilisons pour cela une modélisation des réseaux dynamiques. Nous proposons alors dans ce chapitre un modèle des graphes dynamiques qui les caractérise suivant le nombre de messages qu'ils permettent de transmettre sur leurs liens. D'après cette caractéristique, nous arrivons à définir la dynamique maximale tolérable pour qu'une communication puisse être maintenue.

Nous définissons dans ce chapitre la modélisation du réseau, l'approche *best effort*, ainsi que la métrique de dynamique du réseau.

5.2 Système distribué et réseau dynamique

Dans cette section, nous allons décrire les composants d'un système, le système distribué et le réseau dynamique.

5.2.1 Système distribué

Considérons un système distribué $\mathcal{S}$ composé d'un ensemble de nœuds communicants. Chaque nœud possède une mémoire locale et une unité de calcul séquentiel le rendant capable de lancer des algorithmes localement. La mémoire locale d'un nœud est formée de sa mémoire privée $PRIV_v$, d'une mémoire entrante IN_v et d'une mémoire sortante OUT_v. Les nœuds restent cependant asynchrones.

Afin de communiquer, les nœuds possèdent des outils de communication. Ces communications s'effectuent à travers une primitive de communication simple, dénotée *push*. Quand un nœud effectue un $push(m)$, la valeur de m est sauvegardée dans la mémoire sortante du nœud et copiée dans les mémoires entrantes des différents nœuds récepteurs $v1$, $v2$, ..., vk. Afin qu'un nœud récepteur v puisse recevoir le message envoyé par un nœud u, il faut que plusieurs conditions soient satisfaites. Ces conditions sont fortement liées à la technologie et au protocole de communication et peuvent alors varier. Nous pouvons en citer quelques unes, comme la distance limite, le moyen d'accès disponible, l'absence de collisions aves les autres émetteurs, etc.

Les nœuds récepteurs d'une action *push* effectuée par un nœud u ne connaissent pas u et ne sont pas connus par u. Ces récepteurs sont déterminés suivant des conditions de réception sur le lien de communications. Ces conditions peuvent varier d'une émission à une autre et d'un message à un autre, pour un même nœud v. Nous définissons un lien de communication (u, v) entre u et v si un message m envoyé par une action *push* par u est reçu par v. La disparition de ce lien est observée quand v ne reçoit pas les messages envoyés par u. Un lien (u, v) peut exister alors que le lien (v, u) n'existe pas. La capacité d'un lien est définie comme étant un seul message.

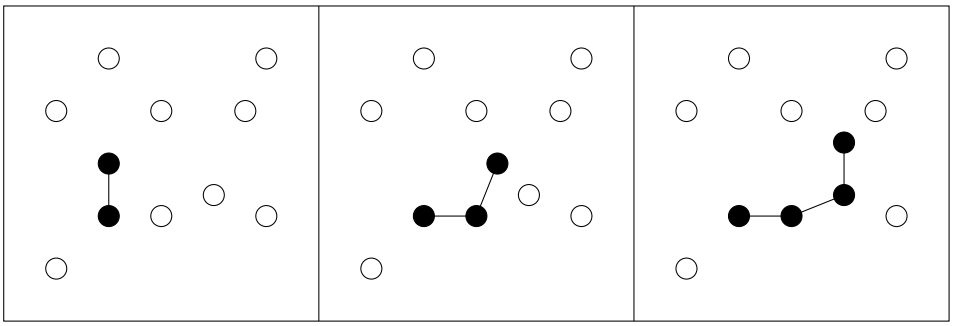


FIGURE 5.1 – Illustration d'un réseau mobile. Les nœuds changent de positions et de voisinage, des liens sont ainsi créés et cassés.

5.2.2 Réseau dynamique

Un réseau est composé de plusieurs liens de communication. Le système distribué est dynamique quand le réseau est dynamique, autrement dit, si les nœuds sont en mouvement et que les liens apparaissent et disparaissent. Ce modèle de communication peut être appliqué à un réseau sans fil avec une capacité d'un message par lien. Une action *push* est implémentée en utilisant un *broadcast* suivi par un repos plus long que la durée maximale de communication (valeur plafonnée par les protocoles sans fil, comme le IEEE 802.11 par exemple). Avec les mouvements des nœuds, des liens se créent et se cassent suivant la portée de la communication (illustré dans la figure 5.1).

5.3 Métrique algorithmique

Un réseau peut être modélisé par une métrique liée au fonctionnement et au déploiement de l'algorithme dans ce réseau.

Un réseau dynamique est caractérisé par les nœuds en mouvement et leur moyen de communications. Ce moyen est fourni par les outils de communication et des protocoles des couches basses qui définissent les échanges. Pour caractériser la dynamique d'un réseau d'un point de vue algorithmique, le mouvement des nœuds ainsi que le moyen de communication représentent des facteurs clés. En fait, quand les nœuds bougent très rapidement, le réseau peut être considéré comme dynamique. Cependant, d'un point de vue algorithmique, et si les communications sont efficaces et fournissent les conditions nécessaires au bon fonctionnement de l'algorithme, la mesure de la dynamique devient moins évidente. En effet, les nœuds peuvent échanger un nombre de messages important avant que leurs voisinages ne changent. Dans le cas inverse, si le moyen de communication est instable, même si le réseau n'est pas fortement mobile, l'échange de données peut échouer, et la dynamique est alors considérée comme très forte. Dans ce contexte, nous pouvons dire qu'un algorithme distribué peut ne pas satisfaire ses spécifications. Il pourra alors ne pas pouvoir effectuer un calcul ou diverger de son comportement attendu. Cette

déstabilisation vient du fait que très peu d'échanges ont eu lieu entre les nœuds et leurs voisins.

Nous proposons alors une modélisation du réseau par des graphes p-dynamiques, dont la définition compense le mouvement des nœuds et l'efficacité des liens de communication. Cette modélisation permet alors de caractériser le réseau d'un point de vue algorithmique. D'autre part, elle permet de caractériser la capacité d'un réseau distribué à tolérer une dynamique donnée. Elle permet alors de définir une métrique algorithmique de la dynamique du réseau. Ceci est nécessaire pour comparer plusieurs types de réseaux avec leur dynamique.

Considérons un système distribué dans un réseau ad hoc mobile. L'évolution de la topologie d'une exécution donnée, peut être définie par une famille de graphes p-dynamiques. Cette famille, notée $\{(G_i^p)_{i\in\mathbb{N}}, p \in \mathbb{N}\}$, est formée d'une succession de graphe G_1^p, G_2^p ... où chaque graphe G_i^p est un graphe p-dynamique dont les liens durent assez de temps pour pouvoir acheminer p messages. Cette modélisation par les liens p-dynamiques concrétise la mobilité du réseau aussi bien que les pertes sur les liens pour pouvoir à la suite comparer les performances d'un algorithme donné dans différentes dynamiques de réseaux et reposant sur différentes technologies. De plus, la variation de la valeur de p pour une exécution donnée, donne lieu à plusieurs familles différentes de graphes p-dynamiques. La valeur de p, imposée pour un bon fonctionnement de l'algorithme, définie la dynamique tolérée dans le réseau où sera implémenté cet algorithme.

Cela dit, nous pouvons conclure qu'une dynamique très forte peut empêcher un algorithme de satisfaire ses spécifications. Aucun algorithme ne peut compenser n'importe quel degré de dynamique. Il existe alors un compromis entre les performances d'un algorithme et la dynamique du réseau dans lequel il est déployé. Ce compromis peut être illustré par l'approche *best effort*.

5.4 Approche *best effort*

Cette approche établit un compromis entre les performances d'un algorithme et la dynamique du réseau, et cela en définissant une condition liée à la dynamique du réseau et qui sera définie par une propriété topologique, et une autre liée aux performances de l'algorithme, et sera définie par une propriété de continuité. La propriété topologique établit une contrainte sur les changements topologiques du réseau, et cite les conditions nécessaires à respecter pour un bon fonctionnement de l'algorithme. D'autre part, la propriété de continuité définie les spécifications que doit respecter l'algorithme, et qui constituent le service que doit fournir cet algorithme.

Ces spécifications d'un système distribué sont données par un ensemble de prédicats définis sur les exécutions. Le modèle *best effort* est exprimé à l'aide de deux propriétés, une propriété topologique dite P_T, et une autre de continuité dite P_C. Ces deux propriétés sont à ajuster suivant les cas d'applications, et changent d'algorithme en algorithme et de réseau en réseau. Dans [33], les spécifications *best effort* ont été utilisées pour compléter le paradigme d'auto-stabilisation.

5.5 Spécifications *best effort*

Les spécifications *best effort* sont définies sur une exécution du système. Autrement dit, elles sont définies sur une suite de configurations.

Une configuration d'un système distribué $\mathcal{S}$ est une instance des états des processeurs et des liens. L'ensemble des configurations de $\mathcal{S}$ est noté $\mathcal{C}$. Un algorithme distribué est un ensemble d'algorithmes locaux tournant sur tous les nœuds de $\mathcal{S}$. Les actions sur les processeurs et les mémoires changent la configuration du système. Une exécution e est une séquence de configurations c_1, c_2, ... La configuration c_1 est considérée comme la configuration initiale de l'exécution e.

Notons $P_T(e)$ le prédicat sur la topologie. $P_T(e)$ est définie sur les exécutions de $\mathbb{E}$ et qualifie l'évolution de la topologie du système $\mathcal{S}$ durant une exécution donnée e. Notons $P_C(e)$ le prédicat sur la continuité. $P_C(e)$ est définie sur les exécutions et qualifie une propriété d'un système $\mathcal{S}$ exécutant un algorithme donné. Les spécifications *best effort* sont définies par $P_T(e) \Rightarrow P_C(e)$. Quand un algorithme satisfait les propriétés *best effort*, il garantie que la propriété de continuité est garantie quand l'évolution de la topologie respecte P_T.

5.6 Graphes temporisés

Dans un graphe dynamique, le temps de communication entre deux voisins est important. Pour cette raison, nous introduisons une modélisation des réseaux dynamiques qui inclut le temps. Nous nous intéressons à l'échange de plusieurs messages successifs sur un lien. Nous considérons qu'il s'avère nécessaire au fonctionnement de certains algorithmes que p messages envoyés successivement soient reçus à l'autre extrémité du lien. Nous considérons alors le temps de transition de ces p messages dans notre modélisation. Cette modélisation sera utilisée pour définir les graphes p-dynamiques dans la section suivante.

Definition 1 *La fonction de durée de transfert $\delta : \mathbb{N} \to \mathbb{R}$ est définie par : $\delta(p)$ est égale au temps nécessaire pour le transfert de p messages successifs entre deux nœuds.*

Cette définition dépend évidemment de la technologie de communication des couches inférieures. La proposition suivante reste valide :

Proposition 2 *Soit $\delta : \mathbb{N} \to \mathbb{R}$ une fonction de durée de transfert. Alors $\delta(p) \leq p \times \delta(1)$.*

Preuve 1 *Supposons le cas inverse. Alors pour envoyer p messages, il est possible de les envoyer l'un à la suite de l'autre sur le lien. Ceci nécessite un délai égal à $p \times \delta(1)$, ce qui contredit $\delta(p) > p \times \delta(1)$.*

En utilisant la fonction de durée de transfert, nous introduisons les liens p-temporisé afin de modéliser le fait qu'il soit possible de transférer p messages entre deux nœuds à un temps donné. Dans ce manuscrit, le "lien" fait référence à un lien de communication

observé quand une communication a pu être réalisée. Autrement dit, soit une communication a eu lieu, soit elle aurait pu avoir lieu au cas où un message a été envoyé par l'émetteur par l'action *push* (ainsi que les conditions nécessaires comme la portée par exemple). Cependant, "arête" fait référence à une modélisation de graphe d'un système distribué. Un lien de communication ne peut être observé s'il n'existe pas d'arête entre l'émetteur et le récepteur. Cela représente une condition nécessaire pour l'établissement d'une communication. Une arête peut cependant exister entre deux nœuds alors qu'il n'existe pas de lien de communication ; une action *push* n'a pas été accomplie ou bien elle a échoué car les conditions nécessaires ne sont pas satisfaites au complet.

Afin de pouvoir introduire les liens p-temporisés, nous introduisons un observateur imaginaire et externe qui permet de noter tous les évènements dans le réseau dynamique. Un tel observateur utilise son horloge interne pour dater tous ces évènements. Les nœuds restent cependant asynchrones sans connaissance d'une horloge globale commune. Cet observateur produit une observation qui permet de noter et de dater à l'aide de sa propre horloge toutes les créations et les disparitions des arêtes dans le réseau.

Definition 3 *Considérons un observateur externe d'un réseau dynamique admettant $\delta()$ comme fonction de durée de transfert. Il existe un lien p-temporisé noté $(u, v)^{t,p}$ entre les nœuds u et v du réseau à la date t si, au cas où u (respectivement v) aurait envoyé p messages à la date $t - \delta(p)$, alors v (respectivement u) aurait reçu le dernier message à la date t.*

Autrement dit, un lien p-temporisé existe assez longtemps pour permettre le transfert de p messages. À noter qu'un lien p-temporisé $(u, v)^{t,p}$ peut exister entre u et v alors qu'aucun message n'est échangé sur ce lien. L'observateur observe les liens capables de transférer p messages si envoyés. À noter aussi qu'un lien $(u, v)^{t,p}$ peut exister sans qu'une communication entre u et v ne puisse aboutir si les conditions nécessaires ne sont pas satisfaites. La durée d'un lien en est un exemple. Un cas particulier est illustré par les réseaux sans fil dans lesquels les réceptions multiples ne sont pas possibles.

Quand la valeur de p est très grande, il est possible qu'aucun lien p-temporisé ne soit observé. Cela veut dire que la dynamique est très élevée pour admettre le transfert de p messages successifs. D'autre part, si la valeur de $\delta(1)$ est très élevée, ce qui veut dire que le protocole d'échange n'est pas assez efficace, il se peut qu'aucun lien 1-temporisé ne puisse être observé. En déduction, suivant un moyen de communication donné, et des liens de communication exigés, il est possible de déduire la dynamique maximale admise pour un réseau donné. Réciproquement, suivant une dynamique donnée d'un réseau, il est possible de déduire la valeur maximale de la fonction de durée de transfert requise afin de choisir et de déterminer le moyen de communication nécessaire pour la durée de lien exigée.

Partant d'une définition des liens p-temporisés, nous définissons les graphes p-temporisés.

Definition 4 *Considérons un observateur externe d'un réseau dynamique. Le graphe p-temporisé à la date t, noté $G^{t,p}(V, E^t, p)$, est défini comme suit : les sommets $v \in V$ correspondent aux nœuds du réseau ; les liens de $E^{t,p}$ sont les liens p-temporisés à la date t.*

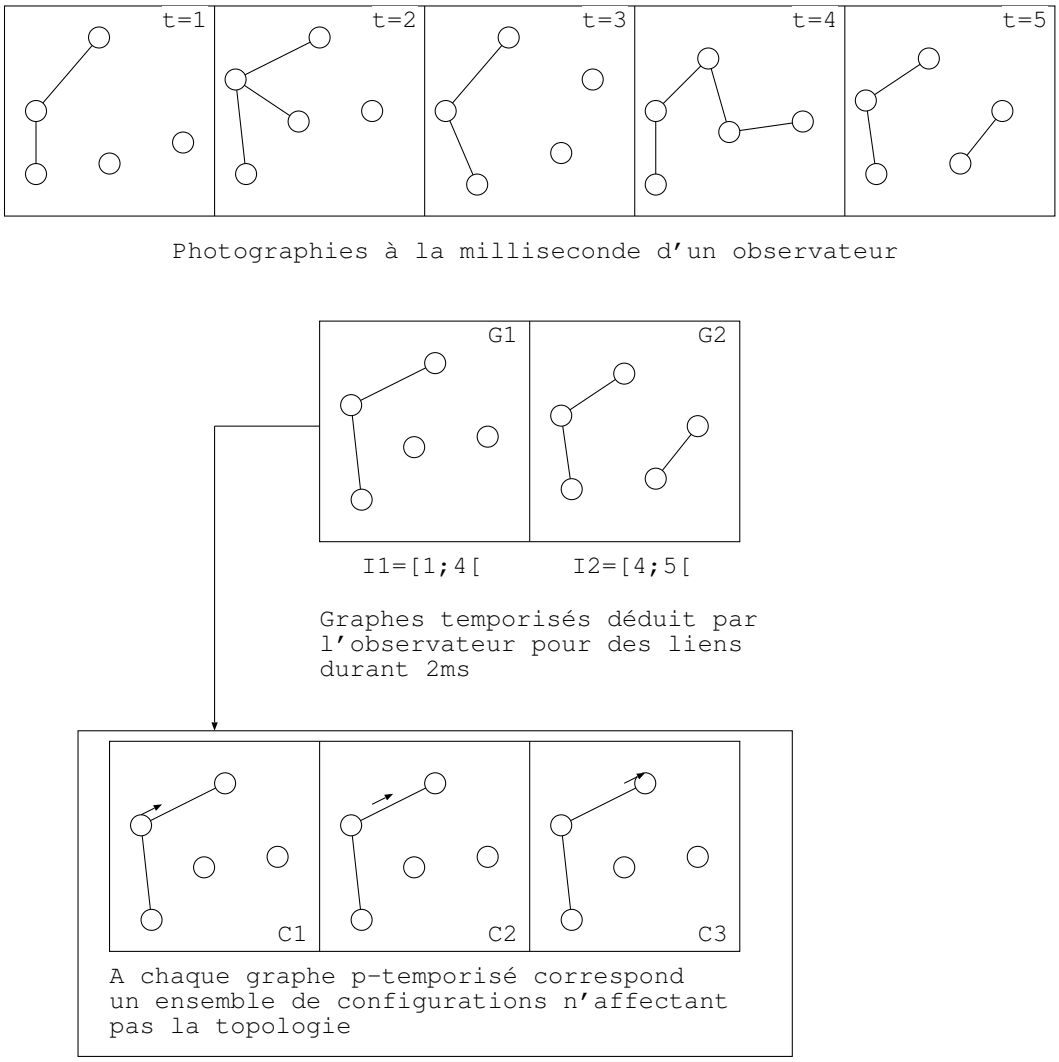


FIGURE 5.2 – Exemple de construction de graphes temporisés pour une observation donnée (à la ms) et une durée de lien de 2 ms. Les graphes p-temporisés G^1 et G^2 sont repérés par un observateur avec des durées de lien de 2 ms. A chacun de ces graphes peuvent correspondre plusieurs configurations.

5.7 Graphes p-dynamiques

Un graphe dynamique, ayant un ensemble de sommets V et noté $(G_k)_{k\in\mathbb{N}}$, est défini par une séquence infinie de graphes $G_\kappa(V, E_\kappa)$ avec $E_\kappa \subset V \times V$. Il est assez évident de modéliser le réseau dynamique par le biais des graphes dynamiques [36, 26, 27]. Ces travaux modélisent le réseau par des graphes en considérant l'échange d'un message sur le lien. Nous proposons d'étendre ce modèle en considérant l'échange de plusieurs

messages sur le lien.

Le temps et l'observateur ne forment pas un facteur commun à tous les algorithmes distribués afin de faire des analyses et des comparaisons. Nous introduisons alors les intervalles de stabilité afin de définir les graphes p-dynamiques.

Definition 5 *Considérons un observateur externe d'un réseau dynamique. Soit* $(G^{\tau,p})_{\tau\in\mathbb{R}+}$ *la séquence des graphes p-dynamiques observés. Les intervalles de stabilité* I^p *de* $(G^{\tau,p})_{\tau\in\mathbb{R}+}$ *est une séquence d'intervalles* $I^p = [t_0, t_1[, [t_1, t_2[\ldots$ *définie par :*

- $t_0 = 0$ *;*
- *pour chaque intervalle* $[t_i, t_{i+1}[$ *de* I_p*, et pour chaque date* $t \in [t_i, t_{i+1}[$*, le graphe p-temporisé à la date t est égal au graphe p-temporisé à la date ti :* $G^{t,p} = G^{t_i,p}$ *;*
- $G^{t_{i+1},p} \neq G^{t_i,p}$.

Il faut noter que tout intervalle dans I_p a une durée supérieure ou égale à $\delta(p)$ car chaque lien doit durer au moins $\delta(p)$ pour qu'il figure parmi les liens observés dans le réseau.

Utilisant les intervalles de stabilité, nous définissons les graphes p-dynamiques d'une observation. Le graphe dynamique est alors une séquence de graphes, mais qui sont désignés par un observateur donné, et où les liens permettent l'envoi de p messages. Une illustration des graphes p-temporisés et de leurs intervalles de stabilité est montrée dans la figure 5.2.

Definition 6 *Considérons un observateur externe d'un réseau dynamique. Soit* $(G^{\tau,p})_{\tau\in\mathbb{R}+}$ *une séquence de graphes p-dynamiques observés, et* $I^p = [t_0, t_1[, [t_1, t_2[\ldots$ *ses intervalles de stabilité. Le graphe dynamique* $(G_i)_{i\in\mathbb{N}}$ *noté par* $G_i = Gt_i, p$ *définit le graphe p-dynamique de l'observation.*

En considérant toutes les observations possibles, un graphe p-dynamique unique peut être défini. En effet, nous considérons plusieurs observations avec des fréquences différentes et assez petites. En les rassemblant, nous évitons de louper des évènements.

Proposition 7 *L'ensemble de toutes les observations possibles définit un graphe p-dynamique unique pour un entier p non nul donné.*

Preuve 2 *Soit* $\bar{o}$ *l'union de toutes les observations réalisées durant une exécution d'un système distribué* $\mathcal{S}$*. Une telle observation contient tous les évènements observés par les observateurs externes. Vu que* $\bar{o}$ *est unique, le graphe p-dynamique résultant, et donc les intervalles de stabilité résultants, sont uniques. Notons par* $(G^{\tau,p}_{\bar{o}})_{\tau\in\mathbb{R}+}$ *et* $I^p_{\bar{o}}$ *la famille de graphe p-dynamique et les intervalles de stabilité.* $(G^p_i)_{i\in\mathbb{N}}$ *est alors défini d'une façon unique par :* $G^p_0 = G^{t_0,p}_{\bar{o}}$, $G^p_1 = G^{t_1,p}_{\bar{o}}$, $G^p_2 = G^{t_2,p}_{\bar{o}}$, $\ldots$ *où* $[t_0, t_1[$, $[t_1, t_2[$, $\ldots$ *sont les intervalles de stabilité de* $I^p_{\bar{o}}$.

Ce graphe p-dynamique unique, ne dépend pas d'un observateur particulier ; il est plutôt lié au réseau en lui même.

Definition 8 *Nous appelons graphe p-dynamique d'un réseau, le graphe p-dynamique* $(G_i^p)_{i\in\mathbb{N}}$ *défini dans la proposition 7, utilisant l'observation unique* $\bar{o}$*. Nous appelons p-graphes, les graphes composant ce graphe p-dynamique.*

Par conséquence, pour toute exécution du système distribué $\mathcal{S}$ d'un réseau dynamique et ayant une fonction de durée de transfert $\delta()$ donnée, il existe un graphe dynamique unique $(G_i^p)_{i\in\mathbb{N}}$ pour un entier $p > 0$ donné. Nous introduisons maintenant la famille de graphes p-dynamiques en prenant en considération tous les entiers cités.

Definition 9 *Nous appelons la famille de graphes p-dynamiques d'un réseau, l'ensemble* $\{(G_i^p)_{i\in\mathbb{N}} \quad p \in \mathbb{N}*\}$ *de tous les graphes p-dynamiques correspondants à toutes les valeurs non nulles de l'entier p.*

Nous considérons que la famille des graphes p-dynamiques caractérise à part entière le réseau dynamique. Elle permet la comparaison entre différents réseaux dynamiques quel que soit le moyen de communication utilisé (technologie et protocoles de communication) et quelle soit la vitesse de mouvement des nœuds. En outre, cette modélisation constitue, comme nous le verrons dans les sections suivantes, un outil assez performant pour analyser le comportement d'un algorithme dans une dynamique donnée et sa capacité à tolérer une telle dynamique.

5.8 Propriétés des graphes p-dynamiques

D'après la définition des configurations, tout changement dans le système distribué mène à une nouvelle configuration. Il existe alors une topologie par configuration. Nous nous intéressons aux communications possibles à réaliser avant ou après une configuration donnée. Nous introduisons alors les *p-graphes de/à partir d'une configuration.* Ils sont définis à la suite.

Definition 10 *Soit* t_c *la date durant l'observation* $\bar{o}$ *où le système distribué atteint une configuration donnée c. Le p-graphe à la configuration c, noté* $G^p_{c|}$ *est le graphe de* $(G_i^p)_{i\in\mathbb{N}}$ *observé à la date* t_c*. Le p-graphe à partir de la configuration c, noté* $G^p_{|c}$*, est le graphe de* $(G_i^p)_{i\in\mathbb{N}}$ *observé à la date* $t_c + \delta(p)$*.*

Le graphe $G^p_{|c}$ est composé de toutes les arêtes permettant l'envoi de p messages à partir d'une configuration c. Durant une exécution $e = e_1, e_2, \ldots$, le graphe $G^1_{|c_1}$ représente la topologie initiale. Il est important de noter que la topologie initiale d'un système dynamique distribué ne peut être observé avant un délai $\delta(1)$, empêchant toute possibilité de communication. En effet, la topologie permettrai d'envoyer un nombre quelconque de messages, en particuliers, ceux nécessaires au fonctionnement de l'algorithme. Un système distribué ne peut être défini comme dynamique si $G^1_{|c_1} = G^p_{|c_1}$ pour tout entier $p > 0$. La proposition suivante est directe ($G(V, E) \subseteq G'(V, E')$ signifie que toute arête de E appartient à E').

Proposition 11 *Soit* $\{(G_i^p)_{i\in\mathbb{N}} \quad p \in \mathbb{N}\}$ *la famille des graphes p-dynamiques d'un réseau donné, et* q, $q' \in \mathbb{N}*$ *deux entiers non nuls. Pour toute configuration c d'une exécution donnée e, si* $q > q'$, *alors* $G^q_{|c} \subseteq G^{q'}_{|c}$ *et* $G^q_{c|} \subseteq G^{q'}_{c|}$.

Non validité des informations à plus de p sauts dans un graphe p-dynamique La famille de graphes p-dynamiques permet d'indiquer la possibilité d'obtenir ou non des informations locales valides sur la topologie à plusieurs sauts. Cette validité consiste à dire qu'à chaque changement de topologie, cette information doit être vraie à sa réception. Prenons l'exemple d'un nœud u qui veut communiquer à un nœud v distant l'arrivée d'un nouveau voisin u'. Quand u' est toujours voisin de u au moment où v reçoit l'information, cette information est dite *valide*.

Proposition 12 *Soit* $p > 1$ *un entier. Si* $(G_i^{p-1})_{i\in\mathbb{N}*} \neq (G_i^p)_{i\in\mathbb{N}*}$, *alors aucun algorithme distribué ne peut procurer à chaque nœud du réseau une information valide sur la topologie à plus de p sauts.*

Preuve 3 *Considérons deux nœuds u et v dans un graphe fixe tel que la distance* $dist(u, v) = p$. *L'envoi d'un message de u vers v nécessite un délai au moins égal à* $p \times \delta(p)$. *Comme* $\delta(p) \leq p \times \delta(1)$ *(comme figure dans la proposition 2), un délai d'au moins* $\delta(p)$ *est nécessaire. Si* $(G_i^{p-1})_{i\in\mathbb{N}*} \neq (G_i^p)_{i\in\mathbb{N}*}$, *il existe alors une arête d'une durée inférieure à* $\delta(p)$. *Considérons par exemple que* (u, u') *représente cette arête. Le nœud v ne reçoit pas l'information avant un délai d'au moins* $\delta(p)$. *L'arête* (u, u') *n'existe plus au moment où v reçoit l'information sur l'arrivée de* u'.

La proposition 12 fournit une indication sur la possibilité de l'utilisation des informations locales sur les changements topologiques pour des propos de routage. En effet, en étudiant les graphes p-dynamiques pour un réseau dynamique donné, il est possible alors de déduire la distance maximale à partir de laquelle le transfert d'informations locales n'est plus intéressent.

5.9 Problème du maintien du chemin

Le *problème de maintien de chemin*, comme défini précédemment, consiste à maintenir un chemin entre deux entités alors que les liens se créent et se cassent.

Dans cette section, nous présentons ce problème et nous expliquons comment le définir dans le cadre d'un réseau dynamique en utilisant les spécifications *best effort* et les graphes p-dynamiques.

5.9.1 Définition du problème

Le *problème de maintien de chemin* consiste à maintenir un chemin ou une route entre deux nœuds voisins sans besoin d'information à plus d'un saut. Une *route* est alors une succession de choix de nœuds relais permettant l'accès à la destination 13.

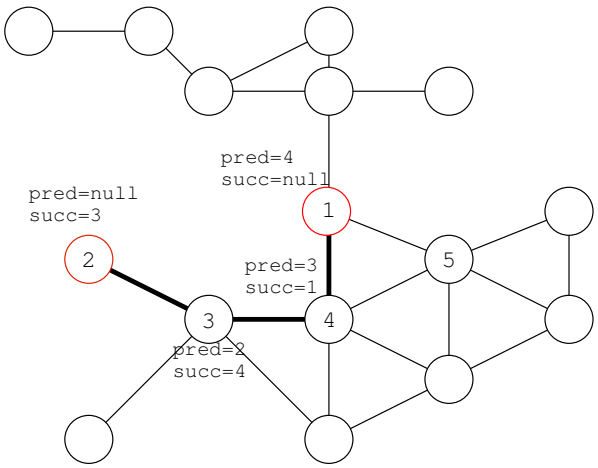


FIGURE 5.3 – Illustration du chemin virtuel par l'intermédiaire des variables *pred* et *succ* sur les nœuds du chemin.

Appelons *source* et *puits* les deux extrémités de cette route. Vu que seules les informations locales sont sauvegardées sur tout nœud v du réseau, deux variables locales $pred_v$ et $succ_v$ sont définies pour cet usage. Si v n'appartient pas au chemin, les variables $pred_v$ et $succ_v$ ont des valeurs nulles. Sinon, $pred_v$ désigne le prédécesseur de v dans le chemin, et $succ_v$ désigne son successeur. $Pred_v$ est nulle pour la source, et $succ_v$ est nulle pour la destination.

Le but de l'algorithme de maintien de chemin est de gérer les variables $pred_v$ et $succ_v$ localement sur chaque nœud v du système distribué $\mathcal{S}$ afin de préserver la route entre le nœud source et le nœud puits. Il faut noter que la source et le puits ont été voisins avec $pred_{puits} = source$ et $succ_{source} = puits$. Cette gestion doit se faire en se reposant sur les informations locales uniquement, donc les informations sur le voisinage du nœud v.

5.9.2 Spécifications du problème

Considérons un système distribué $\mathcal{S}$, une configuration c d'une exécution donnée, et un graphe $G^1_{c|}$ qui modélise la topologie du réseau à la configuration c, nous introduisons le prédicat du *chemin* (illustration dans la figure 5.3).

Definition 13 *Le prédicat du $n-$chemin (n-route) $R^n_{u,v}(c)$ est vraie pour une configuration c si et seulement si il existe des nœuds $u_0, u_1, \ldots, u_n$ dans $G^1_{c|}$ tels que :*

- $u_0 = u$, $pred_{u_0} = null$, $succ_{u_0} = u_1$;
- $pred_{u_i} = u_{i-1}$, $succ_{u_i} = u_{i+1}$ *pour tout* $i \in \{1, \ldots, n-1\}$;
- $u_n = v$, $pred_v = u_{n-1}$, $succ_v = null$;
- $pred_w = succ_w = null$ *pour tout nœud* $w \in V \backslash \{u_0, \ldots, u_n\}$.

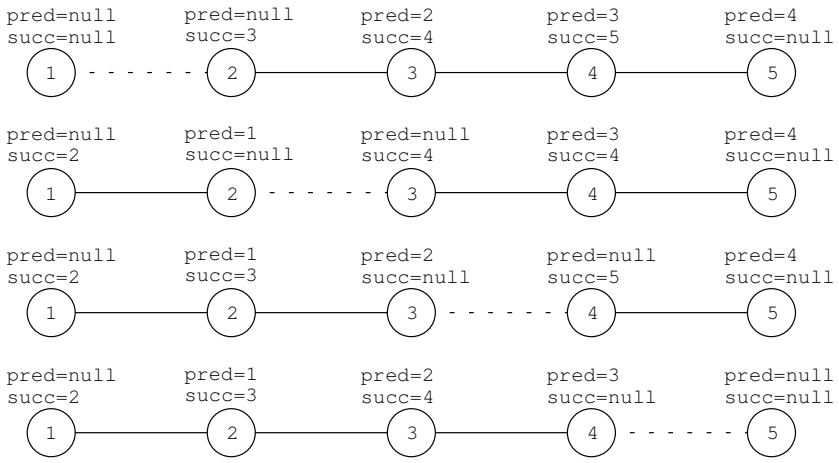


FIGURE 5.4 – La propriété de continuité est définie telle que la destination reste accessible à la source après quelques pauses au cas où le chemin est segmenté.

Le prédicat de chemin $R_{u,v}(c)$ *est vrai si et seulement si* $\exists n \in \mathbb{N}*$, $R^n_{u,v}(c)$ *est vrai.*

Par conséquence, $R_{u,v,n}$ est vrai quand il existe une route de longueur n entre u et v. Les spécifications du *problème du maintien de chemin* est défini alors comme suivant :

Definition 14 *Les* spécifications du problème de maintien de chemin *sont définies par : pour toute exécution* $e = c_1, c_2, \ldots$ *du système distribué* $\mathcal{S}$ *partant d'une configuration initiale* $c_1 \in \mathcal{C}$ *et qui satisfait* $R^1_{source,puits}(c)$*, alors toute configuration* $c_i \in e$ *satisfait* $R_{source,puits}(c_i)$*.*

À noter que, au cas où le système distribué n'est pas dynamique ($G^1_{|c_1} = G^p_{|c_1}$ pour tout $p \in \mathbb{N}^*$), le chemin a toujours une longueur égale à 1. Le problème de maintien de chemin est important dans le cadre des réseaux dynamiques, et ce problème est résolu trivialement dans le cadre des réseaux non dynamiques.

5.9.3 Spécifications *best effort* appliquées à l'algorithme de maintien de chemin

La mise à jour locale du chemin, en ajoutant ou retirant des nœuds, nécessite un échange de quelques messages dans le voisinage. L'algorithme, quel que soit son principe, n'atteindra pas son but quand ce voisinage n'est pas assez stable. Nous introduisons alors les spécifications *best effort* appliquées au cas du problème de maintien de chemin, et cela en introduisant une propriété de continuité, et une autre topologique, liées à ce problème.

La propriété de continuité, notée P_C, annonce que le chemin entre la source et le puits peut être segmenté en plusieurs sous-chemins dans des configurations successives, nécessitant quelques *pauses* sur les nœuds intermédiaires affectés par les mises à jour locales (illustration figure 5.4). Ces pauses signifient que la prochaine destination du prochain saut sera disponible dans une configuration proche et accessible. Cette définition se rapproche du concept de voyage ou *journey* en restant tout de même différent dans la réalisation [22].

Dans le cas de l'algorithme de maintien de chemin, la propriété topologique est définie par l'existence de graphes p-dynamiques, où p correspond au nombre de messages successifs que l'algorithme a besoin d'échanger avant que les liens ne se cassent. La propriété de continuité est définie par l'existence d'un chemin de la source vers la destination dans toutes les configurations de l'exécution.

Même si cette approche est appropriée à notre exemple dans ce manuscrit, elle reste générale et applicable sur d'autres problèmes et d'autres algorithmes.

L'algorithme de maintien de chemin que nous proposons vient résoudre le problème de déconnexion et de cassures de liens pour une communication *unicast* dans un réseau mobile. Nous appellerons ce problème le *problème de maintien de chemin.*

Definition 15 *Soit $e = c_0, c_1, c_2, \ldots$ une exécution du système distribué $\mathcal{S}$. La* propriété de continuité du problème de maintien de chemin*, notée P_C, est vraie et seulement vraie si, pour toute configuration $c_i \in e$, il existe un entier $l \in \mathbb{N}$, l sommets distincts $v_1, \ldots, v_l$, et l entiers $k_1, k_2, \ldots, k_l$ tels que :*
$R_{source,v_1}(c_i), R_{v_1,v_2}(c_{i+k_1}), R_{v_2,v_3}(c_{i+k_1+k_2}), \ldots, R_{v_{l-1},v_l}(c_{i+k_1+\ldots+k_{l-1}}), R_{v_l,puits}(c_{i+k_1+\ldots+k_l})$
sont vrais.

L'entier l présente le nombre de pauses qu'un message émis par le nœud source à une configuration c_i doit rencontrer avant d'atteindre le nœud puits, en suivant le chemin formé. Ces pauses apparaissent à chaque nœud v_j, atteint à une configuration $c_{i+k_1+k_2+\ldots+k_{j-1}}$, jusqu'à ce que les variables locales autour de v_j soient mises à jour. Cette mise à jour est faite dans k_j configurations et le message peut alors quitter le nœud v_j dans la configuration $c_{i+k_1+k_2+\ldots+k_j}$. À noter que, comme le problème de maintien de chemin est dédié aux réseaux dynamiques, ces mises à jour locales sont inévitables.

Une pause est nécessaire à chaque fois qu'une arête (u, v) disparait entre deux nœuds successifs dans le chemin. Cette arête disparue ne peut être réparée que par l'insertion d'un nœud relai w de sorte que l'arête (u, v) soit remplacé par deux arêtes (u, w) et (w, v). Au cas où l'arête (u, v) disparait à la configuration c, elle n'existe plus dans le graphe $G^1_{|c}$ alors qu'elle existait dans la topologie précédente $G^1_{c|}$. Les arêtes (u, w) et (w, v) doivent exister assez longtemps afin de permettre la mise à jour locale des variables sur u, v et w. Au cas où p messages sont requis pour cette mise à jour, les arêtes (u, w) et (w, v) doivent appartenir à $G^p_{|c}$. Cette condition exige la propriété topologique suivante (illustrée dans la figure 5.5) :

Definition 16 *Soit $e = c_0, c_1, c_2, \ldots$ une exécution du système distribué $\mathcal{S}$. La* propriété topologique du problème de maintien de chemin*, noté P^p_T est vraie si et seulement si,*

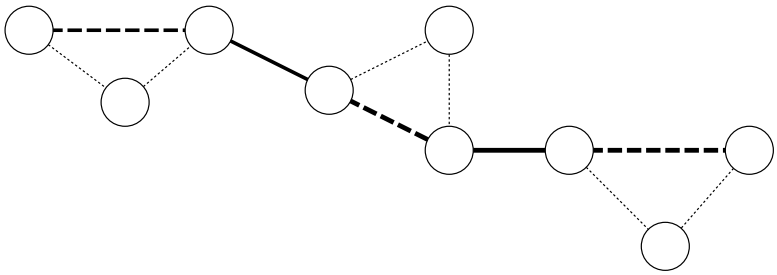


FIGURE 5.5 – La propriété topologique est définie par l'existence d'un nœud voisin pour tout lien qui se casse.

pour toute configuration $c \in e$, l'implication suivante est satisfaite :
$(u,v) \in G^1_{c|} \wedge \neg(u,v) \in G^1_{|c} \Rightarrow \exists w, (u,w) \in G^p_{|c} \wedge (w,v) \in G^p_{|c}$.

En finale, la propriété topologique et celle de continuité définissent les spécifications *best effort* pour le problème de maintien de chemin : si une configuration c du système distribué $\mathcal{S}$ satisfait $P_T(c)$, alors toute configuration c satisfait $P_C(c)$. Cela veut dire que si la dynamique du réseau permet de réparer le chemin localement, le message est alors toujours capable d'atteindre le puits à partir de la source. Cette réparation n'est affectée que par les pauses sur les nœuds, qui représentent les délais nécessaires pour l'échange de p messages, afin de mettre à jour le chemin.

5.9.4 Propriétés du problème de maintien de chemin

Pour toute exécution e du système distribué $\mathcal{S}$, il existe un entier p maximal tel que P^p_T est satisfaite. Nous introduisons alors la définition suivante :

Definition 17 *Soit $e = c_0, c_1, \ldots$ une exécution du système distribué $\mathcal{S}$; $p_e \in \mathbb{N}$ est défini par : $\forall c \in e, P^{p_e}_T(c) \wedge \neg P^{p_e+1}_T(c)$.*

Par conséquence, tout algorithme satisfaisant les spécifications *best effort* du maintien de chemin sur une exécution e, doit être capable d'insérer un nœud relai en utilisant moins de p_e échanges de messages dans le voisinage. Réciproquement, étant donné un algorithme de maintien de chemin nécessitant p messages pour insérer un nœud, il lui sera impossible de satisfaire ses spécifications *best effort* lors d'une exécution e où $p_e < p$. En effet, le nœud relai w ne peut être inséré avant qu'il ne soit voisin de u et de v, que l'arête (u,v) disparait, et que u et w, ainsi que w et v, n'aient eu le temps d'échanger

les messages requis à la mise à jour locale. Le cas échéant, la reconstruction de la route est impossible avec les messages locaux car la distance entre u et v sera supérieure à 1 dans tous les graphes suivants.

Jusqu'à maintenant, nous n'avons considéré que l'extension du chemin, prenant en compte l'insertion des nœuds relais afin d'assurer la continuité. Cependant, il est intéressant que l'algorithme de maintien de chemin parvienne à maintenir ce chemin aussi court que possible, et cela en court-circuitant des nœuds quand possible. Effectivement, la source et le puits peuvent se rapprocher après éloignement. Ce même phénomène peut se produire pour d'autres nœuds du chemin également. Il est important de noter que la réduction ne peut impliquer que les nœuds du chemin ou leurs voisins. D'autres types de réductions nécessitent des informations non locales.

La réduction du chemin nécessite l'échange de quelques messages de contrôle. Le nombre de ces messages de contrôle augmente avec la longueur de la réduction. Ceci dit, la réalisation d'une longue réduction nécessite une certaine stabilité du réseau. Par conséquence, une mesure de l'efficacité de l'algorithme vis à vis de la longueur du chemin peut être reflétée par la longueur maximale de la réduction possible pour un p_e donné (réduction impliquant les nœuds du chemin et leurs voisins).

Preuve

Les deux mécanismes d'adaptation, à savoir l'extension et la réduction du chemin, ne comptent que sur les échanges locaux de messages. L'extension du chemin prend en charge la gestion de la dynamique du réseau, en insérant un nœud v relai entre les nœuds u_i et u_j (voir ci-dessus).

Le nœud voisin v doit être présent assez longtemps pour permettre les mises à jour des variables locales. Il utilise un seuil pour éviter de recommencer trop souvent, dès qu'un message est perdu (ce qui conduit à l'insertion d'un drapeau d'incertitude ? dans les messages). Au début, seuls trois messages sont requises par chaque nœud u_i, v et u_j. Le nœud u_i envoie successivement des messages contenant $u_i > u_j?$, $u_i > vu_j?$ et enfin $u_i > vu_j$. Le nœud v envoie successivement des messages contenant $u_i?v > u_j?$, $u_iv > u_j?$, et enfin $u_iv > u_j$. Le nœud u_j envoie successivement des messages contenant $u_i?u_j >$, $u_i?u_j >$ et enfin $u_ivu_j >$. Étant donné que ces messages sont envoyés successivement, les arêtes (u_i, v) et (v, u_j) doivent exister dans $G^9_{|c}$ si l'arête (u_i, u_j) appartient à $G^1_{c|}$ et non dans $G^1_{|c}$, ce qui signifie que (u_i, u_j) disparait dans la configuration c.

Au cas où il y a des pertes de messages sur un lien (due à une collision par exemple), un message doit être répété plusieurs fois pour assurer sa réception. Soit l le nombre maximal de pertes successives d'un message. Alors les messages doivent être envoyés $l + 1$ fois pour assurer leur réception. Dans certains cas, les arêtes (u_i, v) et (v, u_j) doivent alors exister dans $G^{9(l+1)}_{|c}$.

Proposition 18 *L'algorithme PTH satisfait les spécifications* best effort *du maintien de chemin avec* $p_e = 9 \times (l + 1)$ *où* l *est le nombre maximal de pertes successives de messages sur un lien.*

Par construction, le mécanisme de réduction de chemin ne peut pas découper le chemin. Il assure la progression des réductions parce que le *mutex* sur la partie du chemin partagée par les réductions chevauchées est verrouillé par son premier nœud (proche de la source). Une réduction entre u_i et u_j par un voisin nœud v nécessite $6 + s$ messages successifs, où s est la taille de la réduction (distance entre u_i et u_j dans le chemin). Si $p_e = 9 \times (l + 1)$, les réductions ayant une taille maximum $s = 3$ sont réalisées.

5.10 Conclusion

Tout algorithme peut échouer si la dynamique augmente. Notre algorithme de maintien de chemin a été construit pour mieux supporter la dynamique que les routages classiques.

Dans ce chapitre, nous avons proposé une modélisation, avec l'approche *best effort* et les familles de graphes p-dynamiques, qui permet de caractériser la dynamique maximale tolérée par notre algorithme.

Chapitre 6

Maintien de chemin, le protocole

Au chapitre 4, nous avons présenté un nouvel algorithme pour maintenir le chemin entre deux entités. C'est une alternative aux routages *unicast* classiques pour les réseaux dynamiques. Dans le chapitre 5, nous avons établi formellement les limites supportées par cet algorithme en terme de dynamique.

Dans ce chapitre, nous présentons notre travail permettant d'obtenir un protocole exploitable en pratique à partir de cet algorithme. Nous présentons ensuite les tests sur route qui fournissent une preuve de concept de sa faisabilité ainsi qu'une étude comparative de ses performances.

6.1 Rappel du principe de l'algorithme de maintien de chemin

Le maintien de chemin suppose l'existence d'une communication entre deux entités proches dans le même voisinage, comme le montre la figure 6.1. Il intervient lorsque la communication est interrompue. Dans ce cas, il implique uniquement les nœuds voisins aux deux entités concernées (voisins du lien cassé). Quand la communication est cassée, les nœuds intermédiaires se proposent ; ceci est illustré dans la figure 6.1. Le choix des nœuds relais est fait par les deux parties de la communication. Il est validé d'abord par l'extrémité proche de la source (amont de l'extension), puis accepté par l'extrémité proche de la destination (aval de l'extension). Cet algorithme repose sur un échange continu de messages entre les entités du chemin, à partir du moment où la communication est déclenchée, et réagit aux changements de topologie dans le chemin en gérant l'intégration et la suppression des nœuds intermédiaires du chemin.

L'algorithme de maintien de chemin gère les propositions multiples de la part des voisins. En effet, le nœud relai est choisi par le nœud en amont de l'extension (plus près de la source), et puis accepté par le nœud en aval (plus proche de la destination). Ce mécanisme évite les conflits quand plusieurs voisins se proposent comme relais. De même, la proposition multiple de réductions est gérée par l'algorithme de maintien de chemin (cf. figure 6.2). En fait, une seule réduction peut être effectuée sur un tronçon du chemin. Ce mécanisme, illustré sur la figure 6.3, est expliqué ultérieurement.

La connaissance locale du chemin se fait en définissant deux variables locales "prédécesseur" et "successeur" qui indiquent la position du nœud local dans le chemin par rapport aux autres nœuds. Il se positionne alors par rapport à un nœud qui le précède qu'il définit comme prédécesseur, et un autre qui le suit et qu'il définit comme successeur.

Le principe de fonctionnement de cet algorithme est illustré dans les figures 6.4 et 6.5. Une communication existe entre deux entités proches dans le même voisinage, comme le montre la figure 6.4. L'algorithme de maintien de chemin intervient alors lorsque la communication est interrompue. Dans ce cas, il implique uniquement les nœuds voisins aux deux entités concernées (nœuds 3, 4 et 5). Quand la communication est cassée, les nœuds intermédiaires se proposent ; ceci est illustré dans la figure 6.5 où le nœud 3 se propose. Il devient alors relai entre le nœud 1 et le nœud 2.

Le fonctionnement de l'algorithme est décrit en détails dans la suite du chapitre.

6.2 Notations

Dans cette section, nous allons expliciter les notations utilisées pour décrire le fonctionnement de l'algorithme.

Les différents nœuds échangeront des messages pour maintenir le chemin. Ces messages échangés comprennent un champs appelé "chemin" (*path*). Ce chemin contient une information sur la suite des nœuds composant le chemin, et une information sur la réduction au cas où une telle réduction existe. On notera ce champ P. La partie représentant la réduction est notée R_p.

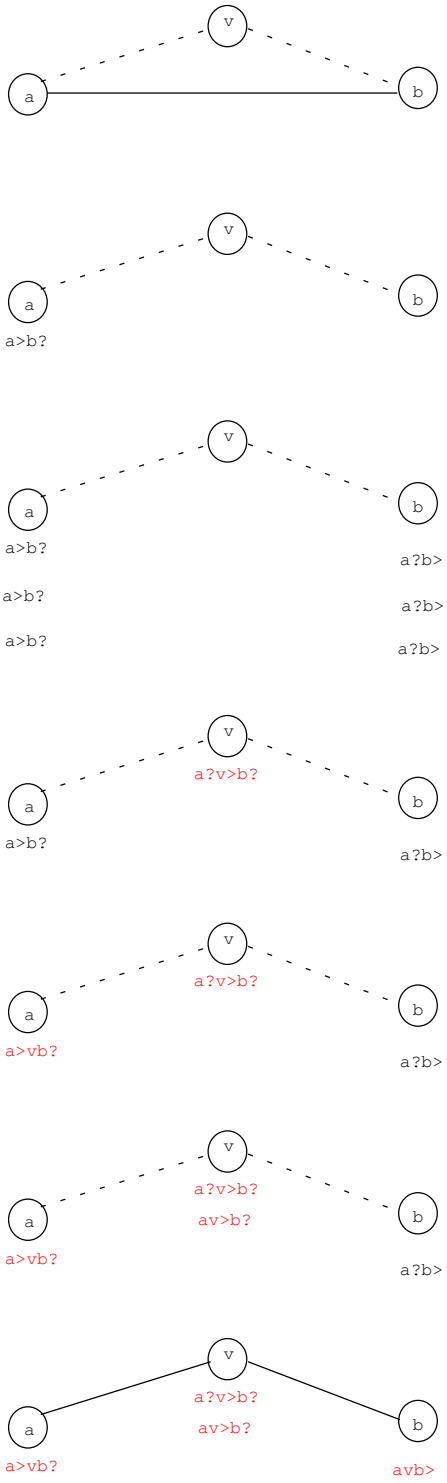


FIGURE 6.1 – Extension du chemin : après la disparition du lien, les deux extrémités envoient des messages indiquant un problème au niveau de l'autre extrémité respectivement. Le nœud voisin du lien cassé se propose pour être relai de la communication. Il est d'abord validé par le nœud en amont du lien cassé puis accepté par l'autre extrémité.

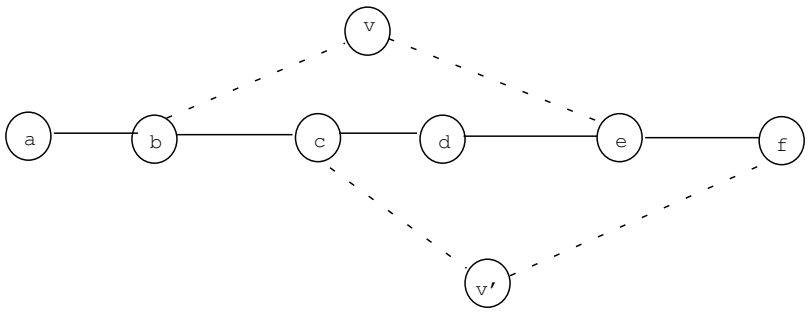


FIGURE 6.2 – Conflit entre deux réductions : quand il existe deux réductions potentielles, et si il n'existe pas un mécanisme de gestion de conflit de réduction, le chemin est partitionné en sous-chemins distincts et le maintien de la communication est alors impossible.

Le chemin P aura le format suivant :
$P = src_p$ ... pre_p $snd_p > rcv_p$ ... dst_p,
où les différents éléments sont définis comme la suite :
- src_p : source des messages dans le chemin P
- pre_p : prédécesseur du nœud émetteur du message dans P
- snd_p : émetteur du message comportant le chemin P traité
- rcv_p : récepteur du message comportant le chemin P traité
- dst_p : destination finale du message comprenant ce chemin, et destination de tous les messages qui vont suivre sur ce chemin

On rajoute à ces définitions deux autres qui seront utiles à la compréhension du traitement sur les messages :
- Prefixe(p) = $src_p \dots pre_p$: tous les nœuds du chemin qui précèdent le nœud émetteur.
- Suffixe(p) = $rcv_p \dots dst_p$: tous les nœuds du chemin qui suivent le récepteur.

Le symbole $>$ est choisi comme délimiteur dans le chemin pour séparer le préfixe du suffixe, et positionner l'émetteur et le récepteur des messages.

Sur un nœud donné x, le prédécesseur et le successeur seront notés $pred_x$ et $succ_x$ respectivement. Une réduction sera représentée par up_p-sc_p-dw_p où up_p décrit le nœud en amont de la réduction, sc_p le nœud réducteur (qui propose de réduire le chemin) et dw_p le nœud en aval de la réduction. Dans ce cas, up_p (pour *upstream*) et dw_p (pour *downstream*) sont deux voisins de sc_p. Un segment de réduction est alors défini, et il regroupe tous les nœuds situés entre up_p et dw_p.

Un problème au niveau du prédécesseur (respectivement successeur) est signalé sur un nœud après l'expiration d'un *timer* dédié et nommé $timer_{pred}$ (respectivement $timer_{succ}$). Ce *timer* est armé à chaque réception de la part du prédécesseur (respectivement successeur), et expire après un certain délai en cas de non réception de la part de ce prédécesseur (respectivement successeur).

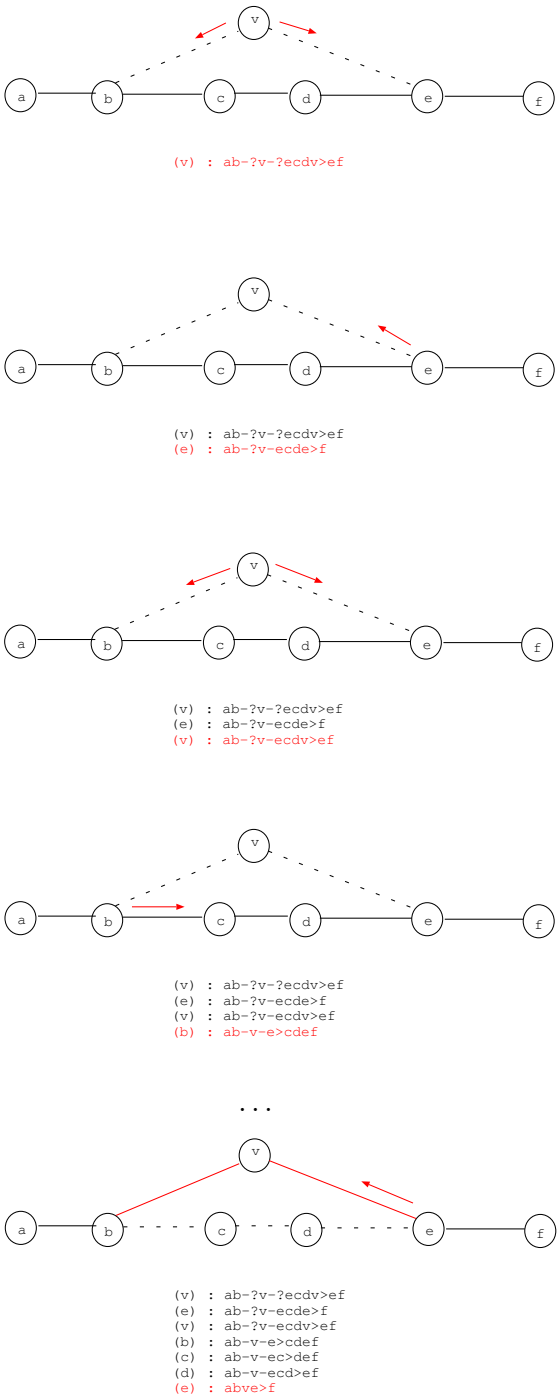


FIGURE 6.3 – Réduction du chemin : un nœud voisin qui propose la réduction attend l'acceptation par le nœud aval pour la retransmette au nœud amont de la réduction. Celui ci, s'il n'admet pas de réduction en cours, retransmet sa validation à son successeur dans le chemin. Cette validation est transmise sur tout le segment de réduction (si il n'existe pas de réduction en cours sur un des nœuds de ce segment). A la réception de la validation, le nœud aval effectue la réduction.

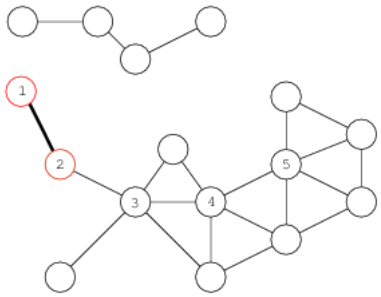


FIGURE 6.4 – Un réseau où la source et la destination sont à portée l'une de l'autre.

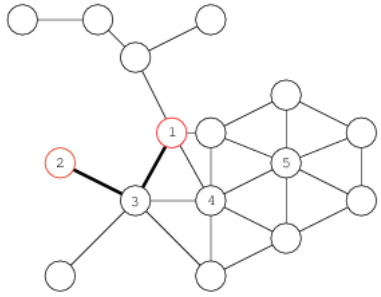


FIGURE 6.5 – Un réseau où la source et la destination se sont éloignées. L'intervention des nœuds voisins est alors nécessaire pour le maintien de la communication.

6.3 Lancement de l'application au temps $t = t_0$

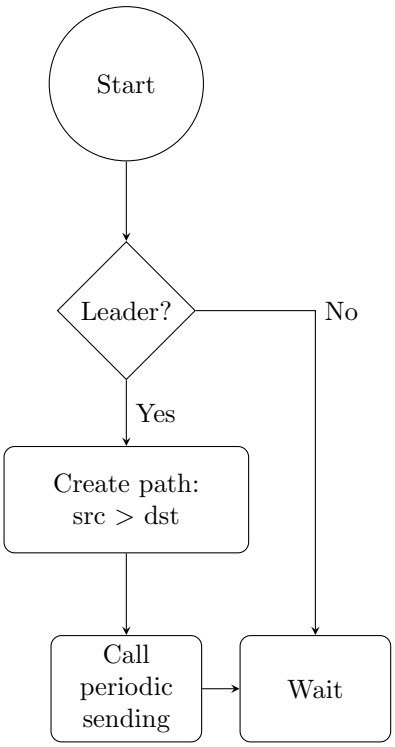


FIGURE 6.6 – Au lancement de l'application, et si le nœud est "leader", il forme un chemin vers la destination et commence son envoi périodique.

Lors du lancement des applications, le nœud source, ayant des messages à transmettre à son voisin, entame une communication avec lui en construisant tout d'abord un chemin vers ce voisin. Le chemin est de la forme : $id_{local} > id_{voisin}$ (figure 6.6).

La création du chemin est suivie par l'appel de la fonction périodique qui a pour rôle d'envoyer à chaque expiration du *timer* local, un message dans le voisinage.

Seul un nœud du réseau peut se définir comme source des messages et définir une destination. Ce nœud est reconnu par une option ajoutée qui le définit comme "source".

Un test se fait sur toutes les entités PTH du réseau lors du lancement de l'application. Un seul véhicule satisfera les conditions du test (*leader*), et enverra alors des messages.

6.4 Envoi périodique

L'envoi périodique est le principe sur lequel repose la mise à jour du chemin sur les nœuds de ce chemin. Chaque nœud du chemin envoie périodiquement dans son voisinage un message comportant essentiellement la liste de ses prédécesseurs et celle de ses

successeurs dans le chemin, et éventuellement un champs utile (*payload*) au cas où une application source lui fournit des données à transmettre. Ces données peuvent provenir d'un capteur comme données brutes, ou à la suite d'un post-traitement ou calcul. Nous dénotons cette application comme application source. En cas d'absence d'application, l'algorithme de maintien envoie des messages avec un champs utile vide.

Un nœud qui n'appartient pas à la liste transmise par son voisin, et plus exactement, s'il n'est pas le nœud destinataire du message, n'a pas le droit de retransmettre ce message dans le voisinage. Ce mécanisme évite tout *broadcast* global au niveau du réseau, et réduit considérablement le nombre de nœuds impliqués dans ces émissions.

6.5 Réception d'un message

Tous les nœuds du réseau sont susceptibles de recevoir un message envoyé sur leurs liens, au cas où ils admettent un lien direct avec le nœud émetteur.

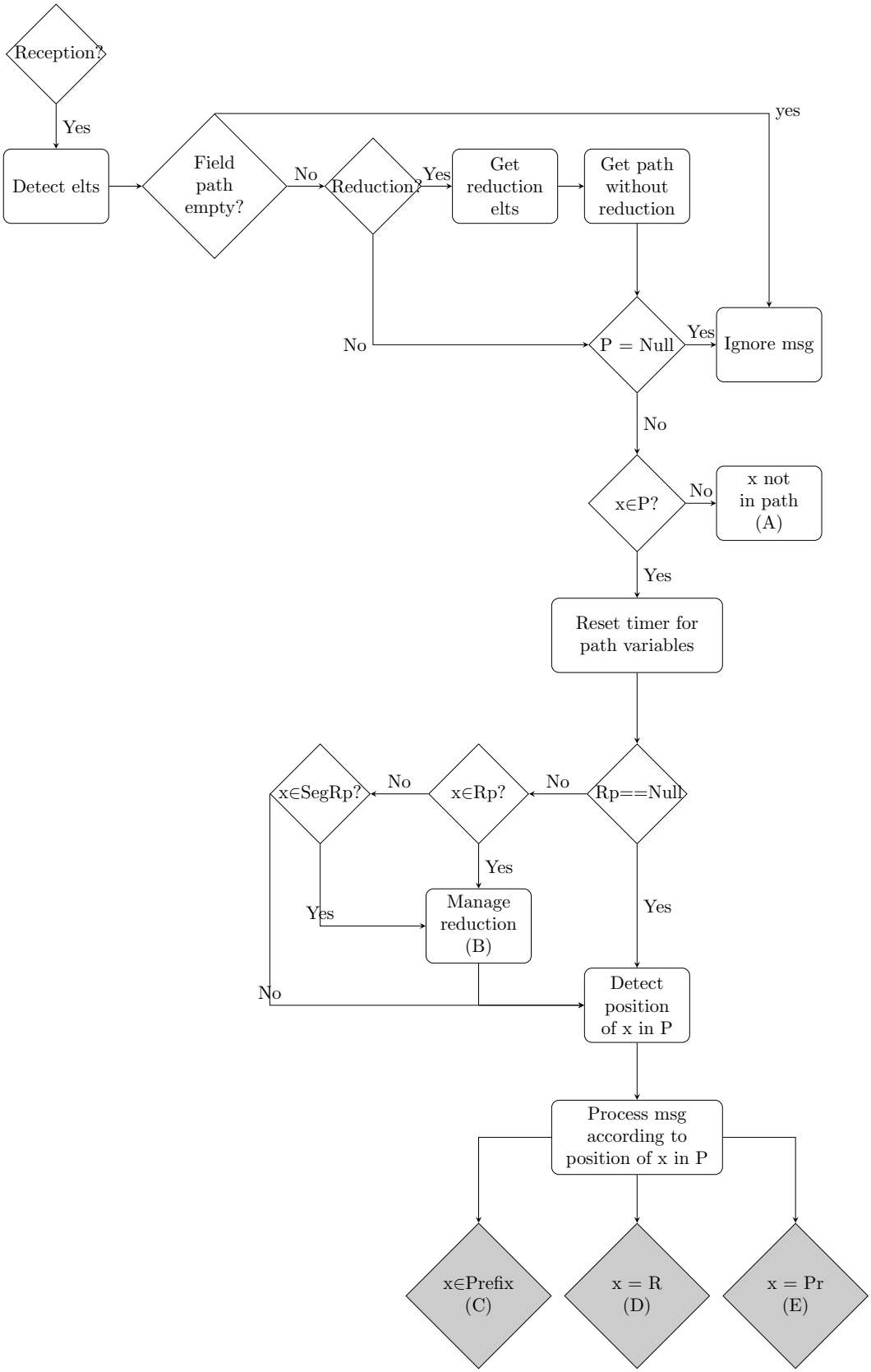


FIGURE 6.7 – A la réception d'un message, le noeud vérifie s'il existe une réduction, détermine sa position et continue le traitement suivant le cas détecté.

Le fonctionnement à la réception d'un message est illustré dans la figure 6.7.

Quand un message est reçu, les différents champs sont détectés (émetteur, destination, chemin, etc.). Si le champs "chemin complet" est vide, le message est ignoré. Sinon, le nœud gère le message de la manière suivante : si le chemin comprend une réduction, cette réduction sera enlevée du chemin pour la suite du traitement, sinon, le traitement peut avoir lieu. Donc à ce niveau, la réduction, si elle existe dans le chemin complet, est omise. Nous distinguons alors deux champs "chemin" et "réduction" à partir du chemin complet brut reçu. Au cas où le champs chemin sans réduction (après omission de la réduction du champs "chemin") est vide, le message est ignoré.

Ensuite, le nœud vérifie s'il fait partie du chemin. Si c'est le cas, il met à zéro toutes ses variables locales liées au chemin. Dans le cas contraire, le traitement est décrit dans la section 6.5.2.

Une fois que le nœud a vérifié sa présence dans le chemin, il vérifie si le chemin comprenait une réduction. Si le message n'admet pas de réduction, le nœud se positionne alors dans le chemin en détectant sa position par rapport à l'émetteur. Un traitement différent est appliqué, suivant qu'il fait partie du préfixe (section 6.5.3), qu'il est le nœud récepteur du message (section 6.5.4), ou bien s'il est le prédécesseur du nœud émetteur (section 6.5.5). Ces traitements sont détaillés dans les sous-sections suivantes.

Si le chemin admet une réduction, autrement dit que le champs "réduction" n'est pas vide, un traitement supplémentaire est nécessaire avant de se situer dans le chemin. Au cas où le nœud fait partie de la réduction, il va gérer ce message en adaptant le retour et en ajustant la réduction (plus de détails dans la section 6.5.1). Cette gestion est également nécessaire au cas où le nœud fait partie du segment de réduction. À noter que le segment de réduction comprend tous les nœuds du chemin situés entre les deux extrémités de la réduction. Si le nœud en question ne fait pas partie de la réduction (réducteur ou extrémité de segment de réduction), et n'appartient pas au segment de réduction, il procède directement à la phase de positionnement dans le chemin.

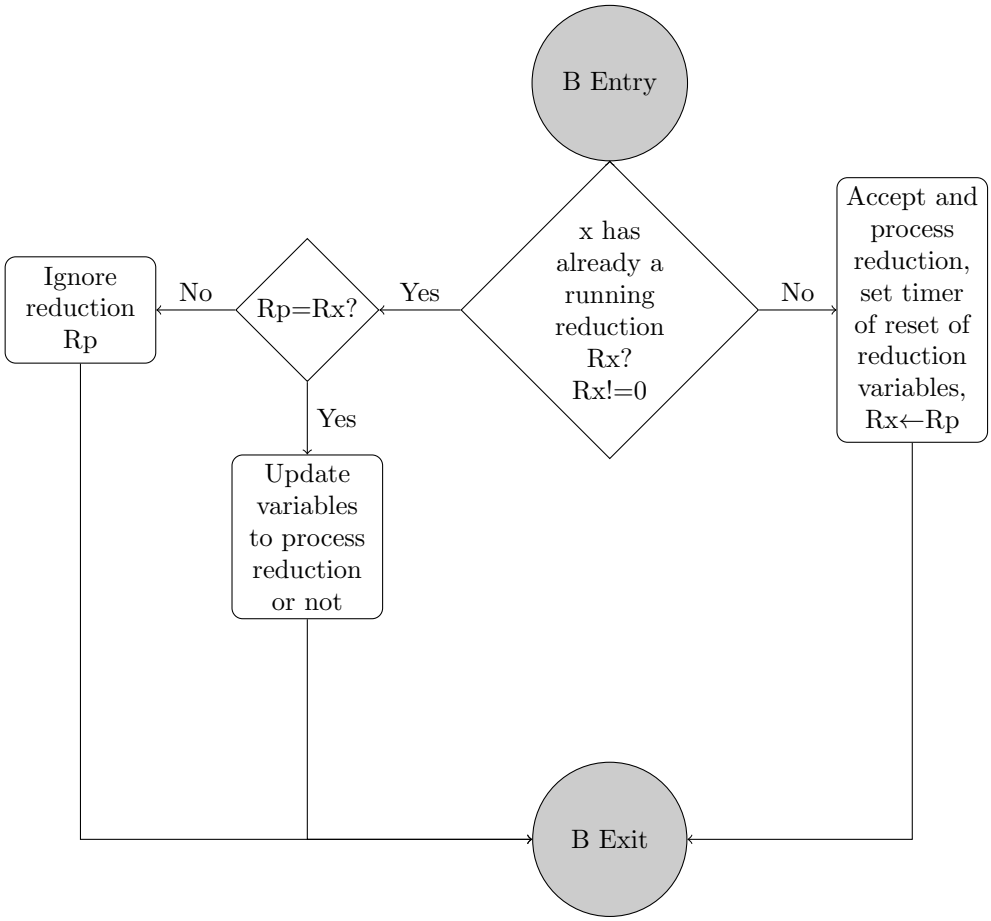


FIGURE 6.8 – Le nœud vérifie s'il existe déjà une réduction en cours de traitement. Si c'est le cas, il va rejeter la nouvelle proposition de réduction. Sinon, il accepte de traiter la réduction reçue.

6.5.1 La gestion de la réduction

Cette gestion (illustrée dans la figure 6.8) est appelée à la réception d'un message quand le nœud s'aperçoit qu'il fait partie de la réduction présente dans le message, ou bien qu'il est dans le segment de cette réduction. Le nœud vérifie d'abord qu'il n'admet pas une réduction en cours. S'il n'admet pas de réduction en cours, il accepte alors la proposition de réduction et arme un *timer* de remise à zéro des variables de réduction.

Au cas où le nœud admet déjà une réduction en cours, il vérifie que celle-ci est identique à celle reçue. Si c'est le cas, il met à jour ses variables de réduction. Sinon, il ignore la réduction reçue.

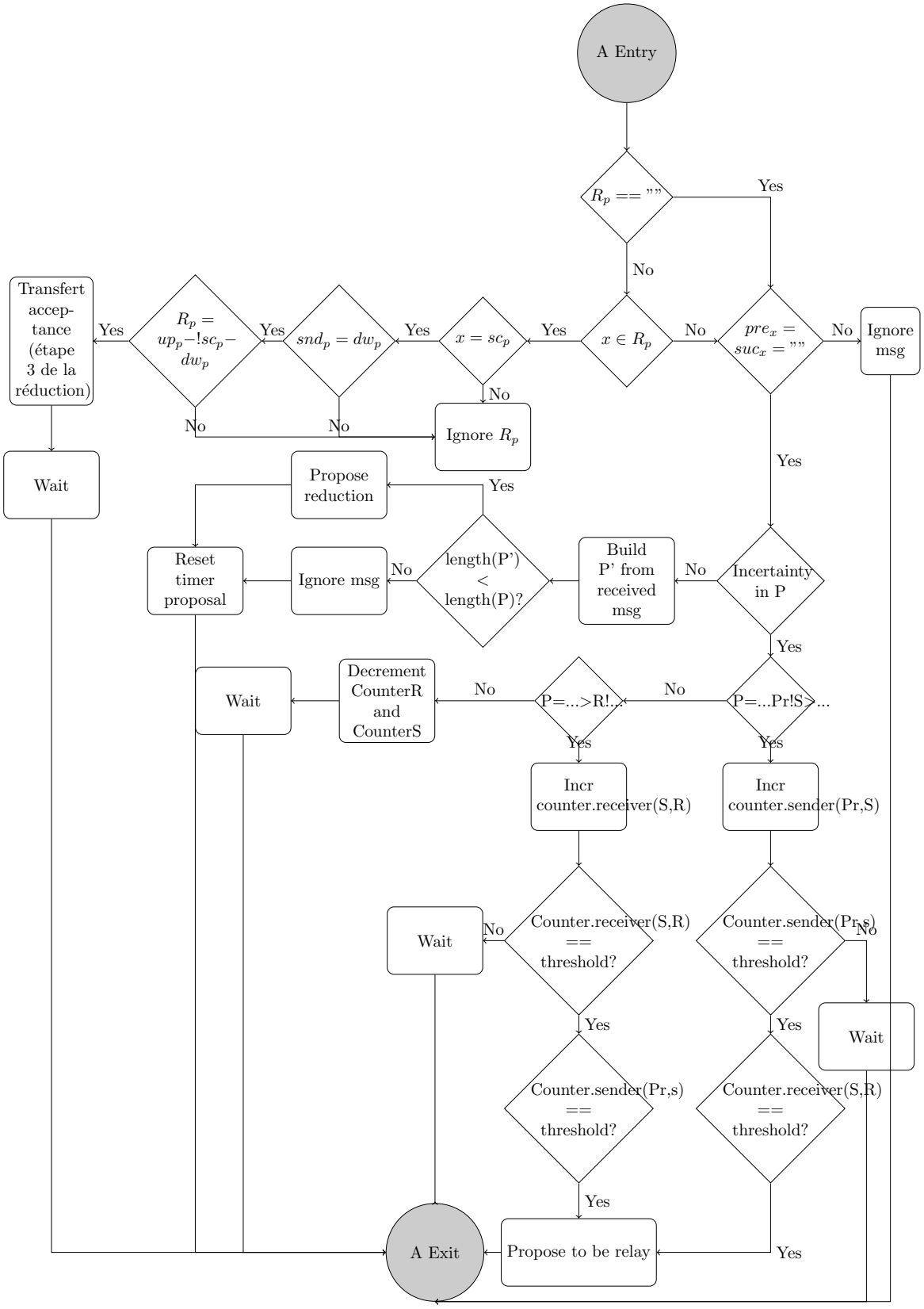


FIGURE 6.9 – Le nœud, faisant pas partie du chemin, vérifie s'il existe une incertitude dans le message reçu et s'il peut se proposer comme un relai. S'il n'y a pas d'incertitude, il vérifie s'il peut se proposer pour réduire le chemin si possible.

6.5.2 Le nœud ne fait pas partie du chemin

Au cas où le nœud ne fait pas partie du chemin (illustré dans la figure 6.9), il doit observer s'il est possible de se proposer comme relai pour un lien cassé, ou s'il est possible de proposer une réduction locale du chemin.

Le nœud vérifie tout d'abord si le chemin complet reçu comporte une réduction. Si le chemin ne porte pas de réduction, donc pas de réduction à traiter, et que les variables "pred" et "succ" locales sont nulles, le nœud peut alors vérifier s'il y a un soucis sur le lien. Le nœud vérifie donc s'il y a une incertitude dans le chemin. Si c'est le cas, il vérifie que cette incertitude provient bien des deux extrémités du lien auquel le nœud est voisin. Si l'incertitude provient de ces deux extrémités, le nœud vérifie qu'il a reçu assez de fois les mêmes incertitudes, et se propose alors pour être relai de la communication.

Le fait qu'il attende le seuil sert à assurer que le problème au niveau du lien n'est pas seulement dû à une éventuelle perte d'un message, mais provient bien d'un problème de déconnexion des deux extrémités.

Si le message ne comporte ni réduction ni incertitude, le nœud construit le chemin depuis le message reçu. Si le chemin formé est inférieur au chemin reçu, il remarque qu'il peut proposer une réduction, et envoie un message proposant cette réduction. Si aucune réduction ne peut être proposée, le message est ignoré.

Un nœud ne peut pas traiter une réduction et une extension à la fois. Il vérifie d'abord l'existence d'une réduction dans le chemin complet reçu. Si le message ne comporte pas de réduction, le nœud traite l'extension du chemin.

Dans le cas où une réduction existe dans le chemin complet reçu, le nœud (n'appartenant pas au chemin) procède aux vérifications successives suivantes :

- il fait partie de la réduction (il est l'un des trois nœuds de la réduction $up_p - sc_p - dw_p$)
- il est le nœud réducteur (sc_p)
- le nœud émetteur est le nœud aval de la réduction (le nœud émetteur est dw_p)
- la réduction ne comporte pas d'incertitude sur le lien nœud réducteur-nœud en aval (le lien $sc_p - dw_p$ est sans incertitude, donc sc_p est accepté par dw_p).

À la suite de toutes ces vérifications, et si les conditions sont satisfaites, le nœud peut transmettre l'acceptation du nœud en aval. Cette acceptation constitue la troisième étape de la réduction. Dans les cas échéants, si l'une des conditions citées n'est pas satisfaite, le message est ignoré.

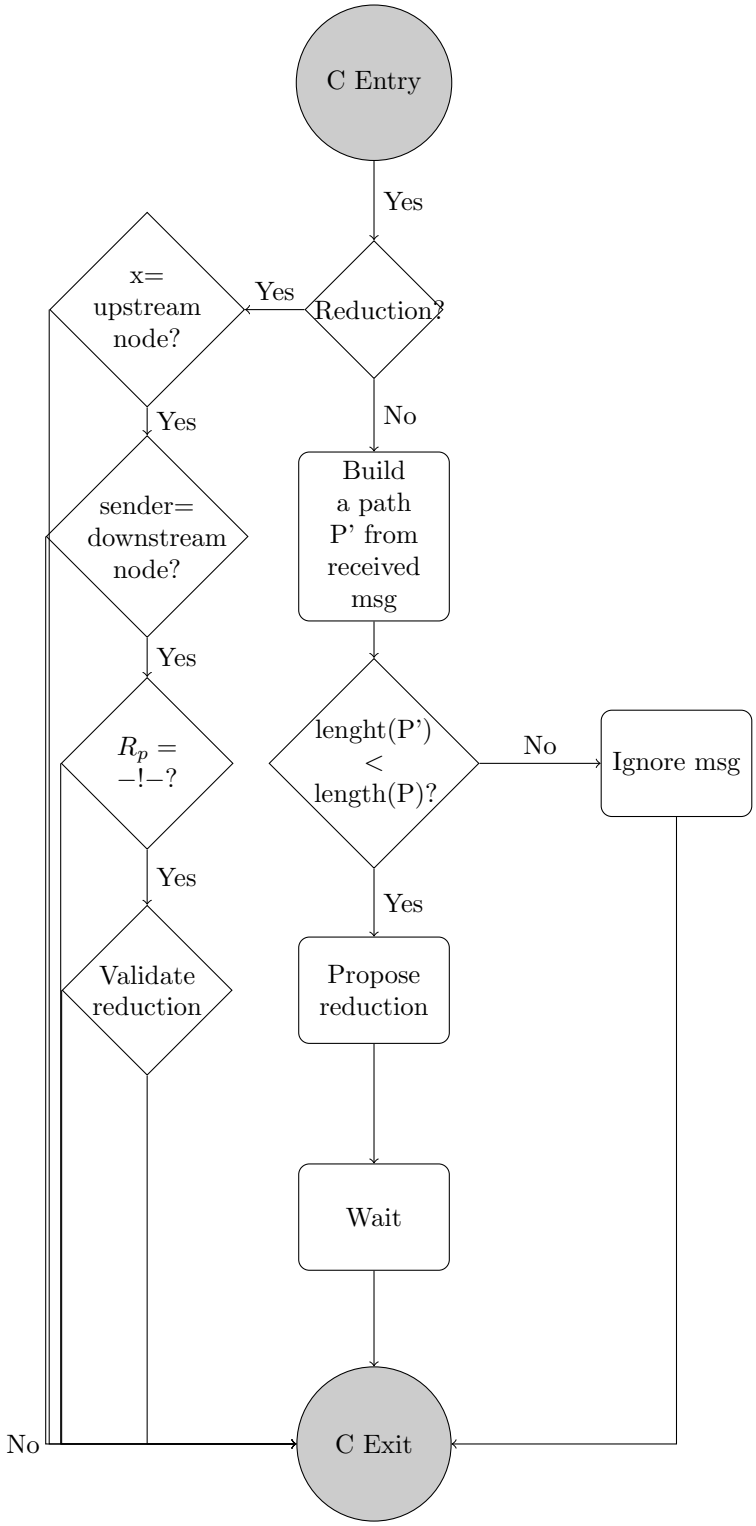


FIGURE 6.10 – Le nœud, appartenant au préfixe du chemin, vérifie s'il existe une réduction dans le message reçu et détermine sa validité. Sinon, il teste s'il peut proposer lui même une réduction. Dans les autres cas, il ignore le message.

6.5.3 Le nœud appartient au préfixe du chemin

Quand le nœud reçoit un chemin dans lequel il fait partie du préfixe (illustré dans la figure 6.10), autrement dit, il reçoit un message d'un nœud qui n'est pas son successeur dans le chemin, deux cas sont alors possibles. Soit il reçoit une demande de réduction, soit il remarque qu'il peut réduire le chemin reçu.

Il détecte d'abord si le chemin reçu comprend une réduction. Si c'est le cas, il vérifie s'il est le nœud en amont de la réduction (nœud local = up_p), et si le nœud émetteur est le nœud en aval de la réduction ($sender = dw_p$). Si ces deux conditions sont vérifiées, et qu'en plus la réduction admet une incertitude sur le lien nœud amont-nœud réducteur uniquement (la réduction a la forme $up_p - !sc_p - dw_p$), la réduction peut être alors validée. Le nœud envoie donc un message de validation de réduction.

Si le message reçu n'admet pas de réduction, le nœud construit un chemin à partir de celui reçu en tentative de réduction. Si ce chemin formé est plus court que celui reçu, il propose alors la réduction. Sinon, le message est ignoré.

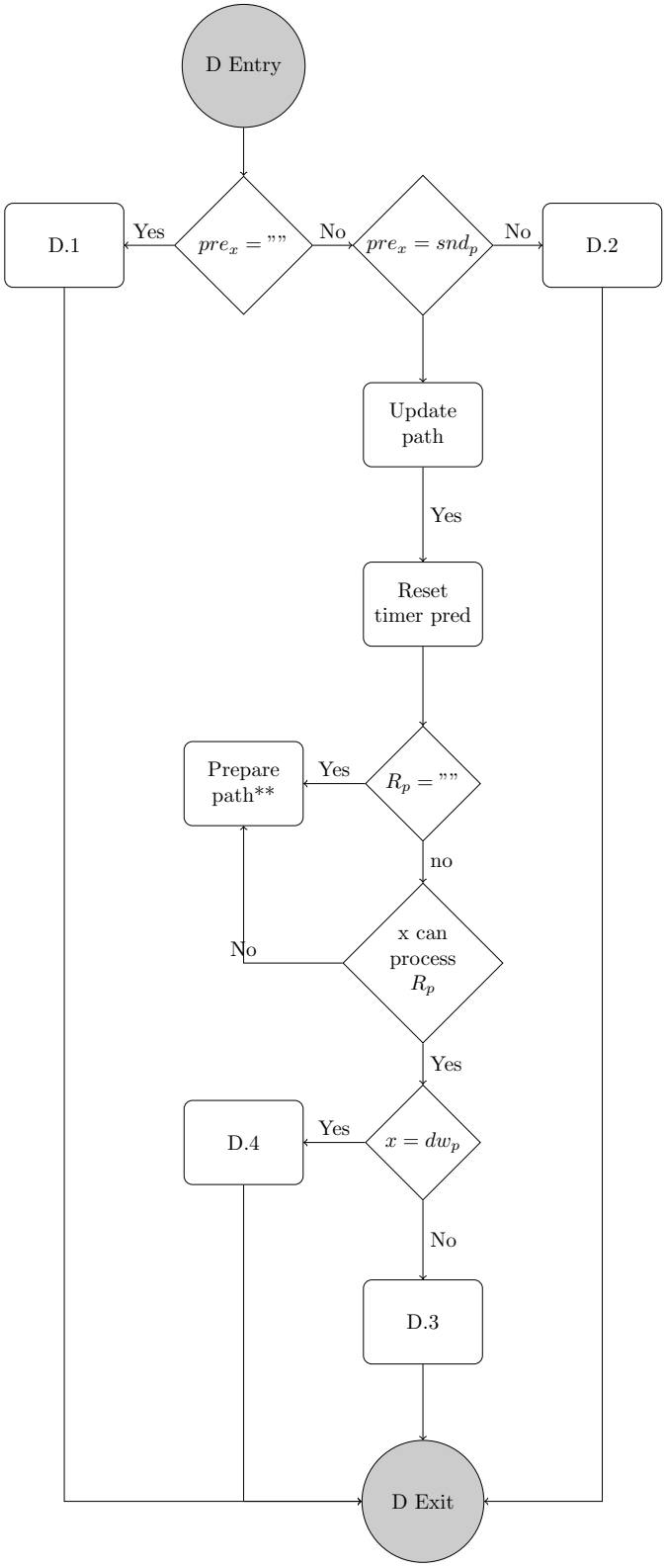


FIGURE 6.11 – Détails du block D de la figure 6.7 : le nœud est récepteur d'un message. Dans ce cas, il met à jour le chemin, et vérifie ensuite si le message contient une réduction et s'il est capable de la traiter.

6.5.4 Le nœud est destinataire du message

À la réception d'un message, le nœud teste s'il est destinataire du message (ce cas est illustré dans la figure 6.11). À la réception, le nœud peut admettre deux cas, un premier où il admet déjà un prédécesseur, autrement dit la variable "pred" définit un nœud émetteur, et un autre où cette variable est nulle.

Même si le nœud admet déjà un prédécesseur, il se peut qu'il reçoive un message d'un nœud qui n'est pas son prédécesseur. Nous allons détailler à la suite le traitement pour ces deux cas distincts.

Le nœud admet un prédécesseur avec $sender = pred$ À la réception d'un message, le nœud récupère l'identité du nœud émetteur. Il compare ensuite cette identité à celle de son prédécesseur. Si le prédécesseur est l'émetteur du message, le nœud peut alors mettre à jour son chemin et met à zéro le $timer_{pred}$ (c'est le timer qui indique et déclenche un problème au niveau du prédécesseur). Le chemin est alors mis à jour quand le message reçu n'admet pas de réduction ou bien lorsque le nœud n'est pas en mesure de traiter la réduction reçue (illustration dans la figure 6.11).

Au cas où le nœud doit effectuer une réduction, il vérifie tout d'abord s'il est le nœud aval de la réduction (l'identité locale est égale à celle du nœud dw_P). S'il est bien dw_p (illustration figure 6.13), il vérifie que la réduction ne comporte pas d'incertitudes sur les liens (sinon le message est ignoré et le chemin est mis à jour). Il met à jour ensuite son chemin, suivant la nature du nœud réducteur (nœud du chemin ou nœud voisin du chemin). Ce nœud réducteur est alors défini comme prédécesseur. Dans le cas où le nœud local n'est pas le nœud aval de la réduction, mais le nœud amont ou le nœud réducteur, il ignore le message. Dans ces conditions, le nœud local ne peur traiter la réduction que s'il fait partie du segment de réduction. Il peut alors mettre à jour son chemin pour transmettre la validation de la réduction au nœud suivant dans le segment de réduction (détails figure 6.12).

Dans les cas échéants, le nœud fait une simple mise à jour du chemin, sans tenir compte de la réduction.

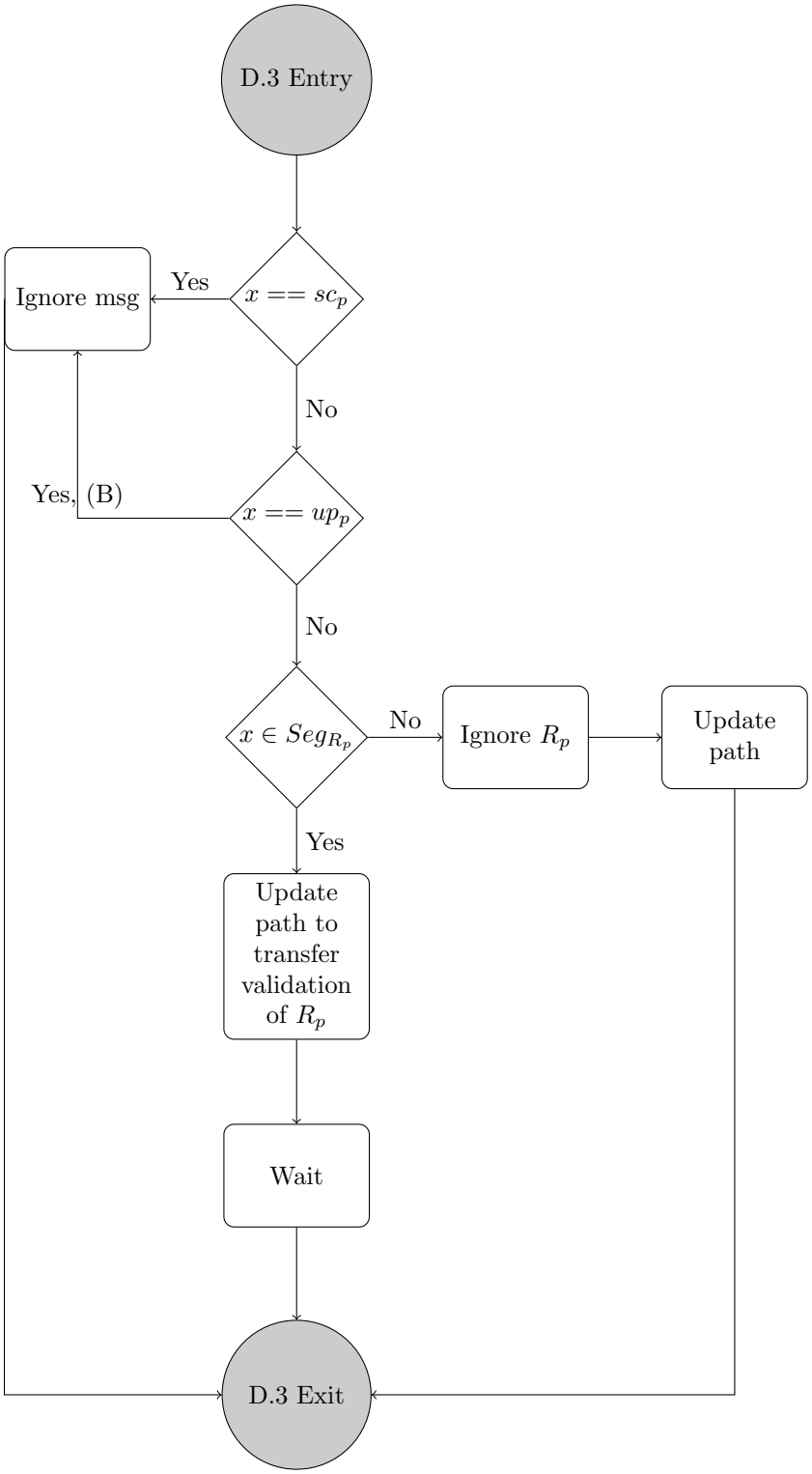


FIGURE 6.12 – Si le nœud est récepteur d'un message avec réduction alors qu'il appartient au segment de cette réduction, il met à jour son chemin pour transmettre la validation de cette réduction. Sinon, il met à jour le chemin en ignorant la réduction proposée. Dans les autres cas, le message est ignoré.

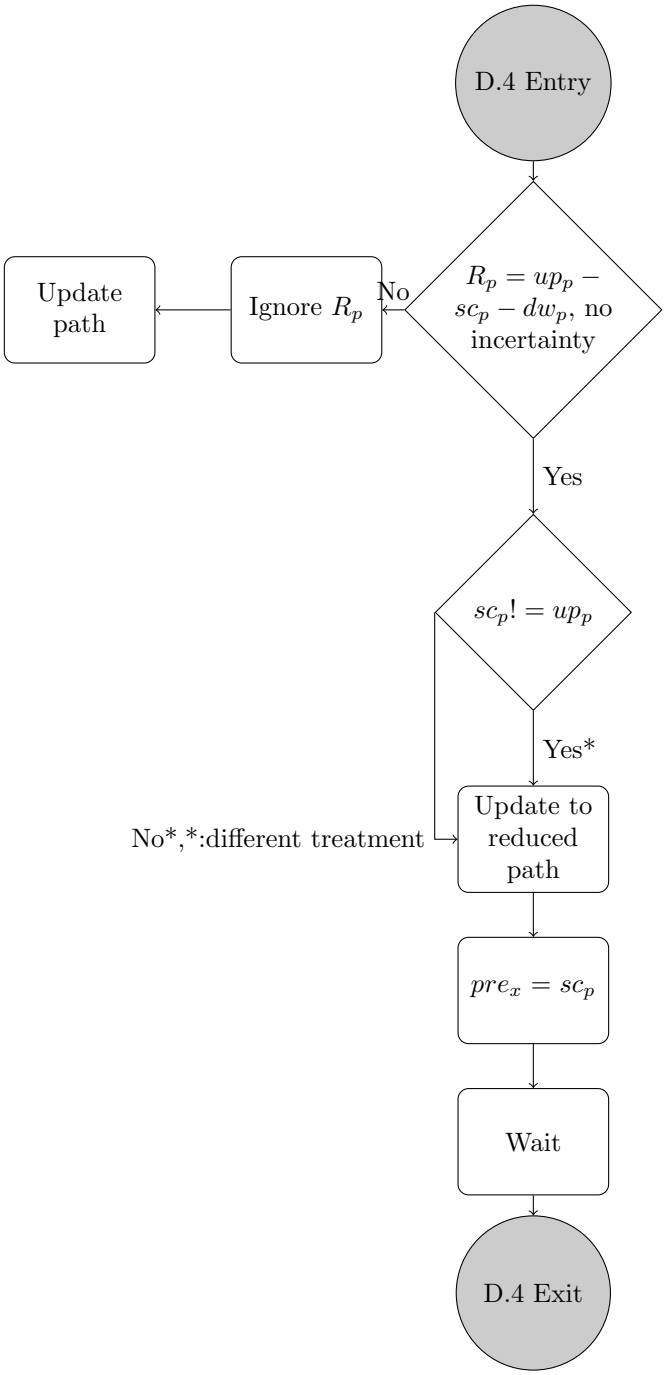


FIGURE 6.13 – Le nœud est récepteur d'un message avec réduction alors qu'il est le nœud aval de cette réduction. S'il reçoit la validation de la réduction (pas d'incertitude sur les éléments de la réduction), il réduit le chemin.

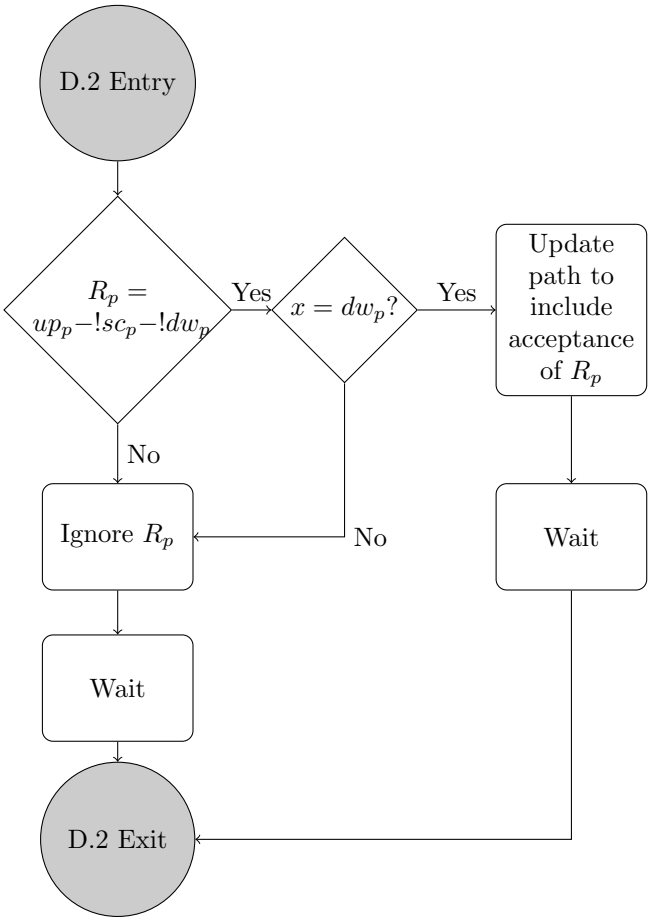


FIGURE 6.14 – Le nœud est récepteur d'un message ne provenant pas de son prédécesseur. S'il est le nœud en aval de la réduction reçue, il met à jour son chemin pour transmettre son acceptation de cette réduction.

Le nœud admet un prédécesseur avec $sender! = pred$ Le nœud admet déjà un prédécesseur, mais reçoit un message qui lui est destiné provenant d'un nœud différent de son prédécesseur. Dans ce cas, le nœud local teste si le message reçu comprend une réduction. Et il teste de plus si celle-ci comprend des incertitudes sur les deux liens de la réduction (lien nœud amont-nœud réducteur et lien nœud réducteur-nœud aval). Si ces deux tests sont vérifiés, il compare son identité à celle du nœud aval de la réduction. Si cette comparaison montre que le nœud local est dw_p, il met à jour son chemin pour inclure une acceptation de la réduction reçue. C'est la deuxième étape de la réduction. Dans les cas échéants, et s'il ne s'identifie pas comme étant le nœud aval de la réduction, il ignore le message. Ce traitement est illustré dans la figure 6.14.

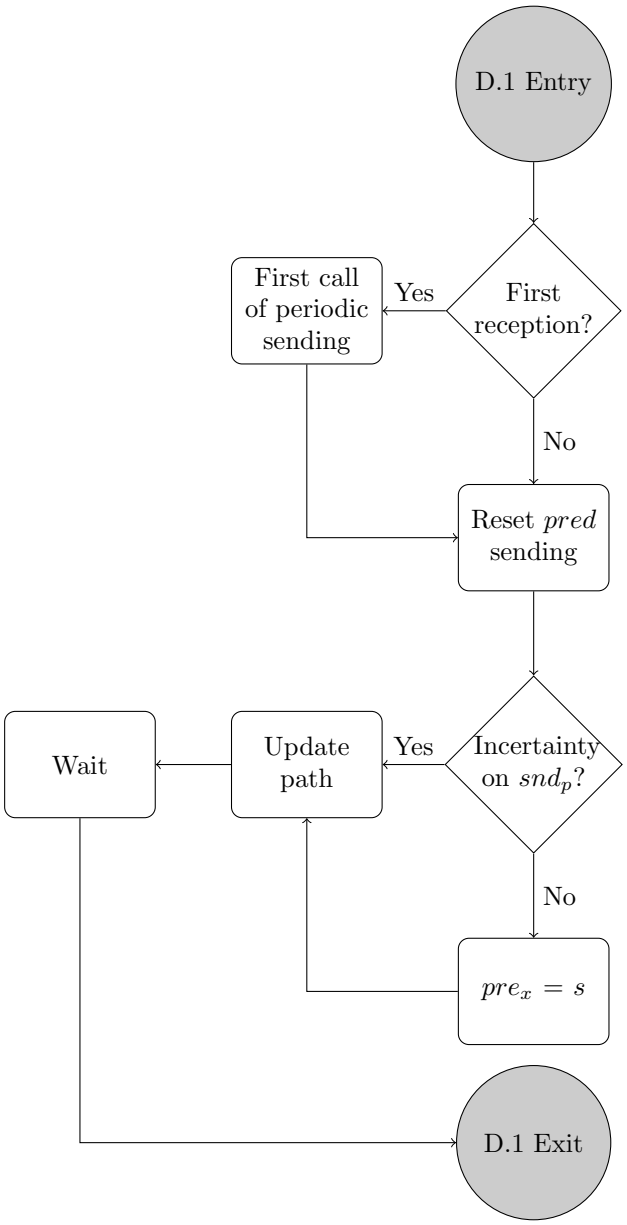


FIGURE 6.15 – Le nœud est récepteur du message alors qu'il n'a pas de prédécesseur. Il vérifie si c'est sa première réception. Dans ce cas il fait appel à l'envoi périodique. Si ce n'est pas le cas, il met à jour son chemin et n'accepte l'émetteur comme prédécesseur que si le chemin ne porte pas d'incertitude sur cet émetteur.

Le nœud n'admet pas de prédécesseur Si le nœud local n'admet pas déjà un prédécesseur, et si c'est sa première réception de message, il fait appel à la procédure d'envoi périodique (illustration dans la figure 6.15). Puis il remet à zéro le $timer_{pred}$

(timer qui fait appel à la procédure qui indique un problème au niveau du prédécesseur). Il vérifie ensuite si le chemin reçu comprend une incertitude sur l'émetteur car il ne déclare cet émetteur comme nouveau prédécesseur que, si et seulement si, il n'y a pas d'incertitude au niveau de ce nœud. À la suite de toutes ces vérifications, il met à jour son chemin.

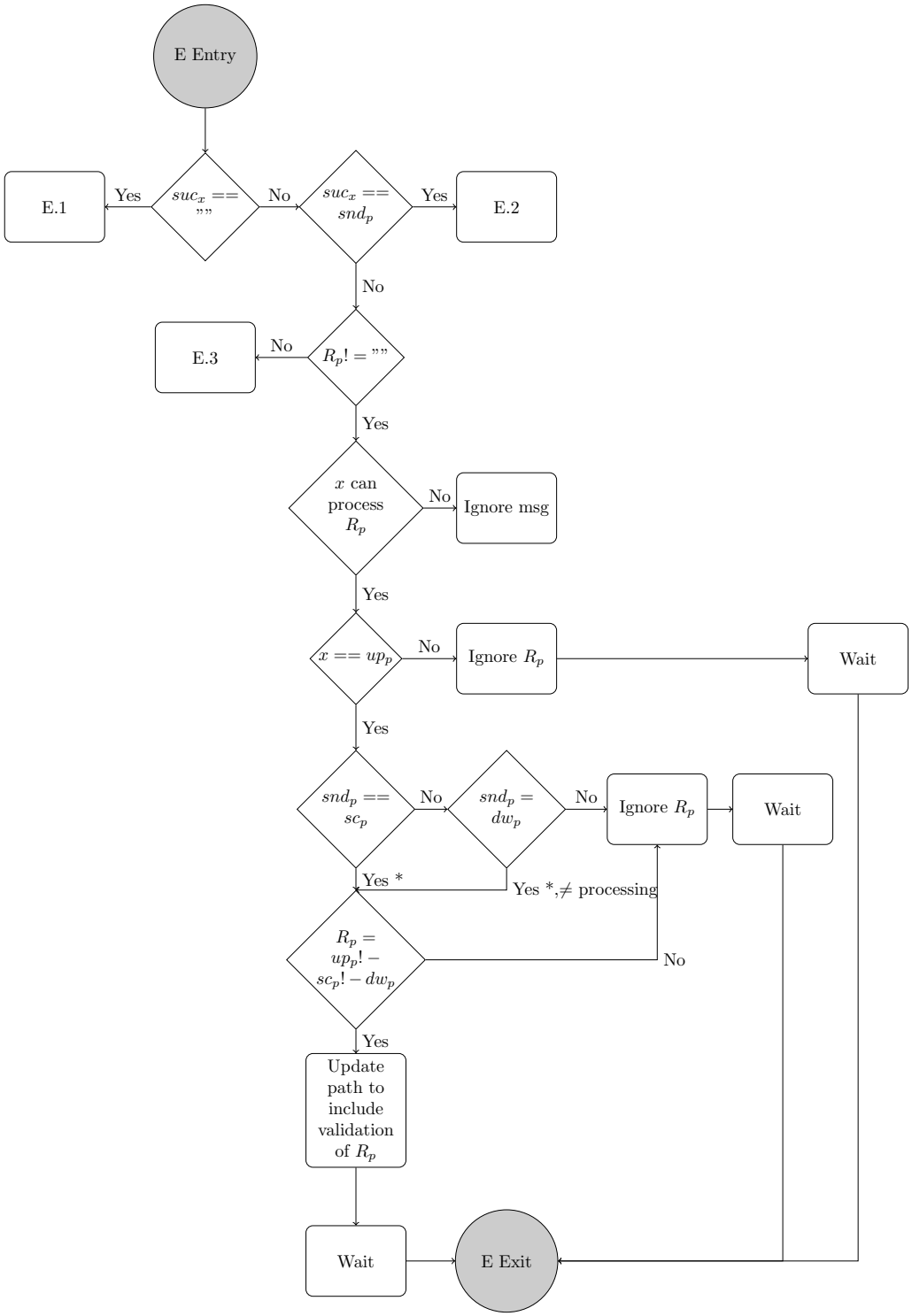


FIGURE 6.16 – Détails du block E. Le nœud est prédécesseur d'un message. Il analyse le message reçu et, suivant les cas, vérifie s'il est capable de valider une réduction proposée ; Si c'est le cas, il met à jour son chemin pour inclure la validation de cette réduction.

6.5.5 Le nœud est prédécesseur de l'émetteur

Dans le cas où le nœud est prédécesseur de l'émetteur, plusieurs cas sont envisageables (détails figure 6.16) :

- ce nœud n'a pas de successeur défini (figure 6.17),
- le successeur du nœud est l'émetteur du message, donc une réception normale ou attendue (figure 6.18),
- le nœud reçoit un message d'un autre qui n'est pas son successeur, alors qu'il a un successeur défini.

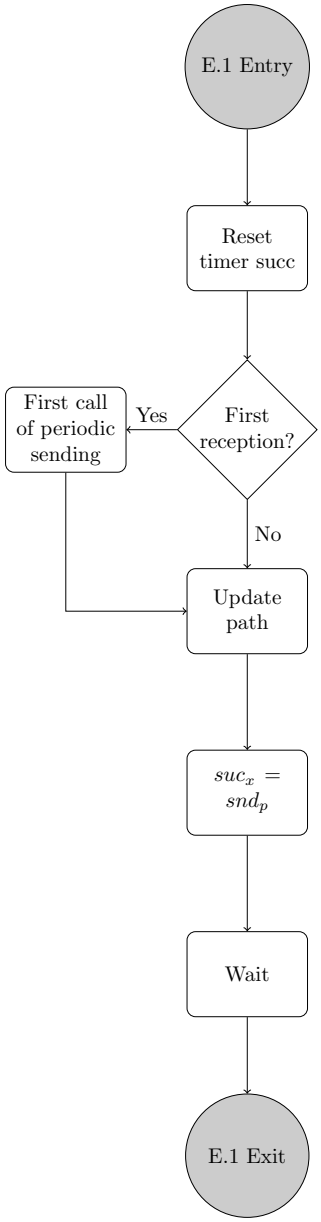


FIGURE 6.17 – Le nœud est prédecesseur du message, alors qu'il n'a pas de successeur. Si c'est sa première réception, il fait appel à l'envoi périodique. Dans tous les cas, il met à jour son chemin et désigne l'émetteur comme successeur.

Le nœud n'a pas de successeur défini Au cas où le nœud reçoit un message alors qu'il n'a pas de successeur défini, il arme un *timer* (le $timer_{succ}$) qui appelle une procédure qui indique un problème au niveau du successeur. Il vérifie ensuite si c'est sa première réception. Si c'est le cas, il fait appel à la procédure d'envoi périodique. Il met

ensuite à jour son chemin, et définit l'émetteur comme successeur (illustration 6.17).

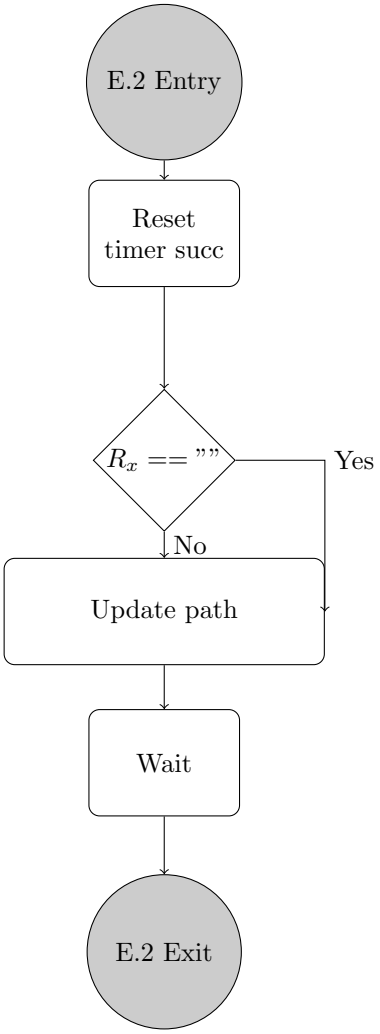


FIGURE 6.18 – Le nœud est prédécesseur d'un message de son successeur. Il met à jour son chemin suivant s'il admet une réduction ou non.

Le nœud reçoit un message de son successeur À la réception d'un message de son successeur, le nœud réarme son $timer_{succ}$. Ensuite il met à jour le chemin et le message à envoyer, selon qu'il y ait une réduction à traiter ou pas (illustration figure 6.18).

Le nœud reçoit un message contenant une réduction (qu'il peut traiter), d'un autre nœud qui n'est pas son successeur Dans le cas où le nœud reçoit un message contenant une réduction, mais qui provient d'un nœud qui n'est pas son successeur, il vérifie si le chemin reçu dans ce message est plus court que le chemin

sauvegardé localement. Si c'est le cas, il vérifie s'il est le nœud amont de la réduction, et si l'émetteur est le nœud réducteur ou le nœud aval de la réduction. Si ces conditions sont vérifiées, il effectue la réduction du chemin. Dans les autres cas, il ignore la réduction (illustration figure 6.19).

Le nœud reçoit un message sans réduction (alors qu'il a déjà une réduction en cours) d'un nœud qui n'est pas son successeur Si le nœud reçoit un message comportant une réduction et provenant d'un nœud qui n'est pas son successeur dans le chemin, il vérifie qu'il a le droit de traiter la réduction. S'il n'a pas le droit, il ignore le message. Dans le cas contraire, il vérifie s'il est le nœud amont de la réduction. S'il ne l'est pas, il ignore la réduction. Sinon, il vérifie que l'émetteur du message est le nœud réducteur ou le nœud aval de la réduction. Il met à jour son chemin pour inclure la validation de la réduction. Dans les cas échéants, la réduction est ignorée (détails fin de la figure 6.16).

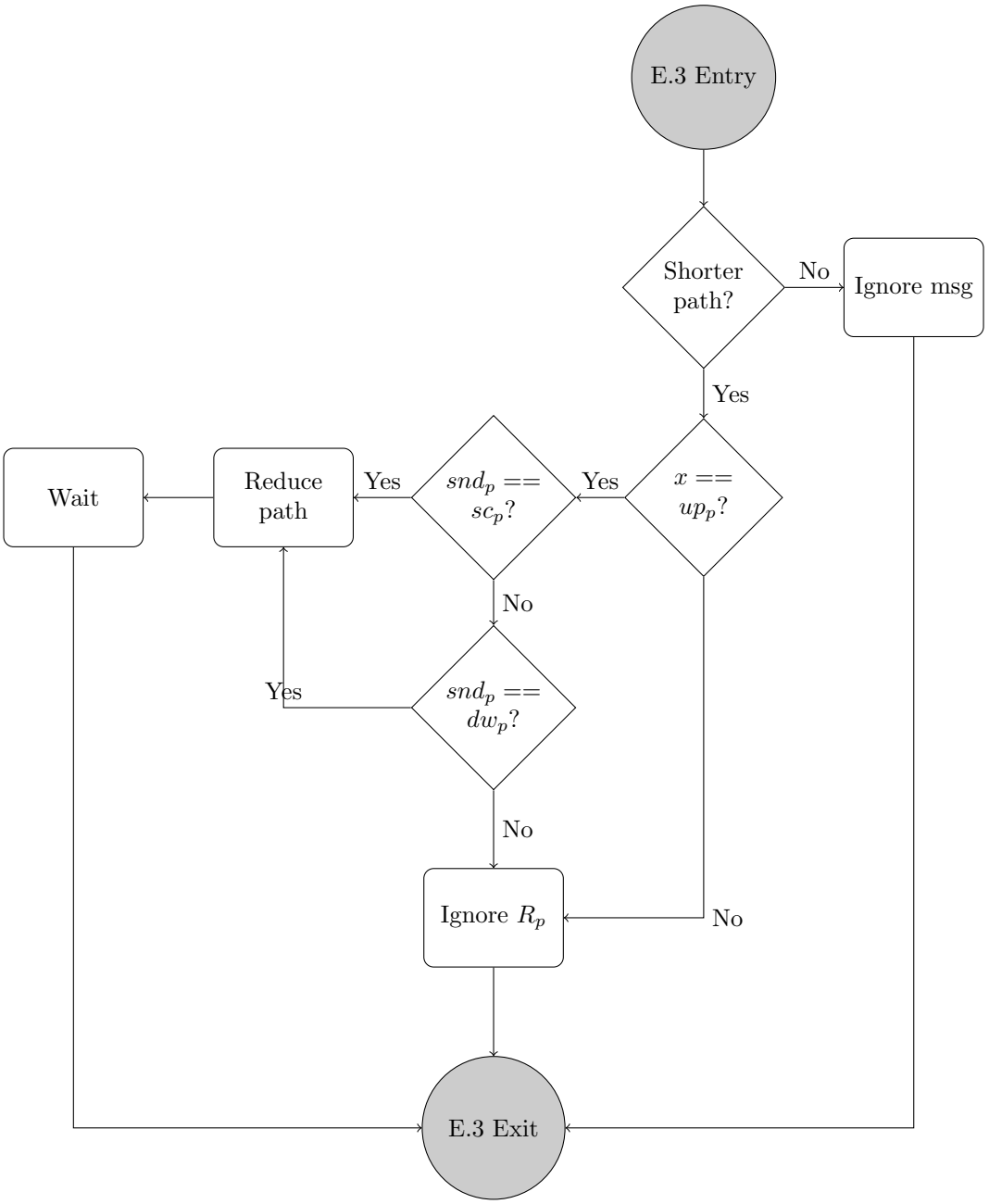


FIGURE 6.19 – Le nœud est prédécesseur d'un message avec un chemin plus court, suite à une réduction reçue où il est le nœud amont, il réduit alors son chemin. La réduction est reçue dans un message antérieur.

6.6 Validation expérimentale : protocoles, scénarios et résultats

L'algorithme de maintien de chemin a été validé suite à des tests sur routes.

PTH a été testé lors de deux scénarios de tests effectués pour la preuve de concept de l'algorithme. Les logs GPS de ces tests ont été ensuite utilisés dans l'émulateur de la plateforme *Airplug* [23] dans le but d'analyser ses performances et de le comparer avec d'autres types de routage.

La plateforme *Airplug* (détaillée dans le premier chapitre) a servi pour les tests.

PTH a été comparé à un routage géographique basic avec un service de localisation, et à un *broadcast* traditionnel, pour un même scénario sous l'émulateur de la plateforme *Airplug*. Il a été testé et comparé à l'aide de deux scénarios différents. Un premier scénario est composé de 31 voitures et un deuxième de 6 voitures.

Les émulations pour un même scénario ont commencé à partir du même état initial où deux voitures voisines entament une communication. Dans ce cas, un chemin de longueur 1 est établi entre ces deux voitures. Avec l'évolution de la topologie et la dynamique du réseau, les deux entités voisines s'éloignent l'une de l'autre. Elles deviennent alors séparées de plusieurs sauts. A ce niveau, et si un processus de maintien de chemin n'est pas déployé, la communication entre ces deux voitures est cassée vue qu'elles ne sont plus à la portée l'une de l'autre.

6.6.1 Le routage géographique

Le protocole de routage géographique est basé sur un service de localisation qui localise la destination dans le réseau et renvoie sa position géographique au nœud source. Il diffuse périodiquement une requête pour chercher la position de la destination. Cette position est utilisée par le nœud source qui renvoie le message à un de ses voisins, celui qui est le plus proche de la destination. Quand ce message atteint une certaine zone géographique (définie par l'algorithme et représentant un pourcentage de la distance séparant la source de sa destination), il est alors diffusé dans cette zone. Ce mécanisme est utilisé pour atteindre la destination, qui peut avoir changé de position entre l'instant d'envoi de sa position à la source et l'instant où elle en reçoit le message.

6.6.2 Le *broadcast*

Le *broadcast* utilisé pour la comparaison est un simple *broadcast* où tous les nœuds diffusent le message reçu. Il est utilisé pour représenter le scénario du pire cas en nombre de messages échangés entre les nœuds du réseau.

6.6.3 Le premier scénario

Le premier scénario inclut 31 voitures qui se déplacent à Compiègne, entre l'UTC (centre de recherche) et la gare. On note N le nombre de messages envoyés par le nœud source dans le réseau.

Résultats Le *broadcast* ayant inondé le réseau avec $8.4 \times N$ messages, le routage géographique avec son service de localisation a envoyé $5 \times N$ messages, et PTH a envoyé $2.4 \times N$ messages. A noter que quand 26 voitures envoient des messages avec le *broadcast* et 31 avec le routage géographique, seules 6 voitures étaient incluses dans les émissions avec PTH. Les résultats sont affichés dans la figure 6.20. Comme indiqué, PTH est avantageux dans les réseaux denses, en comparaison avec d'autres protocoles de routage.

Protocole de routage	Nombre de messages émis sur le réseau	Nombre de voitures impliquées dans les émissions
BRD	$8.4 \times N$	26
GEO	$5 \times N$	31
PTH	$2.4 \times N$	6

FIGURE 6.20 – Tableau récapitulatif des nombres de messages envoyés sur le réseau en utilisant les différentes techniques de routage, ainsi que le nombre de voitures impliquées dans ces émissions. PTH réduit le nombre de messages échangés sur le réseau, ainsi que le nombre de voitures impliquées dans les émissions.

6.6.4 Le deuxième scénario

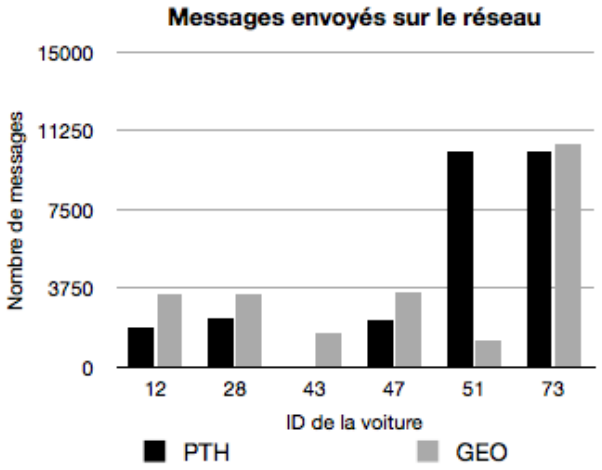
Nous considérons maintenant le cas où seulement un nombre réduit de nœuds est observé, et le cas d'un scénario où tous ces nœuds sont utilisés pour relayer la communication. Le but de cette composition est de tester PTH dans un des cas extrêmes, où il doit prouver son efficacité dans des réseaux réduits. Pour ce scénario, nous considérons seulement PTH et le routage géographique, vu qu'on sait déjà que le *broadcast* va générer le nombre maximal de messages échangés.

Ce scénario comporte 6 voitures se déplaçant entre Compiègne et Verberie (ville voisine dans l'Oise). Il ne comporte que les véhicules du convoi. Cinq voitures sur 6 seront impliquées dans la communication, où deux représentent la source et le puits, et trois représentent les voitures relais.

Résultats Ce scénario montre que PTH et le routage géographique ont des performances similaires dans les réseaux restreints où tous les nœuds sont impliqués dans la communication. Comme nous pouvons l'apercevoir dans la figure 6.21, GEO envoie $2.3 \times N$ messages sur le réseau alors que PTH en envoie $2.6 \times N$. Nous devons mentionner que la destination (ici représentée par le puits dans les figures 6.21 et 6.23, doit acquitter les réceptions avec PTH, en retransmettant le message reçu avec le chemin ajusté. En utilisant GEO, ces retransmissions ne sont pas nécessaires, et par conséquence, le nœud destination émet moins de messages sur le réseau. Cela est détaillé dans la figure 6.21.

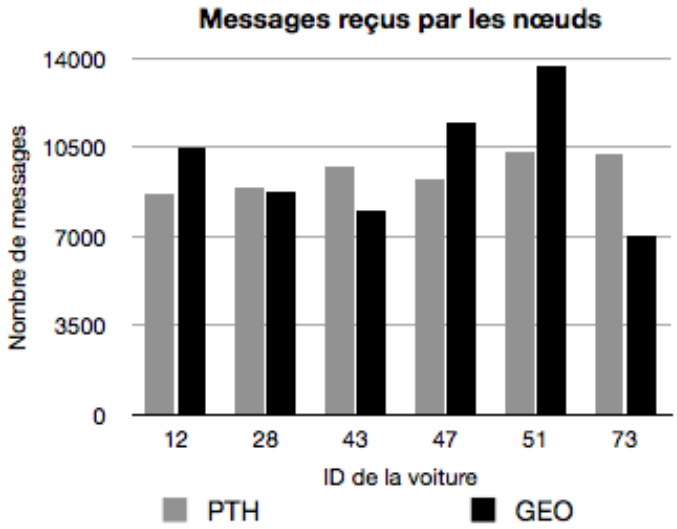
Protocole de routage	Nombre de messages envoyés sur le réseau	Nombre de voitures impliquées dans l'émission
GEO	$2.3 \times N$	6
PTH	$2.6 \times N$	5

FIGURE 6.21 – Les résultats expérimentaux comparant PTH à un routage géographique dans le second scénario sont représentés. PTH assure les mêmes performances que le routage géographique, tout en impliquant moins de voitures dans les émissions.



	relai1	relai2	relai3	relai4	puits	source
PTH	1830	2308	0	2252	10235	10244
GEO	3481	3473	1598	3521	1289	10580

FIGURE 6.22 – Nombre de messages émis par voiture lors du deuxième scénario.



	relai1	relai2	relai3	relai4	puits	source
PTH	8624	8906	9713	9280	10272	10254
GEO	10467	8776	8009	11456	13709	7008

FIGURE 6.23 – Nombre de messages reçus par voiture

6.6.5 L'analyse des résultats

PTH assure de bonnes performances en comparaison avec un routage géographique. Le *broadcast* représente le pire cas en terme de nombre de messages émis sur le réseau.

Les deux scénarios ont pour but de montrer l'efficacité de PTH dans deux des cas extrêmes de l'utilisation. Un premier cas où le réseau est très dense et que la communication n'implique que quelques voitures, et un second où le réseau est réduit à quelques voitures presque toutes impliquées dans la communication.

Dans le premier scénario, nous pouvons constater que PTH réduit le nombre de messages échangés sur le réseau, ainsi que le nombre de voitures impliquées dans les émissions. Dans le cas où les messages de de requêtes de localisation, de réponse de localisation ou d'émissions périodiques de coordonnées sont omis, PTH émet 12254 messages alors que GEO en émet 11735 (broadcast en phase finale inclus). La différence entre les deux protocoles en nombre de messages effectifs est assez réduit. Ces résultats montrent et valident l'efficacité de PTH dans ce scénario.

Dans le deuxième scénario, PTH montre des performances similaires à celles d'un routage géographique. Le nombre de messages émis sur le réseau est légèrement supérieur à celui de GEO, et cela est dû aux acquittements implicites des messages par la destination avec PTH. Au cas où nous omettons les messages de requête de localisation, de réponse de localisation ou d'émission périodique des coordonnées, nous obtenons un total de

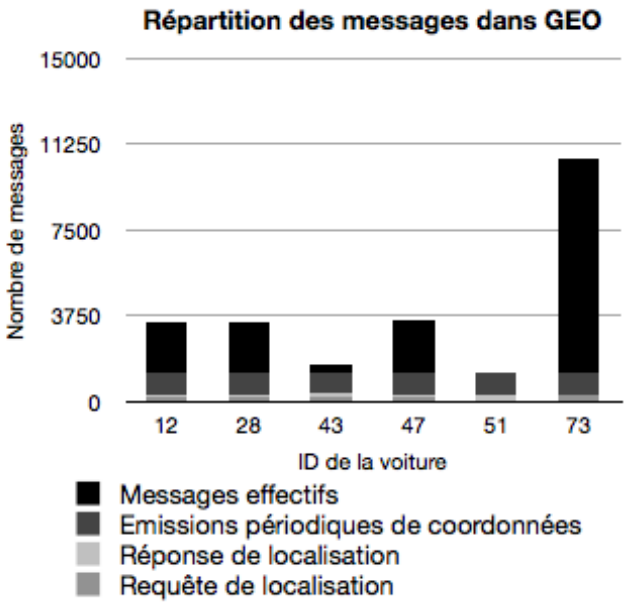


FIGURE 6.24 – Distinction entre les différents types de messages échangés avec GEO.

16237 messages effectifs échangés avec GEO (figure 6.24), ce qui revient à 40% du total des messages échangés par PTH. A noter que l'échange périodique des coordonnées et le service de localisation sont présents dans les routages géocast.

Considérons un géocast basé sur un échange de coordonnées entre la source A et la destination B voisines à la base. Supposons que ces deux entités se déplacent l'une loin de l'autre, et qu'un nœud intermédiaire remarque qu'il est voisin du destinataire et de la source. Il relaie alors les messages. Quand la distance entre A et B devient plus importante, les nœuds intermédiaires proches de la source se mettrons à envoyer en *broadcast* les messages reçus. La source indique dans son message la dernière position connue de la destination. Pour pallier aux déplacement rapide de la destination, la source doit indiquer une zone de diffusion finale plus étendue. De même, la destination avertit régulièrement la source de sa position, cette dernière devra augmenter la zone pour assurer la communication. L'imprécision augmente avec la vitesse de déplacement la destination et avec la distance entre A et B qui accroit les délais de la communication et donc l'imprécision. La zone de *broadcast* devient de plus en plus importante. Par conséquence, le géocast finira par avoir les mêmes performances qu'un *broadcast*.

Nous pouvons déduire de ces études que PTH est assez performant dans la majorité des scénarios possibles. Il garantit une implication minimale des voitures dans l'envoi de messages et un nombre réduit de messages échangés.

6.7 Conclusion

Dans ce chapitre, nous avons présenté le travail de conception du protocole PTH basé sur l'algorithme de maintien de chemin étudié précédemment. Ce travail nous a permis d'avoir une preuve de concept réelle sur route de la faisabilité et de l'intérêt de cet algorithme en pratique pour les réseaux de véhicules.

En outre, nous avons pu mener une étude comparative de ses performances via l'émulateur *Airplug-EMU*. Au cours de cette étude, notre protocole s'est avéré plus efficace dans les réseaux denses, et satisfaisant dans les réseaux à faible densité.

Chapitre 7

Conclusion et perspectives

Les réseaux mobiles en général et les réseaux véhiculaires en particulier sont en cours de développement. Les travaux sur ces réseaux se multiplient, balayant les aspects communications et performances. Les communications V2V et V2I ont évolué avec la croissance du nombre d'applications dédiées aux réseaux mobiles et véhiculaires. La technologie émergente offre des facilités à ces développements.

7.1 Les travaux de cette thèse

Nous nous sommes intéressés dans cette thèse à étudier les performances des réseaux mobiles. Des études ont été menées dans notre laboratoire et sur route afin de mieux comprendre la nature de ces réseaux. Les pertes étant importantes dans les communications dans un environnement sans fil mobile, nous avons proposé une architecture opportuniste pour l'accès V2I depuis un réseau véhiculaire pour la remontée des données.

L'architecture proposée a prouvée son efficacité en réduisant les pertes de paquets dans le réseau véhiculaire.

D'autre part, les communications V2V étant un facteur primordial pour la réussite des communications dans les réseaux, nous nous sommes intéressés particulièrement à l'étude et la proposition d'une solution pour la continuité des communications *unicast*. Le routage étant obsolète dans les réseaux fortement dynamiques, nous avons proposé un algorithme de maintien de chemin pour assurer la continuité d'une communication entre deux entités voisines.

Cet algorithme a été implémenté et validé sur route et par émulation. L'étude de cet algorithme nous a orienté vers une modélisation des réseaux dynamiques. Nous nous sommes ainsi intéressés à lier le comportement d'un algorithme à la dynamique du réseau dans lequel il est implémenté. Nous avons aussi proposé une métrique algorithmique pour la mesure de performances des algorithmes. Elle permet de comparer leurs performances dans des réseaux différents et des conditions différentes.

L'algorithme de maintien de chemin assure des performances assez satisfaisantes pour le maintien des communications établies entre deux voisins. Il a été comparé à un algorithme géocast dans deux types de scénarios.

Le travail décrit dans cette thèse comprend plusieurs aspects de la communication dans les réseaux mobiles. Cependant, plusieurs idées restent en perspective pour l'amélioration de ces travaux.

7.2 Le transport

Le maintien de chemin a été conçu comme première étape vers une communication fiable entre deux entités. Il doit alors être complété par un protocole de transport qui utilisera le chemin construit par cet algorithme pour assurer le moins de pertes sur les liens de ce chemin.

Le protocole de transport que nous pouvons envisager sera une implémentation indépendante du protocole de maintien de chemin. Ce dernier fournira le chemin construit au protocole de transport qui gèrera les messages reçus sur les entités du chemin, ainsi que sur les voisins de ce chemin.

La gestion des messages reçus pourra se faire par des listes. Nous pouvons envisager la répartition suivante pour les messages reçus : messages reçus et acquittés, messages envoyés et non acquittés et messages envoyés et acquittés.

L'augmentation du nombre de messages envoyés et non acquittés indique un problème au niveau du lien entre le nœud courant et son successeur dans le chemin. L'augmentation de délai entre deux réceptions de la part du successeur indique un ajustement de débit au niveau de ce successeur.

Quand le nœud présent remarque un problème au niveau de son successeur, il doit ajuster son débit de façon à réduire la fréquence d'envoi de messages sur ses liens, laissant le temps à l'algorithme de maintien de chemin pour réparer le chemin sur les liens cassés.

Des réflexions sur la problématique du transport pour une communication *unicast* dans un réseau mobile ont été entamées. Ce travail reste à poursuivre. Notre protocole de routage pourra être utilisé avec un protocole de transport assurant le contrôle de bout en bout sur les données.

7.3 La sécurité

La sécurité dans les réseaux mobiles représente un défi pour les communications. Il est nécessaire d'assurer la protection de la communication contre les attaques et les malveillances causées par d'autres nœuds dans le réseau.

La sécurité des communications *unicast* est généralement assurée par l'échange de clés entre les parties de la communication. La gestion des clés publiques et privées est souvent coûteuse en nombre de messages échangés. De plus, une mise à jour fréquente de ces structures est nécessaire pour garantir la validité de ces informations.

La sécurisation de la communication doit porter sur une confiance dans le nœud avec lequel la source établit la communication (la destination doit aussi avoir une confiance dans la source), et sur une confiance dans les données échangées.

Le principe du maintien du chemin, partant d'une communication entre deux voisins, permet l'échange de clés et limite et anticipe les problèmes de confiance et le problème

du *man in the middle*.

7.4 Le déploiement

Le déploiement des protocoles cités restent un challenge. L'installation du matériel, même à bas coût, dans les véhicules n'est pas encore une réalité. Même si plusieurs fabricants automobiles fournissent des véhicules équipés de Bluetooth ou de 3G, les équipements nécessaires pour des communications V2V ou V2I comme nous les prévoyons, ne sont pas encore disponibles.

Une sensibilisation envers l'utilité et l'utilisation de ces services est une étape primordiale pour rendre leur déploiement utile et efficace.

D'autre part, pour que ces protocoles soient capables de tourner d'une façon robuste dans les véhicules, un processus de validation expérimentale doit être mis en œuvre. Des tests dans les cas les plus critiques doivent être effectués. Il reste donc à prévoir des tests de plus grande ampleur de nos travaux. Le laboratoire Heudisayc travaille en ce sens en construisant un dispositif communiquant embarqué. Nous pouvons espérer que nos solutions pourraient être utilisées à large échelle moyennant quelques améliorations.

Bibliographie

[1] http://www.vanet.info/projects.

[2] B.a.t.m.a.n. http://www.open-mesh.org/projects/open-mesh/wiki.

[3] CALM, continuous communications for vehicles. http://www.calm.hu/.

[4] The CVIS project. http://www.cvisproject.org/.

[5] Equipex utc. http://www.utc.fr/recherche-innovation/equipex-robotex-figures.php.

[6] The gst project. http://www.gstforum.org/.

[7] Labex utc. http://www.utc.fr/recherche-innovation/laboratoire-excellence-ms2t.php.

[8] Pre-Drive C2X. http://www.pre-drive-c2x.eu/.

[9] RITA/ITS, IEEE 1609 - family of standards for wireless access in vehicular environments (WAVE). http://www.standards.its.dot.gov/fact_sheet.asp?f=80.

[10] Safespot. http://www.safespot-eu.org/.

[11] Secure vehicular communication. http://www.sevecom.org/.

[12] SIM-TD Field test of car2X communication. http://www.cvisproject.org/en/links/sim-td.htm.

[13] Vehicle infrastructure integration. http://www.vehicle-infrastructure.org/.

[14] Car 2 car communication consortium manifesto, overview of the c2c-cc system. http://www.car-2-car.org/fileadmin/downloads/C2C-CC_manifesto_v1.1.pdf, August 2007.

[15] IEEE standard for information technology telecommunications and information exchange between systems local and metropolitan area networks specific requirements. part 11 : Wireless lan medium access control. http://www.ahltek.com/WhitePaperspdf/802.11-20%20specs/802.11-2007.pdf, 6 2007.

[16] A. F. Anta, A. Milani, M. A. Mosteiro, and S. Zaks. Opportunistic information dissemination in mobile ad-hoc networks : the profit of global synchrony. *Distributed Computing*, 25(4) :279–296, 2012.

[17] R. Baldessari, C. Bernardos, and M Calderon. *GeoSAC*-scalabe address autoconfiguration for VANET using geographic networking concepts. In *PIMRC*, septembre 2008.

[18] R. Baldoni, A. Fernández Anta, K. Ioannidou, and A. Milani. The impact of mobility on the geocasting problem in mobile ad-hoc networks : Solvability and cost. *Theor. Comput. Sci.*, 412(12-14) :1066–1080, 2011.

[19] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward. A distance routing effect algorithm for mobility (DREAM). *ACM/IEEE Mobicom*, 1998.

[20] B. Bellur and R. G. Rogier. A reliable, efficient topology broadcast protocol for dynamic networks. *IEEE Computer and Communications Societies (INFOCOM'99)*, 1 :178–186, 1999.

[21] P. Borgnat, E. Fleury, J.-L. Guillaume, and C. Robardet. Characteristics of the dynamic of mobile networks. In *Proceedings of Bioinspired Models of Network, Information, and Computing Systems - 4th International Conference, BIONETICS 2009 (revised selected papers)*, volume 6811, pages 130–139, 2009.

[22] B. Bui-Xuan, A. Ferreira, and A. Jarry. Evolving graphs and least cost journeys in dynamic networks. In *Proceedings of WiOpt'03 – Modeling and Optimization in Mobile, Ad-Hoc and Wireless Networks*, pages 141–150, Sophia Antipolis, March 2003. INRIA Press.

[23] A. Buisset, B. Ducourthial, F. El Ali, and S. Khalfallah. Vehicular networks emulation. In *International Conference on Computer Communication Networks (ICCCN 2010)*, Zurich, Suisse, Aout 2010.

[24] V. Bychkovsky, B. Hull, A. Miu, H. Balakrishnan, and S. Madden. A measurement study of vehicular internet access using in situ wi-fi networks. *International Conference on Mobile Computing and Networking (MobiCom), Los Angeles, CA, USA*, page 50–61, 2006.

[25] M. Calderon, H. Moustafa, C. Bernardos, and R. Baldessari. *IP Address autoconfiguration in vehicular networks*, chapter 9 in *Vehicular Networks, Techn., Standards and App.* CRC Press (Taylor & Francis Group), Auerbach, 2009.

[26] A. Casteigts, S. Chaumette, and A. Ferreira. Distributed computing in dynamic networks : Towards a framework for automated analysis of algorithms. *CoRR*, abs/1102.5529, 2012.

[27] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. Time-varying graphs and dynamic networks. In *Proceedings of ADHOC-NOW*, volume 6811 of *LNCS*, pages 346–359, Paderborn, 2011.

[28] V. Devarapalli, R. Wakikawa, A. Petrescu, and P. Thubert. Network mobility (NEMO) basic support protocol. IETF RFC 3963, January 2005.

[29] B. Ducourthial. About efficiency in wireless communication frameworks on vehicular networks. In *Proceeding of the ACM WIN-ITS workshop colocated with IEEE ACM QShine'07*, Vancouver, British Columbia, August 2007.

[30] B. Ducourthial, Y. Khaled, and M. Shawky. Conditional transmissions : a communication strategy for highly dynamic vehicular ad hoc networks. *IEEE Transactions on Vehicular Technology, special issue on vehicular communication networks*, 56(6) :3348–3357, Novembre 2007.

[31] B. Ducourthial, Y. Khaled, and M. Shawky. Conditional transmissions, a strategy for highly dynamic vehicular ad hoc networks. In *8th IEEE Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2007)*, Helsinky, Finland, 18-21 Juin 2007.

[32] B. Ducourthial and S. Khalfallah. A platform for road experiments. In *69th IEEE Vehicular Technology Conference*, Barcelone, Espagne, Avril 2009.

[33] B. Ducourthial, S. Khalfallah, and F. Petit. Best-effort group service in dynamic networks. In *SPAA*, pages 233–242, Santorini, Greece, 2010.

[34] T. Ernst. Le support des réseaux mobiles dans ipv6. *Revue Technique et Science Informatiques*, 25(5) :573–598, 2006.

[35] M. Fazio, P. Palazzi, S. Das, and M. Gerla. Facilitating real-time applications in vanets through fast address auto-configuration. Proc. of IEEE CCNC/NIME, January 2007.

[36] A. Ferreira. Building a reference combinatorial model for MANETs. *IEEE Network*, 18(5) :24–29, 2004.

[37] E. Fonseca, A. Festag, R. Baldessari, and R. Aguiar. Support of anonymity in vanets, putting pseudonymity into practice. Proc. IEEE WCNC, March 2007.

[38] R. Gass, J. Scott, and C. Diot. Measurements of in-motion 802.11 networking. *IEEE Workshop on Mobile Computing Systems & Applications (WMCSA)*, page 69–74, April 2006.

[39] V. Gauthier. Crosslayer et qos dans les réseaux ad hoc sans fils. *Mémoire de master, Institut National des Télécommunications, Evry, France*, 2003.

[40] M. Goncalves Rubinstein, F. Ben Abdesslem, S. Rodrigues Cavalcanti, M. Elias Mitre Campista, R. dos Santos Alves, L. Henrique Maciel Kosmalski Costa, M. Dias de Amorim, and O. Carlos Muniz Bandeira Duarte. Measuring the capacity of in-car to in-car vehicular networks. *Proc. of the 69th IEEE VTC 2009-Spring, Barcelona.*, 2009.

[41] D. Hadaller, S. Keshav, T. Brecht, and S. Agarwal. Vehicular opportunistic communication under the microscope. *ACM International Conference on Mobile Systems, Applications, and Services (MobiSys), San Juan, Puerto Rico*, June 2007.

[42] B. Haeupler and D.R. Karger. Faster information dissemination in dynamic networks via network coding. *CoRR*, abs/1104.2527, 2011.

[43] F. Harary and G. Gupta. Dynamic graph models. *Mathematical and Computer Modelling*, 25(7) :79–88, 1997.

[44] H. Hartenstein and K. Laberteaux. *VANET : Vehicular Applications and Inter-Networking Technologies*. Wiley Online Library, 2010.

[45] G. He. Destination-sequenced distance vector (DSDV) protocol.

[46] Y. Khaled, B. Ducourthial, and M. Shawky. IEEE 802.11 performances for inter-vehicle communication networks. *Proceedings of the 61st Semiannual Vehicular Technology Conference (VTC 2005-Spring), Stockholm, Sweden*, May-June 2005.

[47] S. Khalfallah and B. Ducourthial. Bridging the gap between simulation and experimentation in vehicular networks. In *IEEE 72nd Vehicular Technology Conference (VTC2010-Fall)*, Ottawa, Canada, 6–9 Septembre 2010.

[48] F. Kuhn, N. Lynch, and R. Oshman. Distributed computation in dynamic networks. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC '10, pages 513–522, New York, NY, USA, 2010. ACM.

[49] Y. Lin, Y. Chen, and S. Lee. Routing protocols in vehicular ad hoc networks : A survey and future perspectives. *J. Inf. Sci. Eng.*, 26(3) :913–932, 2010.

[50] J. Luo and J.-P. Hubaux. A survey of inter-vehicle communication. *School of Computer and Communication Sciences, EPFL, Switzerland*, technical report IC/2004/24.

[51] M. Mauve, J. Widmer, and H. Hartenstein. A survey on position-based routing in mobile ad hoc networks. *IEEE Network Magazine, 15(6)*, page 30–39, November 2001.

[52] N. Meghanathan. Survey and taxonomy of unicast routing protocols for mobile ad hoc networks. *The International Journal on Applications of Graph Theory in Wireless Ad hoc Networks and Sensor Networks*, 1(1), 2009.

[53] J. Monteiro, A. Goldman, and A. Ferreira. Performance evaluation of dynamic networks using an evolving graph combinatorial model. *WiMob*, 2006.

[54] S. Murthy and J. J. Garcia-Luna-Aceves. An efficient routing protocol for wireless networks. *Mob. Netw. Appl.*, 1(2) :183–197, October 1996.

[55] R. Ogier, F. Templin, and M. Lewis. Topology dissemination based on reverse-path forwarding (TBRPF). *RFC 3684, Internet Engineering Task Force (IETF)*, 2003.

[56] J. Ott and D. Kutscher. Drive-thru internet : IEEE 802.11b for "automobile" users. *IEEE Conference on Computer Communications (INFOCOM), Hong Kong*, March 2004.

[57] R. O'Dell and R. Wattenhofer. Information dissemination in highly dynamic graphs. In *Proceedings of Joint Workshop on Foundations of Mobile Computing DIALM-POMC*, pages 104–110, 2005.

[58] V. D. Park and M. S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. *IEEE Computer and Communications Societies (INFOCOM'97)*, 3 :1405–1413, 1997.

[59] M. R. Pearlman and Z. J. Hass. Determining the optimal configuration for the zone routing protocol. *IEEE, Selected Areas in Communication*, 17 :1395–1414, 1999.

[60] G. Pei, M. Gerla, and T. Chen. Fisheye state routing in mobile ad hoc networks. In *In ICDCS Workshop on Wireless Networks and Mobile Computing*, pages 71–78, 2000.

[61] A. Qayyum, A. Laouiti, and L. Viennot. Multipoint relaying technique for flooding broadcast messages in mobile wireless networks. *HICSS*, Janvier 2002.

[62] M. Wellens, B. Westphal, and P. Mahonen. Performance evaluation of IEEE 802.11-based wlans in vehicular scenarios. *IEEE Vehicular Technology Conference, Dublin, Ireland*, Apr. 2007.

[63] J. Whitbeck, M. Dias de Amorim, V. Conan, and J.-L. Guillaume. Temporal reachability graphs. In *Proceedings of ACM Mobicom*, Istambul, 2012.