THÈSE

présentée à
l'Université Louis Pasteur de Strasbourg
Département Informatique
Laboratoire LSIIT, UMR CNRS-ULP N°7005

Pour obtenir le grade de
Docteur de l'Université Louis Pasteur
Mention SCIENCES
Spécialité INFORMATIQUE

par

Christophe JELGER

Gestion des équipements mobiles et communications de groupe dans l'Internet Nouvelle Génération

Soutenue publiquement le 29 octobre 2004 devant le jury composé de :

M. **Philippe CLAUSS**, Rapporteur interne
Professeur à l'Université Louis Pasteur - Strasbourg
M. **Olivier FESTOR**, Rapporteur externe
Directeur de recherche à l'INRIA
M. **Eric FLEURY**, Examinateur
Professeur à l'INSA de Lyon
M. **Thomas NOEL**, Co-encadrant de thèse
Maître de conférences HDR à l'Université Louis Pasteur - Strasbourg
M. **Jean-Jacques PANSIOT**, Directeur de thèse
Professeur à l'Université Louis Pasteur - Strasbourg
M. **Guy PUJOLLE**, Rapporteur externe
Professeur à l'Université Pierre et Marie Curie - Paris

Remerciements

Je tiens dans un premier temps à remercier Jean-Jacques Pansiot, Thomas Noel, et Benoît Hilt pour m'avoir permis de faire cette thèse ainsi que pour la confiance qu'ils m'ont témoignée. Sans eux, ce travail n'aurait certainement jamais vu le jour.

Je remercie également Messieurs Olivier Festor, Guy Pujolle, et Philippe Clauss pour l'intérêt qu'ils ont porté à mes travaux en acceptant d'être les rapporteurs de ma thèse. Je remercie pareillement Monsieur Eric Fleury d'avoir bien voulu participer au jury en tant qu'examinateur.

Merci à Thomas, qui a encadré mes travaux de recherche et qui m'a permis de prendre une part active dans des projets de collaboration avec d'autres équipes de recherche. Je le remercie vivement pour sa confiance et son amitié, ainsi que pour son enthousiasme communicatif et son indéfectible dynamisme.

Merci à Jean-Jacques, mon directeur de thèse, pour m'avoir fait bénéficier de ses conseils, de son expérience, et de son incroyable connaissance dans le domaine des réseaux informatiques. Son éternel bon sens et sa sobriété en ont fait une source inépuisable d'idées et de suggestions.

Un grand merci à tous les membres de l'équipe Réseaux et Protocoles du LSIIT avec lesquels j'ai pris beaucoup de plaisir à travailler. Merci à Arnaud pour le travail qu'il a réalisé, pour sa bonne humeur, et pour son soutien. Merci également à Nicolas pour son amitié et ses encouragements, et merci à Jean pour avoir toujours été présent dans les moments critiques.

Merci à ma famille et à Sophie pour leur soutien sans faille.

à mes parents.

Table des matières

Introduction

Contexte général

Ces dernières années, les applications multimédia sont devenues très populaires auprès des utilisateurs de l'Internet. Un vaste choix de services tels que la radio et la télévision par Internet est en effet disponible : on peut ainsi écouter certaines stations de radio depuis n'importe quel ordinateur connecté au réseau planétaire. De même, on assiste à un engouement croissant pour les jeux en réseau qui permettent de regrouper des joueurs géographiquement dispersés sur toute la planète. Au niveau professionnel, on observe également une utilisation croissante de la visioconférence, qui permet d'organiser des réunions entre des participants parfois distants de plusieurs milliers de kilomètres. Toutes ces applications ont en commun la particularité de faire communiquer un groupe d'utilisateurs.

La nature de ce type d'applications rend ainsi particulièrement attractif l'utilisation des communications multicast. Les différents protocoles mettant en œuvre les communications multicast permettent en effet de faire communiquer un groupe d'utilisateurs tout en minimisant l'utilisation de la bande passante du réseau. En effet, les communications classiques de type unicast (point à point) qui mettent en jeu deux entités communicantes ne sont pas optimisées pour les communications de groupe. En multicast, un flux d'informations destiné à plusieurs participants d'un groupe est ainsi envoyé de manière unique. L'infrastructure du réseau est responsable de la duplication de ce flux lorsque le chemin emprunté par ce dernier diverge pour atteindre plusieurs destinataires. Une communication de groupe peut également être unidirectionnelle ou multidirectionnelle. Par exemple, une émission de radio sur Internet est de type 1 vers N, c'est-à-dire qu'il existe une unique source de données (le serveur de la station de radio) et N récepteurs. Au contraire, un jeu en réseau ou une visioconférence est une communication de type M vers N : elle met en œuvre M sources et N récepteurs (un récepteur n'est pas forcément une source et une source n'est pas nécessairement un récepteur).

Dans le même temps, on assiste à un déploiement massif de réseaux locaux sans-fil de type IEEE 802.11 (regroupés sous le terme "WIFI"). Parallèlement, l'utilisation d'assistants électroniques et d'ordinateurs portables permettant d'utiliser de tels réseaux se banalise. De plus et à plus grande échelle, les technologies de téléphonie mobile de troisième génération sont capables de proposer un accès à des débits qui permettent le transfert de contenu multimédia vers et à partir de téléphones portables. Au Japon par exemple, il est ainsi possible d'écouter une radio Internet sur un téléphone portable multimédia. Toutes ces technologies introduisent un nouveau paramètre dans les communications numériques : la mobilité des utilisateurs. Dans le cadre de l'Internet Nouvelle Génération, la gestion de la mobilité repose notamment sur la version 6 du protocole IP (i.e. IPv6) et sur les spécifications du protocole Mobile IPv6 qui permet en particulier de maintenir les connexions des utilisateurs mobiles lors de leurs déplacements. De plus pour un opérateur de réseaux sans-fil, il est bien entendu impératif d'être capable de mettre à disposition des utilisateurs mobiles les services multimédia décrits précédemment. En particulier, cela implique de pouvoir offrir un service multicast au niveau des réseaux sans-fil. Or, les spécifications actuelles du protocole Mobile IPv6 n'évoquent que très brièvement l'utilisation des communications multicast dans un environnement mobile utilisant ce protocole. En outre, certains aspects de l'intégration des communications multicast et de la mobilité n'ont jamais été évalués. Enfin, les propositions existantes pour gérer cette intégration sont souvent très éloignées des technologies actuelles permettant de gérer les communications multicast : elles sont donc difficilement déployables en pratique.

Réseaux sans-fil à infrastructure

Dans un premier temps, nous avons voulu évaluer la situation actuelle de l'intégration des communications de groupe dans des réseaux sans-fil à infrastructure. Ces réseaux sont basés sur une infrastructure de points d'accès reliés entre eux ainsi qu'à l'Internet par un réseau d'interconnexion généralement filaire. De plus, toutes les communications à partir et vers une station mobile se font via un point d'accès. Dans le cadre de ces réseaux, nous avons en particulier étudié les performances des deux solutions évoquées dans les spécifications du protocole Mobile IPv6 afin de gérer la mobilité des récepteurs d'un groupe multicast. Cette étude nous a permis de mettre en avant les avantages et les inconvénients de chacune des deux solutions. Nous avons ainsi pu proposer et évaluer une extension permettant de tirer parti des avantages de ces deux solutions. Nous avons quantifié le gain apporté par notre optimisation en la comparant aux solutions retenues dans le protocole Mobile IPv6. Nous avons ensuite considéré le cas particulier d'une source mobile dans le cadre d'arbres multicast spécifiques à une source. Ces arbres sont en effet enracinés à la position courante de la source. Lorsque cette dernière est mobile, la seule solution évoquée par le protocole Mobile IPv6 introduit un routage triangulaire qui réduit fortement les performances de la diffusion multicast. Nous proposons donc une alternative permettant en particulier de construire de manière dynamique et décentralisée un arbre multicast enraciné à la position courante d'une source mobile. Nous utilisons une approche s'intégrant parfaitement avec le protocole Mobile IPv6 et ne nécessitant en particulier pas de modifications au niveau des routeurs de l'Internet. Les performances de notre proposition ont été évaluées et comparées à celles de la solution évoquée dans Mobile IPv6. Comparée à cette dernière solution, notre proposition permet de diviser en moyenne par deux la longueur des routes multicast entre la source mobile et les récepteurs.

Réseaux ad hoc hybrides

Un réseau ad hoc est un réseau sans-fil qui, contrairement aux réseaux sans-fil classiques, n'a pas besoin d'une infrastructure pour exister. Dans un réseau ad hoc, les équipements constituant le réseau doivent coopérer afin de permettre à deux stations se trouvant hors de portée radio (l'une de l'autre) de communiquer grâce à des nœuds intermédiaires. Le fonctionnement d'un tel réseau nécessite donc l'utilisation d'un protocole de routage multi-sauts. Contrairement à un réseau ad hoc classique fonctionnant de manière autonome, un réseau ad hoc hybride est capable d'offrir une connectivité vers l'Internet. Pour se faire, une ou plusieurs stations du réseau ad hoc deviennent des passerelles pour les autres nœuds du réseau.

La notion de connectivité globale implique que chaque nœud du réseau ad hoc hybride ait une adresse IPv6 globale. Il est bien entendu préférable que cette adresse soit obtenue de manière automatique et dynamique. Nous avons ainsi proposé un mécanisme permettant d'attribuer des adresses globales aux nœuds d'un réseau ad hoc. Cette proposition est construite autour du concept innovant (dans un réseau ad hoc) de continuité de préfixe. La continuité de préfixe n'a de sens que lorsque plusieurs passerelles utilisant des préfixes différents sont accessibles à partir du réseau ad hoc. Ce concept implique que tous les nœuds utilisant un préfixe particulier forment un graphe connexe. En particulier, il existera toujours une route entre un nœud N et sa passerelle tel que tous les nœuds intermédiaires formant la route utilisent le même préfixe que le nœud N et sa passerelle. La continuité de préfixe est une conséquence inhérente au fonctionnement de notre proposition. Nous avons également proposé et évalué différents algorithmes permettant de sélectionner de manière dynamique et suivant différents critères la passerelle la plus appropriée pour un nœud donné. La continuité de préfixe permet en particulier d'organiser un réseau ad hoc (hybride) en sous-graphes connexes, ce qui permet par exemple d'éviter l'utilisation d'en-têtes de routage.

De plus, nous avons également étudié l'applicabilité du concept de continuité de préfixe aux communications multicast dans les réseaux ad hoc hybrides. Nous avons dans un premier temps proposé un certain nombre de règles qui, appliquées à un protocole de routage multicast, permettent de respecter la continuité de préfixe et de contruire des arbres multicast dont tous les membres partagent un même préfixe. Ces arbres sont interconnectés grâce aux passerelles qui connectent chaque sous-réseau ad hoc. Cette approche permet de plus de centraliser la construction des arbres multicast au niveau des passerelles, ce qui permet par exemple d'y effectuer des opérations de management, de contrôle d'accès et de facturation. Un autre avantage de cette architecture est qu'elle interdit de par sa nature même le transit du trafic multicast à l'intérieur du réseau ad hoc : le trafic multicast d'un sous-réseau à un autre doit obligatoirement passer par les passerelles. Cette fonctionnalité est en effet appréciable pour les réseaux ad hoc dont la bande passante et les ressources des nœuds sont limitées. L'architecture que nous avons proposée est de plus parfaitement compatible avec le protocole de routage multicast actuellement déployé dans l'Internet. Nous avons enfin évalué l'impact de la continuité de préfixe sur la structure des arbres multicast respectant ce concept en comparant les caractéristiques de ces arbres avec celles des arbres de plus courts chemins. Nous avons notamment montré que les caractéristiques topologiques globales de ces deux types d'arbres sont très proches et respectent un modèle mathématique présent dans les arbres multicast des réseaux filaires.

Plan du mémoire

Ce mémoire est composé de deux parties.

Dans la première partie nous étudions les communications de groupe avec des participants mobiles dans les réseaux sans-fil à infrastructure. Nous présentons tout d'abord la nouvelle version du protocole IP qui est au cœur de l'Internet Nouvelle Génération. Nous exposons ensuite les protocoles actuels permettant de mettre en œuvre des communications de groupe dans l'Internet, puis présentons les solutions permettant le support des ordinateurs mobiles dans le cadre de communications point à point. Nous étendons ensuite cette étude en présentant les propositions permettant de gérer la mobilité des membres d'une communication multicast. Dans le chapitre suivant, nous présentons et validons l'environnement de simulation utilisé au cours de nos travaux en étudiant en particulier les caractéristiques des arbres multicast résultant de l'utilisation de cet environnement. Nous évaluons ensuite les deux solutions retenues à l'IETF afin de gérer la mobilité des récepteurs d'un groupe multicast. De plus, nous proposons une alternative à ces approches et la comparons avec ces deux propositions. Enfin, nous présentons une solution permettant de gérer la mobilité de la source d'un arbre multicast spécifique à une source. Notre proposition permet en particulier de construire de manière dynamique un arbre de diffusion des plus courts chemins enraciné à la position courante de la source. Nous évaluons notre solution en la comparant à l'alternative évoquée dans les spécifications du protocole de gestion de la mobilité dans l'Internet.

Dans la deuxième partie de ce mémoire, nous étudions les communications de groupe dans les réseaux ad hoc. Dans un premier temps, nous présentons les protocoles de routage unicast et multicast qui ont été spécifiquement développés pour ce type particulier de réseaux. Nous étendons cette présentation en introduisant la problématique de l'autoconfiguration d'adresses dans un réseau ad hoc *hybride* possédant une interconnexion avec l'Internet. Nous présentons en particulier les solutions ayant été proposées afin de réaliser cette autoconfiguration. Nous introduisons alors le concept de *continuité de préfixe* dans un réseau ad hoc et décrivons une solution d'autoconfiguration d'adresses permettant de mettre en œuvre ce concept. Nous validons et évaluons ensuite notre proposition en comparant les différents algorithmes que nous proposons afin de réaliser l'autoconfiguration d'adresses. Enfin, nous étendons le concept de continuité de préfixe aux communications de groupe. Nous proposons une méthode alternative de construction des arbres de diffusion multicast conduisant à des structures composées uniquement par des nœuds partageant tous le même préfixe réseau. Afin d'étudier l'impact de notre proposition, nous analysons les caractéristiques des arbres multicast respectant la continuité de préfixe en les comparant en particulier à celles des arbres de plus courts chemins construits à partir de protocoles classiques de routage multicast.

Nous terminons enfin ce mémoire par une synthèse de nos contributions, puis nous évoquons les perspectives offertes par nos travaux en particulier dans le domaine des réseaux ad hoc. Nous présentons plusieurs pistes de travaux futurs qui permettraient d'étendre de manière substantielle la portée de nos travaux.

Première partie

La mobilité et le multicast dans les réseaux sans-fil à infrastructure

Chapitre 1

Contexte de recherche

1.1 Le protocole Internet Nouvelle Génération

1.1.1 Introduction

L'Internet, qui interconnecte plusieurs millions d'ordinateurs et d'utilisateurs, est le plus grand réseau existant. Pour que ces machines puissent communiquer, il a été nécessaire de définir un certain nombre de standards. Historiquement, l'architecture des machines et des systèmes de communication a été définie dans un modèle de référence : le modèle OSI (Open Systems Interconnection) [1]. Afin de réduire la complexité liée à la conception des systèmes de communication, le modèle OSI décompose les fonctionnalités d'une communication en *couches*. Le rôle de chaque couche est de réaliser un certain nombre d'opérations pour fournir des services à la couche immédiatement supérieure dans le modèle. Les détails techniques d'implémentation sont dissimulés : une couche au niveau n du modèle n'a pas besoin de savoir comment la couche $n-1$ lui fournit ses services. Enfin, la couche n d'une machine communique avec la couche n d'une autre machine, qui peut être un correspondant ou une machine intermédiaire chargée de transmettre les données. En émission, les données transitent des couches supérieures vers les couche inférieures jusqu'à atteindre la couche la plus basse. A la réception, les données sont transmises de la couche la plus basse vers la couche de plus haut niveau.

Bien que le modèle de référence OSI reste d'actualité, l'Internet est basé sur une architecture simplifiée très proche du modèle OSI : le modèle TCP/IP. Dans ce modèle, il n'y a en particulier pas de couches présentation et session. La couche application est donc en relation directe avec la couche transport. Le modèle TCP/IP ainsi que les protocoles associés à chacune de ses couches sont devenus les standards *de facto* des communications dans l'Internet. Ces protocoles sont standardisés par l'IETF (Internet Engineering Task Force) qui est l'organisme chargé de la spécification des modèles de communication dans l'Internet. Ainsi, la couche réseau repose sur la version 4 du protocole IP (Internet Protocol ou IPv4) [2] dont la fonction principale est le routage des données à travers le réseau. Chaque machine est désignée par une adresse IP permettant son identification et sa localisation dans l'Internet. La couche transport est principalement associée à deux protocoles appelés TCP (Transmission Control Protocol) [3] et UDP (User Datagram Protocol) [4]. Le protocole TCP offre aux applications une connexion fiable point à point. L'échange des données est garanti sans pertes tant que la connexion est maintenue. Au contraire, le protocole UDP offre un service non fiable ne garantissant pas la livraison des données.

Le développement de l'Internet induit par la popularité exponentielle de ce réseau est cependant limité par la version actuelle du protocole IP (IP version 4). Les adresses IPv4 sont en effet codées sur 32 bits ce qui permet théoriquement d'adresser 2^{32} machines, soit à peu près 4 milliards. En fait, l'attribution des adresses se fait par réseau : une partie des 32 bits identifie le préfixe d'un réseau, et les bits restant servent à identifier les machines du réseau considéré. En pratique un préfixe est attribué à des organisations telles que des universités, des entreprises ou des opérateurs de télécommunications. Ce découpage limite en pratique le nombre de machines adressables à environ 2 milliards [5] alors qu'on estime en 2003 à plus de 600 millions le nombre d'utilisateurs de l'Internet [6]. De plus, l'attribution des adresses IPv4 dans le monde est très inégale ce qui limite sévèrement le développement de l'Internet dans certains pays. Enfin, les tables de routage des routeurs du maillage central de l'Internet atteignent des tailles très importantes et deviennent de plus en plus difficiles à maintenir. Les adresses ont en effet été attribuées de manière *chaotique* depuis les années 80 ce qui empêche de réduire la taille des tables de routage de manière suffisante en utilisant des techniques d'agrégation [7].

Il était donc devenu impératif de proposer une nouvelle version du protocole IP afin de régler les problèmes liés à l'explosion de la taille des tables de routage et à l'épuisement des adresses disponibles [8]. Un groupe de travail à l'IETF s'est donc attelé à proposer une nouvelle version du protocole IP en profitant de l'expérience acquise au cours de plus de vingt années d'utilisation de IPv4. Ce travail débuté en 1992 a donné lieu à la spécification de la version 6 du protocole IP (IPv6) en vue du déploiement de l'Internet Nouvelle Génération.

1.1.2 Le protocole IPv6

La version 6 du protocole IP (IPv6) est le fruit d'un travail de plusieurs années au sein de l'IETF. Ce protocole a été spécifié pour la première fois en 1995 [9] puis mis à jour en 1998 [10]. Les principales contraintes de son développement étaient les suivantes [11] :

- adresser au moins un milliard de réseaux
- proposer un plan de transition incrémental sans "jour J"
- prendre en compte à terme des services optionnels tels que la mobilité

Le protocole IPv6 propose ainsi l'utilisation d'adresses de 128 bits qui permettent d'adresser plusieurs centaines de milliards de machines [8]. Avec le plan d'adressage agrégé retenu [12], les 64 premiers bits d'une adresse IPv6 sont utilisés pour indiquer le préfixe du réseau, alors que les 64 bits restant identifient la machine (ou plutôt l'interface physique) à laquelle cette adresse est attribuée. Le système d'attribution des adresses a également été modifié pour permettre d'agréger les informations de routage de manière optimale afin de diminuer la taille des tables de routage des routeurs de l'Internet. De plus, un système d'options pouvant être rajoutées à l'en-tête IPv6 des paquets de données permet de mettre en œuvre des extensions, par exemple pour gérer la mobilité des nœuds du réseau. Enfin, une grande force du protocole IPv6 est le développement d'une méthode d'autoconfiguration d'adresses sans état, c'est à dire ne nécessitant pas de serveur centralisé. Enfin, la représentation textuelle d'une adresse IPv6 se fait en découpant le mot de 128 bits en 8 mots de 16 bits séparés par le caractère ":".

Chaque mot est représenté en hexadécimal. Par exemple :

2001 : 0000 : 0000 : 0102 : 0200 : 1cff : fed1 : 5105

De plus, il n'est pas nécessaire d'écrire les zéros placés en tête de chaque nombre héxadécimal :

2001 : 0 : 0 : 102 : 200 : 1cff : fed1 : 5105

Il est également possible de remplacer plusieurs champs nuls consécutifs par le signe "::", qui ne peut apparaître qu'une seule fois dans une adresse (pour éviter toute ambiguïté) :

2001 :: 102 : 200 : 1cff : fed1 : 5105

Enfin, la représentation d'un préfixe réseau IPv6 est similaire à la notation utilisée pour les préfixes IPv4 pour laquelle on indique par le symbole $/l$ la longueur l (en bits) du préfixe :

2001 : 0 : 102 : 200 :: /64

1.1.2.1 L'autoconfiguration d'adresses sans état

Le système d'autoconfiguration sans état est basé sur la notion d'identifiant d'interface. Comme défini précédemment, les 64 bits de poids faible d'une adresse IPv6 sont utilisés pour identifier de manière unique une machine d'un réseau. En fait, ces 64 bits identifient l'interface physique permettant de joindre la machine sur un lien (réseau) auquel elle est connectée. Le plan d'adressage agrégé propose de construire cet identifiant à partir de l'adresse MAC (Medium Access Control) de l'interface, c'est à dire à partir de l'adresse physique de l'interface. En pratique, les adresses MAC de type IEEE 802 ont une longueur de 48 bits et une adresse identifie de manière unique une interface physique. L'identifiant d'interface d'une adresse IPv6 est ainsi basé sur l'utilisation d'un identifiant Ethernet unique (ou EUI-64 [13]) construit à partir de l'adresse MAC IEEE 802 de l'interface physique connectant une machine à un réseau. Pour les communications limitées au lien physique du réseau (c'est à dire sans routage), une station peut ainsi créer automatiquement une adresse *lien-local* à partir d'un préfixe prédéfini d'une longueur de 64 bits (fe80 ::/64). L'unicité de cette adresse sur le lien est alors vérifiée en utilisant le protocole de découverte des voisins de IPv6 [14]. Il est en effet possible pour un utilisateur de modifier l'adresse MAC d'une carte réseau, ce qui peut donc conduire à la duplication d'une adresse MAC. Après avoir créé son adresse lien-local, une station exécute donc une procédure de détection d'adresse dupliquée (Duplicate Address Detection ou DAD). La station envoie un message de sollicitation d'un voisin avec pour cible l'adresse qu'elle vient de créer. Ce message est envoyé avec comme adresse source l'adresse unicast non sollicitée (::) qui est utilisée par les nœuds n'ayant pas encore d'adresse lien-local. Si une machine du réseau utilise déjà l'adresse dont l'unicité est testée, elle répond par l'envoi d'un message d'annonce de voisin qui indique que l'adresse est déjà en service. Elle ne pourra donc pas être attribuée à l'interface : l'autoconfiguration s'arrête et une intervention humaine est nécessaire. Au contraire, si la machine désirant tester l'unicité de son adresse ne reçoit pas de message d'annonce de voisin pendant une seconde (valeur par défaut), elle considère que l'adresse n'est pas déjà attribuée et peut donc l'utiliser.

La configuration automatique sans état des équipements du réseau est l'un des attraits principaux du protocole IPv6. En pratique, ce protocole permet de configurer les machines d'un réseau sans intervention humaine car les adresses MAC IEEE 802 sont uniques. De plus, un identifiant d'interface ne doit être unique que sur le lien du réseau auquel appartient une machine ce qui diminue fortement la probabilité de découvrir une adresse dupliquée (dans le cas d'une utilisation *anormale* des interfaces de type IEEE 802).

1.1.3 Conclusion

Les travaux principaux concernant les spécifications du protocole IPv6 sont maintenant terminés et ce protocole a atteint un degré de maturité suffisant permettant d'envisager son déploiement à grande échelle. Le protocole IPv6 offre un espace d'adressage qui permet d'envisager une durée de vie très longue et des mécanismes d'autoconfiguration (présentés plus en détails dans la section 4.3.1) qui facilitent grandement l'administration des réseaux IPv6. De plus, il existe de nombreux mécanismes permettant la transition incrémentale des réseaux vers IPv6 tout en permettant la communication avec le *monde* IPv4 [15].

IPv6 est donc LE protocole de l'Internet Nouvelle Génération. Son déploiement à grande échelle s'accélère actuellement grâce à des projets divers comme WIDE au Japon, 6NET et Euro6IX en Europe, ou encore au sein du réseau RENATER en France. Cependant, ce déploiement est plus lent que prévu pour diverses raisons [16]. Premièrement, IPv6 est une évolution du protocole IPv4 et non une *révolution* : ses mécanismes de base sont en effet identiques à IPv4 et beaucoup d'utilisateurs restent sceptiques quant à son utilité. En effet, la vaste majorité des adresses IPv4 (66%[6]) a été attribuée aux Etats-Unis qui disposent donc d'une réserve conséquente d'adresses et ne semblent en conséquence pas prêts à investir les sommes nécessaires à la transition vers IPv6. Tant que la pénurie d'adresses n'aura pas touché les Etats-Unis, beaucoup d'opérateurs et d'utilisateurs américains continueront à penser que l'utilisation d'adresses de 128 bits est un luxe dont ils peuvent se passer. Cependant, le déploiement du protocole IPv6 est très actif en Asie et en Europe pour deux raisons. Premièrement, ces deux continents (dont en particulier l'Asie) font face à une pénurie d'adresses alors qu'un pays comme la Chine a un nombre potentiel gigantesque de futurs utilisateurs d'Internet. Deuxièmement, l'indifférence relative des Etats-Unis par rapport au déploiement de IPv6 donne une opportunité unique à l'Europe et à l'Asie de prendre une avance substantielle dans le développement de solutions techniques liées à IPv6.

Enfin, l'apparition de nouveaux usages de l'Internet tels que la mobilité et les communications de groupe accompagnent le déploiement du protocole IPv6. La multiplication des réseaux d'accès sans-fil induit en effet une forte croissance de la population d'utilisateurs mobiles. De plus, l'augmentation très rapide des débits des réseaux rend possible le développement d'applications multimédia telles que la télévision et la radio par Internet. Ces applications sont particulièrement adaptées à l'utilisation de communications multicast qui permettent de mettre en communication un grand nombre d'utilisateurs.

1.2 Les communications multicast avec IP version 6

Le modèle IP multicast est une extension du protocole IP qui a été proposée originellement par S. Deering [17]. Ce modèle (le *modèle de Deering*) introduit en particulier la notion de *groupe* de machines en proposant l'utilisation d'adresses IP dites *de groupe*. Un groupe est un ensemble de stations identifiées par une adresse IP unique. La composition d'un groupe est dynamique : une station peut rejoindre ou quitter un groupe à tout moment et il n'y a aucune restriction quant à la taille d'un groupe et à la localisation de ses membres. De plus le modèle de Deering n'impose pas à une station de faire partie d'un groupe pour pouvoir lui envoyer des données : le modèle de groupe est *ouvert*. L'application de ce modèle à un réseau à médium partagé est simple. Une adresse IP de groupe est associée à une adresse physique comme par exemple une adresse Ethernet [18]. Lorsqu'une machine désire adhérer à un groupe donné, une procédure particulière permet de notifier la couche réseau qu'une application désire recevoir les données à destination de cette adresse de groupe. Cette procédure déclenche en particulier la reconfiguration de l'interface physique afin de pouvoir recevoir les paquets envoyés à destination de la paire *adresse IP de groupe - adresse physique associée*.

Les complications liées au routage multicast apparaissent lorsque tous les membres d'un groupe n'appartiennent pas au même réseau physique. Ceci nécessite en effet de propager des paquets multicast à travers des routeurs. Il devient donc nécessaire de mettre en place des mécanismes permettant d'identifier les réseaux d'appartenance des membres d'un groupe multicast, et de propager ces informations afin de mettre en place le routage multicast. Avec la version 6 du protocole IP, les mécanismes permettant à une station d'adhérer ou de quitter un groupe multicast sont définis par le protocole MLD. Le protocole de routage multicast actuellement déployé dans l'Internet Nouvelle Génération est présenté dans la section 1.2.2. Nous rappelons également quels protocoles de routage multicast ont été déployés par le passé.

1.2.1 MLD : Multicast Listener Discovery

L'objectif du protocole MLD [19] est de permettre à tout routeur IPv6 de détecter la présence de stations désirant adhérer à un groupe multicast. En particulier, chaque routeur maintient pour chacune de ses interfaces physiques une liste des groupes multicast auxquels ont adhéré les stations directement connectées au lien physique. Cette information est ensuite utilisée par le protocole de routage multicast afin d'assurer le routage des paquets multicast entre les différents réseaux physiques ayant des membres pour un groupe donné.

Le protocole MLD est *asymétrique*, c'est à dire que son comportement est différent selon que la station exécutant ce protocole est un routeur ou une simple machine du réseau. Pour chaque lien physique, un mécanisme permet de sélectionner un routeur particulier, le *querier*, parmi tous les routeurs du lien. Ce routeur émet périodiquement une requête d'appartenance (*MLD query*) à destination de l'adresse multicast de tous les nœuds du lien (ff02::1). A la réception d'un tel message, une station répond sur la même interface en émettant un rapport d'appartenance (*MLD report*) pour chaque groupe auquel elle participe. Afin d'éviter l'envoi de réponses simultanées, une station attend un délai aléatoire avant d'émettre un rapport. Si pendant ce délai, une autre station émet le même rapport d'appartenance, l'envoi du rapport en attente de transmission est annulé. En effet, un routeur multicast n'a pas besoin de connaître tous les membres d'un groupe pour un lien donné : il lui suffit de savoir qu'il y a un membre pour propager les paquets multicast sur le lien concerné. Enfin, lorsqu'une station ne souhaite plus recevoir les données pour un groupe particulier, elle informe le routeur désigné en envoyant un message de retrait (*MLD done*). A la réception d'un tel message, le routeur désigné émet une requête d'appartenance spécifique (*MLD specific query*) à destination de l'adresse du groupe spécifié afin de vérifier si il y a encore des membres pour ce groupe. Si aucun rapport d'appartenance n'est envoyé pendant un court laps de temps, le protocole de routage multicast est informé qu'il n'y a plus de membres pour le groupe sur ce lien.

Dans sa version 2 [20], le protocole MLD a été étendu afin de permettre à une station de spécifier dans ses messages d'appartenance de quelle(s) source(s) multicast elle désire recevoir des paquets multicast pour un groupe donné. Ces extensions ont en particulier conduit à modifier le comportement des routeurs et des hôtes afin de permettre le filtrage des sources. En ce qui concerne le routage multicast, le protocole PIM-SM (Protocol Independent Multicast - Sparse Mode) s'est imposé pour le routage des données multicast dans les réseaux IPv6.

1.2.2 Routage multicast : le protocole PIM-SM

Le routage multicast consiste à mettre en place dans les routeurs les états de propagation nécessaires au transport des paquets multicast lorsque les membres d'un groupe donné sont dans différents réseaux physiques.

Une première approche du routage multicast est le routage en *mode dense* utilisé historiquement par le protocole DVMRP [21] et réutilisé par le protcole PIM-DM [22]. Il repose sur un procédé d'inondation/élagage. Ainsi, les données multicast sont propagées dans tout le réseau, puis les routeurs feuilles propagent vers la source des messages d'élagage si ils n'ont aucun membre sur toutes leurs interfaces filles. Les informations d'élagage doivent être rafraîchies périodiquement ce qui nécessite une inondation fréquente du réseau. Ce type de protocole souffre ainsi d'un problème de mise à l'échelle dû à l'inondation périodique du réseau.

Le protocole MOSPF [23] utilise les informations topologiques du protocole OSPF [24] afin de construire des arbres de plus courts chemins. Des extensions permettent en particulier d'enrichir la base de données maintenue par OSPF avec des informations d'appartenance aux groupes multicast. Chaque routeur est donc capable de calculer l'arbre des plus courts chemins permettant de connecter tous les membres d'un groupe. Cette méthode a l'avantage d'éviter l'inondation périodique du réseau des solutions précédentes. L'inconvénient majeur de MOSPF est cependant son besoin de puissance lié à la complexité du calcul des arbres multicast. De plus, cette approche peut conduire à certaines incohérences si tous les routeurs n'ont pas la même vision du réseau et de l'appartenance aux groupes à un instant donné.

Les difficultés liées aux propositions précédentes ont conduit à l'adoption d'une troisième approche basée sur le concept d'adhésion explicite et sur l'utilisation d'arbres partagés. Les protocoles CBT [25] et PIM-SM [26] sont ainsi basés sur ce fonctionnement. En pratique, PIM-SM [27] a été déployé avec succès à relativement grande échelle, et c'est de plus le seul protocole disponible pour l'Internet Nouvelle Génération : c'est actuellement le seul protocole de routage multicast utilisé avec IPv6. Avec PIM-SM, un routeur est sélectionné de manière automatique pour jouer le rôle de *point de rendez-vous* (ou RP pour RendezVous Point). Dans un domaine PIM, le RP est le point de rencontre entre les sources et les récepteurs. Lorsqu'un récepteur désire adhérer à un groupe, son routeur désigné va envoyer un message d'adhésion explicite (*PIM Join*) en direction du RP afin d'adhérer à l'arbre partagé du groupe noté (*,G) (arbre commun à toutes les sources). Ce message se propage de saut-en-saut vers le RP, et sa propagation s'arrête dès qu'il rencontre un routeur faisant déjà partie de l'arbre de diffusion partagé. En parallèle, le routeur désigné d'une source S encapsule à destination du RP (dans des messages *PIM register*) les paquets multicast envoyés par la source S. Le RP construit alors la première branche d'un arbre par source noté (S,G) en envoyant un message d'adhésion de saut-en-saut vers la source. Les données peuvent alors être propagées en multicast entre la source et le RP. Les données multicast sont ensuite propagées par le RP dans l'arbre partagé. Lorsqu'un routeur membre de l'arbre partagé commence à recevoir des données, il initie la construction d'une branche vers l'arbre enraciné à la source afin de permettre la réception des données par le plus court chemin entre les réseaux de la source et du récepteur. Tous ces mécanismes sont illustrés par la figure 1.1.

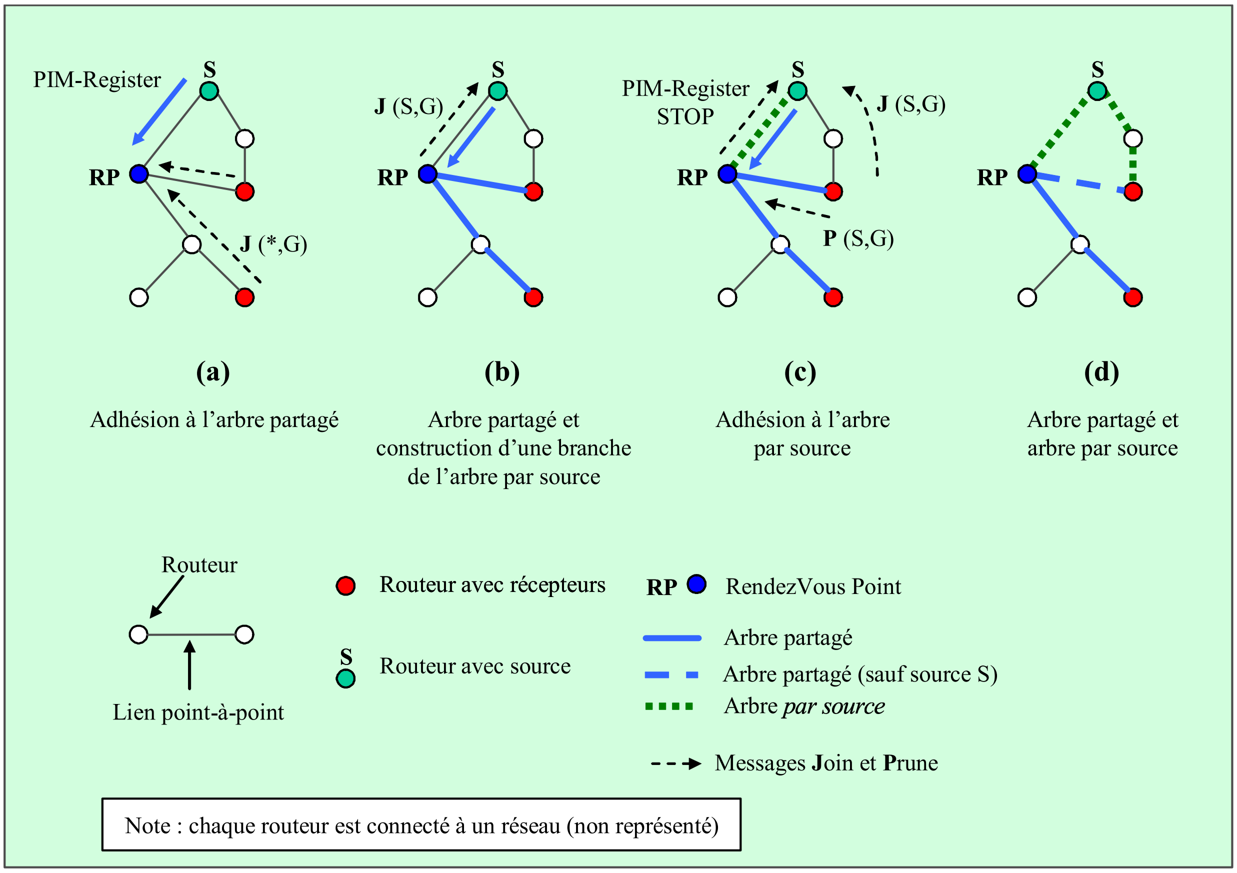

FIG. 1.1 – Les mécanismes d'adhésion dans PIM-SM

Bien que le protocole PIM-SM ait connu un vif succès tant au niveau de la mise en œuvre que du déploiement, ce modèle rencontre un certain nombre de difficultés. En pratique, le déploiement s'effectue en effet à l'intérieur d'un *domaine PIM* qui regroupe des réseaux dont les opérateurs ont décidé de collaborer afin, par exemple, de décider où et par qui sont hébergés les points de rendez-vous. Avec la version 4 du protocole IP, ce déploiement nécessite l'utilisation supplémentaire du protocole MSDP [28] afin de permettre aux différents domaines PIM de partager les informations concernant les sources actives. De plus, le modèle de Deering étant *ouvert*, il est en pratique difficile d'effectuer un contrôle d'accès au niveau des sources d'un groupe. Cette limitation empêche actuellement les opérateurs de proposer un service multicast *grand public*. Enfin, le problème lié à l'attribution des adresses de groupe n'a pas encore trouvé de solution satisfaisante à grande échelle. Ces inconvénients ont conduit à la proposition d'un modèle simplifié : le multicast spécifique à une source.

1.2.3 PIM-SSM : Source Specific Multicast

PIM-SSM trouve son origine dans le protocole EXPRESS [29] pour EXPlicitly REquested Single-Source. EXPRESS introduit en particulier la notion de canal multicast. Un canal noté (S,G) est composé d'une adresse de groupe (G) et de l'adresse S de la source des données multicast. Dans ce modèle, seule la source peut envoyer des données à destination de G : il s'applique donc tout particulièrement aux applications de type *1 vers N*, dans lesquelles une station unique émet vers un ensemble de récepteurs. On citera par exemple la télévision et la radio par Internet. Le modèle introduit par le protocole EXPRESS a conduit à la spécification de PIM-SSM [30] qui est une extension du protocole PIM-SM. Pour IPv6, une plage d'adresses multicast a été réservée (ff3x::8000:0000/101) et peut être utilisée librement par n'importe quelle source.

Les avantages de ce modèle sont nombreux et ils permettent d'envisager le déploiement de cette solution à l'échelle de l'Internet. Tout d'abord, l'attribution des adresses de groupe d'un canal SSM devient un problème local à chaque source. Les *collisions* d'adresses de groupe sont donc évitées. De plus, la complexité introduite par le modèle à arbre partagé est supprimée. En effet, PIM-SSM ne nécessite ni points de rendez-vous, ni protocole pour la découverte des sources entre domaines PIM. Enfin, la nature même de ce modèle introduit un certain niveau de sécurité et de filtrage puisque seule la source d'un canal peut émettre des données. En pratique, la mise en place d'un arbre uni-directionnel permet de renforcer ce contrôle d'accès en empêchant à un récepteur d'usurper l'adresse de la source. Dans ce cas de figure, les données envoyées avec une adresse usurpée ne seront pas re-transmises par le routeur désigné de l'émetteur car elles n'arrivent pas de l'interface qui mène à la source.

1.2.4 Conclusion sur les communications multicast

Bien que le routage multicast ait été l'objet de nombreux travaux depuis une vingtaine d'années [31], un seul protocole de routage multicast est actuellement déployé à grande échelle dans l'Internet Nouvelle Génération [32] : le protocole PIM-SM et sa variante PIM-SSM. L'évolution et le déploiement du multicast dans l'Internet ont été relativement lents au cours des années 90 [33] mais l'apparition de nouvelles technologies d'accès comme l'ADSL (Asymmetric Digital Subscriber Line) [34, 35] a récemment relancé l'intérêt des opérateurs pour le multicast. Avec l'augmentation des débits, il devient en effet possible de proposer à l'utilisateur final des services multimédia attractifs comme la télévision et la radio par Internet. Le protocole PIM-SSM est particulièrement adapté à ce type de services dans lesquels une source émet des données vers un grand nombre de récepteurs potentiels [32]. Ce protocole ne souffre pas de problème de mise à l'échelle et ne nécessite pas la mise en œuvre d'une infrastructure complexe pour le routage multicast inter-domaines.

De plus, on assiste également à une multiplication des technologies d'accès sans-fil comme IEEE 802.11 [36] et Bluetooth [37]. La popularité de ces équipements a conduit à un déploiement très rapide de réseaux sans-fil [38] et cet engouement se poursuit grâce à l'augmentation des débits proposés par les nouvelles versions du protocole 802.11 (54 Mb/s pour 802.11a et 802.11g). La conséquence directe du déploiement croissant de réseaux sans-fil est la multiplication des utilisateurs mobiles. Il devient ainsi nécessaire de pouvoir gérer la mobilité des utilisateurs, aussi bien au niveau de la couche liaison de données que de la couche réseau (IP).

Ces constatations ont conduit à la proposition de protocoles de gestion de la mobilité dans l'Internet Nouvelle Génération.

1.3 La mobilité dans l'Internet Nouvelle Génération

La démocratisation des technologies de communication sans-fil et leur soudaine popularité a entraîné un vif intérêt pour ce domaine auprès des acteurs de la recherche dans l'Internet. Les technologies *grand public* telles que IEEE 802.11 et Bluetooth connaissent en effet un vif succès tant au niveau des entreprises que des universités et des particuliers. Cet engouement s'est traduit par la volonté de proposer des protocoles permettant de gérer la mobilité des utilisateurs au niveau de l'Internet. L'IETF, l'organisme de standardisation des protocoles de l'Internet, a ainsi proposé un certain nombre de protocoles permettant de prendre en charge les déplacements des utilisateurs mobiles en assurant en particulier le maintien des communications en cours. Ces réflexions ont conduit à la création de deux groupes de travail ayant pour objectifs de proposer des mécanismes génériques permettant de gérer la mobilité au niveau IP : Mobility for IPv4 (MIP4) et Mobility for IPv6 (MIP6). La première version du protocole Mobile IPv4 a ainsi vu le jour en 1996 [39], la version actuelle datant de 2002 [40]. Dans le même temps, la version 6 du protocole IP a été définie en 1995 [9].

C'est à cette même époque qu'ont également démarré les spécifications du protocole Mobile IPv6. Mobile IPv6 utilise les nouvelles fonctionnalités définies dans IPv6 afin de simplifier et de faciliter la gestion de la mobilité des équipements. Après près de dix ans d'efforts, une première spécification de Mobile IPv6 vient enfin de voir le jour [41]. Toutefois, ce protocole de gestion de la mobilité dans l'Internet Nouvelle Génération repose sur de nombreuses années d'expérimentation et de validation de l'ensemble de ses mécanismes. Ce protocole hérite en effet des fonctionnalités du protocole Mobile IPv4 et des nouvelles extensions du protocole IPv6. On peut ainsi affirmer que Mobile IPv6 sera certainement le premier protocole de gestion de la mobilité déployé à terme à grande échelle dans l'Internet Nouvelle Génération. Nous présentons cependant une alternative à la gestion de la mobilité qui est basée sur l'utilisation d'un adressage logique indépendant de la localisation topologique. Cette approche utilise des mécanismes très proches de ceux utilisés dans Mobile IP et apparaît donc comme une alternative viable au protocole Mobile IPv6.

1.3.1 Le protocole Mobile IPv6

Le protocole Mobile IPv6 a pour principal objectif de permettre à un équipement mobile de changer de point d'attachement tout en maintenant ses connexions au niveau de la couche transport. Les protocoles de transport UDP et TCP identifient en effet une communication à l'aide d'un quadruplet composé des adresses et ports source et destination des participants d'une communication. Ces deux protocoles de la couche transport ayant été définis bien avant le développement des équipements mobiles, ils font l'hypothèse que les adresses source et destination d'une communication restent identiques au cours du temps. Or dans l'Internet les adresses IP sont étroitement liées à la position géographique des équipements : une adresse IP sert à identifier un nœud ainsi que sa localisation puisque chaque adresse contient le préfixe du réseau dans lequel elle est utilisée. Un hôte mobile devra donc acquérir une adresse topologiquement correcte à chaque fois qu'il changera de point d'attachement et de sous-réseau lors de ses déplacements. Cette contrainte implique que l'adresse courante d'un nœud ne peut pas être utilisée comme un identifiant invariant dans le temps.

Le protocole Mobile IPv6 propose de résoudre ce problème en assignant une adresse IP unique à tout équipement mobile : *l'adresse mère*. Cette adresse est utilisée afin d'identifier (en partie) de manière permanente une communication au niveau de la couche transport. L'adresse mère d'un hôte mobile est également liée au concept de *réseau mère* : il s'agit du réseau d'appartenance de l'hôte mobile, c'est à dire son réseau d'origine. L'adresse mère d'une station mobile est définie comme étant l'adresse IP de cet équipement lorsqu'il est dans son réseau mère. Les différentes phases de la gestion de la mobilité par Mobile IPv6 sont illustrées par la figure 1.2.

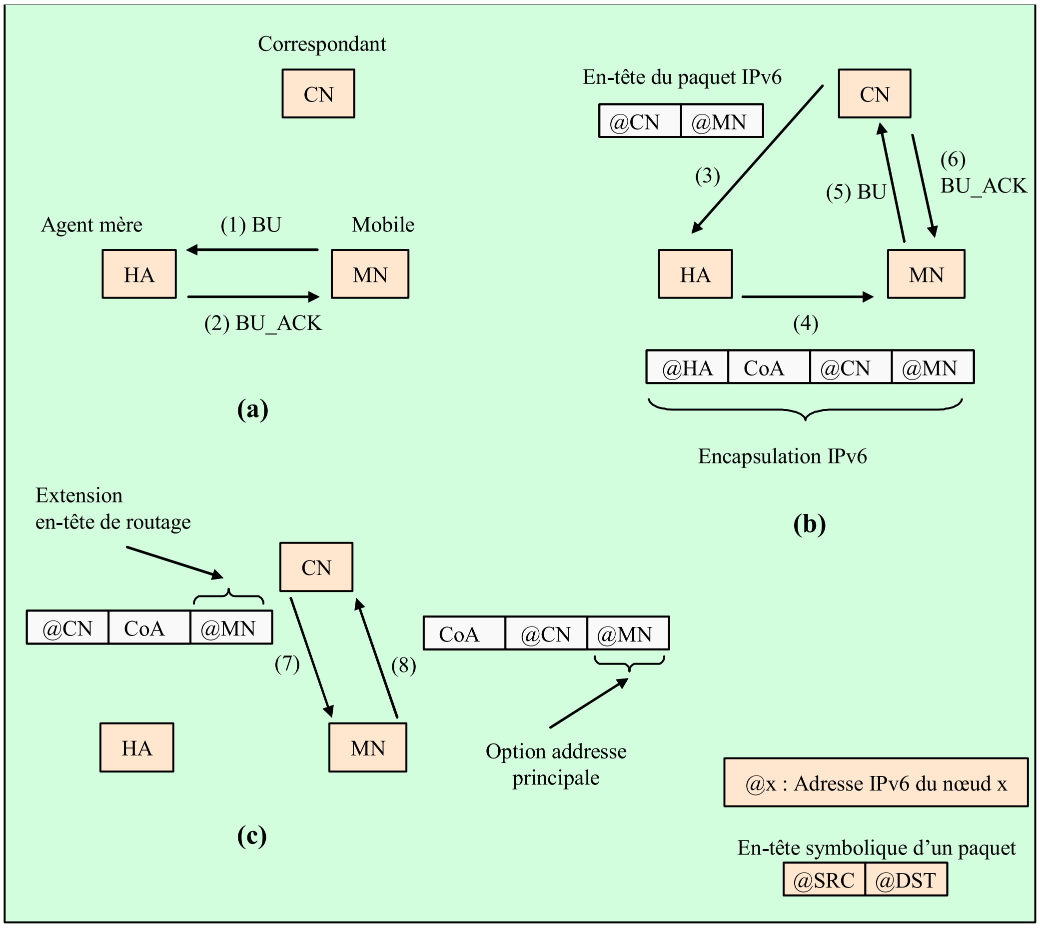

FIG. 1.2 – Les mécanismes de gestion de la mobilité IPv6

Avec Mobile IPv6, lorsqu'un hôte mobile se déplace dans un réseau visité, il acquiert une *adresse temporaire* (ou CoA pour Care-of-Address) à l'aide des mécanismes d'autoconfiguration d'IPv6 [42]. La création de cette adresse déclenche l'envoi par le nœud mobile d'un message de *mise à jour de l'association* (ou BU pour Binding Update) à son agent mère. A la réception d'un tel message, l'agent mère d'un hôte mobile maintient une association entre l'adresse mère du mobile et son adresse temporaire. Cette phase est illustrée par la figure 1.2 (a). Lorsqu'un correspondant désire communiquer avec l'hôte mobile, il envoie ses paquets à destination de l'adresse mère (@MN) du nœud mobile car c'est l'adresse enregistrée par les serveurs de noms de l'Internet ([43]). Or l'agent mère sait que le mobile n'est pas joignable dans le réseau mère. Il intercepte donc les paquets à destination de l'adresse mère du mobile et les encapsule vers la position courante du mobile comme représenté par la figure 1.2 (b). Un mécanisme optionel permet également au mobile d'indiquer sa position courante au correspondant. Si le correspondant supporte les fonctionnalités de Mobile IPv6, il peut alors envoyer directement ses données vers la position courante du mobile en utilisant un en-tête de routage. De même, le mobile peut envoyer directement ses données à son correspondant en utilisant une option destination *adresse mère*. Ces deux optimisations sont illustrées par la figure 1.2 (c). Elles nécessitent néamoins la mise en place d'associations de sécurité entre l'hôte mobile, ses correspondants, et son agent mère [41].

1.3.2 Utilisation d'un adressage logique

Le problème principal de la mobilité dans l'Internet est qu'une adresse IP ne peut pas être utilisée comme identifiant unique invariant pour un nœud mobile. Alors que le protocole Mobile IPv6 propose l'utilisation d'une option *adresse mère* afin d'identifier de manière unique et permanente une station mobile, d'autres propositions se basent sur la mise en place d'un adressage logique indépendant de la localisation d'un nœud dans le réseau. Cette approche a été proposée initialement dans les spécifications du protocole VIP [44] qui reposait sur une modification du modèle en couche classique, et qui a conduit à la proposition du protocole LINA [45] qui propose notamment d'intégrer l'identifiant unique d'un nœud dans l'adresse topologique servant à le localiser. Cette approche a donné lieu à la spécification du protocole LIN6 [46] qui est l'application de LINA pour IPv6. Ce protocole apparaît comme une alternative viable au protocole Mobile IPv6 et fait en particulier toujours l'objet de travaux au sein de l'IETF [47]. On citera également le protocole LAR [48] qui est présenté dans la section 1.4.5 pour sa gestion unifiée des communications unicast et multicast avec des participants mobiles. Enfin, on peut mentionner le protocole I3 [49] qui utilise des mécanismes proches de ceux proposés par LAR.

1.3.2.1 Le protocole LIN6

LIN6 (Location Independent Network Architecture for IPv6) propose de séparer les identificateurs logiques des localisateurs physiques des nœuds dans l'Internet, et ceci sans utiliser une option d'en-tête supplémentaire comme le fait Mobile IPv6. Le protocole LIN6 est basé sur le concept d'identifiant LIN6 unique à chaque nœud. Cet identifiant de 64 bits appelé LIN6-ID identifie de manière unique et permanente toute machine de l'Internet. En pratique, le LIN6-ID est construit comme un identifiant unique Ethernet (EUI-64) en utilisant un préfixe de 24 bits (OUI pour Organizationally Unique Identifier) assigné à LIN6 (02-0C-21). Afin de rester compatible avec l'adressage sur 128 bits de IPv6, le LIN6-ID est concaténé à un préfixe LIN6 de 64 bits afin de créer une adresse de 128 bits : l'adresse LIN6 généralisée. Le préfixe LIN6 est assigné au protocole LIN6 et n'est associé à aucun réseau de l'Internet : il n'identifie donc aucun réseau en particulier et n'est pas routable dans l'Internet. Les adresses LIN6 généralisées sont en effet utilisées par la couche transport uniquement afin d'identifier les communications : ces adresses sont invariantes et identifient une machine quelle que soit sa localisation dans l'Internet.

L'association entre l'adresse LIN6 généralisée d'une station mobile et son adresse dans un réseau visité est enregistrée par chaque nœud mobile auprès d'agents de localisation, ou *mapping agents (MA)*. Lorsqu'un mobile se déplace, il crée une adresse topologiquement correcte d'une manière similaire au protocole Mobile IPv6. La seule différence avec LIN6 est que le LIN6-ID est utilisé au lieu de l'EUI-64. L'adresse LIN6 ainsi obtenue est donc la concaténation du préfixe du réseau visité (64 bits) et du LIN6-ID. Une fois cette adresse obtenue, une station mobile enregistre l'association entre l'adresse LIN6 généralisée et l'adresse LIN6 courante (le localisateur) auprès d'un MA. De plus, chaque station mobile est enregistrée dans le système de noms de domaine (DNS [43]) avec une adresse particulière : elle est formée par le préfixe réseau du MA de ce mobile et le LIN6-ID de la station mobile. Lorsqu'un correspondant veut communiquer avec un hôte mobile, il fait une requête DNS et obtient donc l'identifiant unique du mobile (le LIN6-ID) ainsi que le préfixe du MA qui enregistre l'association entre l'adresse généralisée et l'adresse courante du mobile. Le correspondant peut alors reconstruire l'adresse généralisée du mobile (préfixe LIN6 + LIN6-ID), et peut également envoyer une requête au MA afin d'obtenir l'adresse courante du mobile. L'adresse du MA est dérivée en concaténant le préfixe obtenu lors de la requête DNS avec un identifiant particulier appelé IFID. Cette procédure de communication est résumée par la figure 1.3.

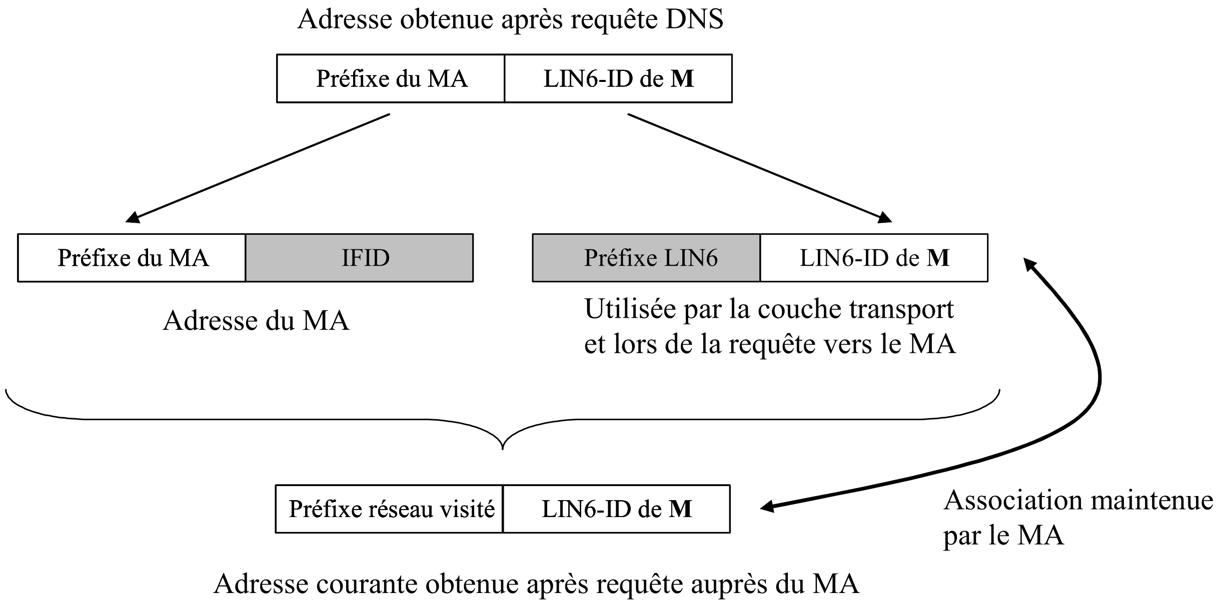

FIG. 1.3 – Résolution de l'adresse d'un mobile M avec LIN6

1.3.3 Comparaison entre Mobile IPv6 et LIN6

Il y a de nombreux parallèles entre LIN6 et Mobile IPv6. Ainsi, l'agent de localisation de LIN6 (le MA) maintient l'association entre l'identifiant du mobile et son adresse temporaire comme le fait l'agent mère avec Mobile IPv6. Ces deux entités sont également utilisées comme nœud relais lorsque le correspondant d'une station mobile ne supporte pas la mobilité, c'est à dire lorsqu'une optimisation de route n'est pas possible. Les données sont alors encapsulées entre le mobile et l'agent mère (Mobile IPv6), ou entre le mobile et le MA (LIN6). Cependant et ce malgré leur ressemblance, il susbsiste un grand nombre de différences entre ces deux protocoles.

Premièrement, avec LIN6 un nœud mobile peut enregistrer son association auprès de plusieurs agents de localisation, ce qui implique donc plusieurs enregistrements dans le système de résolution de noms de domaine. Cette particularité est avantageuse car elle permet d'éviter le problème du point central de panne lié à l'enregistrement unique auprès de l'agent mère avec Mobile IPv6. Cette optimisation introduit cependant une surcharge supplémentaire car une station mobile devra enregistrer son association auprès de plusieurs agents de localisation. Une alternative non résolue [46] consiste à mettre en œuvre un mécanisme de synchronisation entre les agents de localisation afin de pouvoir réaliser l'enregistrement de l'association auprès d'un unique agent de localisation.

La seconde différence majeure entre LIN6 et Mobile IPv6 est que LIN6 ne nécessite pas l'utilisation d'un en-tête de routage et d'une option destination *adresse mère* lorsque l'optimisation de route est possible. Contrairement à LIN6, Mobile IPv6 ne maintient en effet aucun état (l'association entre l'adresse mère et l'adresse temporaire) dans les correspondants. LIN6 se démarque ainsi de la *philosophie* classique de l'Internet qui consiste à ne pas maintenir d'états dans les correspondants. L'avantage de cette méthode est l'élimination de la surcharge induite par l'utilisation d'en-tête de routage et d'option destination. Le principal inconvénient de cette approche est que les communications reposent entièrement sur la capacité des nœuds à maintenir ces états : toute communication repose en effet sur un contexte qui doit être maintenu sous peine de rendre impossible tout échange de données au niveau de la couche transport.

En conclusion, ces deux protocoles ont des fonctionnalités intéressantes qui en font des candidats potentiels pour la gestion de la mobilité dans l'Internet Nouvelle Génération. Le protocole Mobile IPv6 a cependant eu la préférence au sein de l'IETF et dans le monde de la recherche. On peut donc sérieusement penser que Mobile IPv6 sera le premier protocole de gestion de la mobilité déployé à grande échelle dans l'Internet.

1.3.4 Conclusion

La gestion de la mobilité dans l'Internet Nouvelle Génération est devenue incontournable. En particulier, le protocole Mobile IPv6 a atteint une certaine maturité qui permet d'envisager son déploiement à grande échelle dans les prochaines années. Dans le même temps, on assiste à un regain d'intérêt autour du routage multicast suite à la popularité croissante de services multimédia comme la télévision et la radio par Internet. Or, il est naturel de penser que les utilisateurs mobiles veulent avoir accès à la même palette de services disponible à partir de réseaux filaires. Il devient donc indispensable de pouvoir proposer un routage multicast adapté à la mobilité des participants d'une communication de groupe. Cette constatation a conduit à la proposition de diverses solutions permettant de gérer la mobilité des participants d'une communication de groupe. Dans la section suivante nous présentons les solutions les plus pertinentes et décrivons les mécanismes les plus intéressants permettant la gestion de membres mobiles au sein d'une communication multicast.

1.4 Les communications multicast avec des participants mobiles

Bien que les protocoles de routage multicast aient été conçus afin de gérer l'adhésion et le retrait dynamique des membres d'un groupe multicast, ils ne sont pas optimisés pour la gestion de membres mobiles. On pourrait penser que les deux problèmes sont identiques et que la mobilité peut être traitée de la même façon que l'adhésion et le retrait dynamique de membres. En fait, la principale différence réside dans la continuité de service qu'il faut pouvoir offrir suite à la mobilité de certains membres. On voudra ainsi minimiser l'interruption de la réception d'un flux multicast à laquelle doit faire face un mobile après un déplacement. De plus, lorqu'une station mobile est source de trafic multicast, il devient crucial de permettre la reprise immédiate de la diffusion des données après un changement de point d'attachement de la source. Ce problème est exacerbé si l'on considère un arbre par source enraciné à la position courante d'une station mobile.

Il existe plusieurs propositions dont l'objectif est de permettre une gestion optimisée des déplacements d'une station mobile participant à une communication de groupe. Nous présentons les deux propositions retenues actuellement à l'IETF, et décrivons ensuite des solutions cherchant à optimiser les mécanismes permettant d'acheminer du trafic multicast de et vers des stations mobiles.

1.4.1 La mobilité et le multicast avec Mobile IPv6

Le problème introduit par la participation de stations mobiles à des communications multicast a été globalement étudié dans le cadre du protocole Mobile IP. Ces réflexions ont conduit assez rapidement à l'adoption de deux propositions simples destinées à permettre à des hôtes mobiles de prendre part à des communications de groupe. Ces deux propositions, l'utilisation d'un *tunnel bi-directionnel* et l'*adhésion depuis le réseau visité*, ont été à la fois décrites pour Mobile IPv4 [50] et Mobile IPv6 [51]. Elles sont ainsi évoquées dans les spécifications actuelles du protocole Mobile IPv6 [41] et illustrées par la figure 1.4.

L'adhésion depuis le réseau visité est une approche extrêmement simple, puisque le mobile se comporte comme n'importe quel hôte statique du réseau visité. L'adhésion se fait en envoyant un message d'appartenance (*MLD report*) à partir de la position courante du mobile. La particularité induite par la mobilité est que cette adhésion doit être renouvelée après chacun des déplacements du membre mobile. Ceci nécessite la mise en place d'un mécanisme permettant aux deux piles protocolaires que sont MLD et Mobile IPv6 de communiquer afin d'effectuer une ré-adhésion immédiate après le déplacement d'un mobile [52]. Un inconvénient de cette solution est la latence induite par chaque ré-adhésion après le déplacement d'un membre mobile : pendant cette phase les données multicast ne sont pas délivrées à la position courante du mobile. Enfin, le principal désavantage de cette proposition est qu'elle ne permet pas de gérer les déplacements de la source d'un canal SSM si celle-ci est mobile. Dans le cas d'un arbre partagé, le déplacement d'une source est moins pénalisant car les données sont envoyées au point de rendez-vous. Cependant, la source apparaîtra commme une nouvelle source auprès des récepteurs après chaque déplacement. Un mécanisme récent permet néanmoins d'éviter ce problème [53].

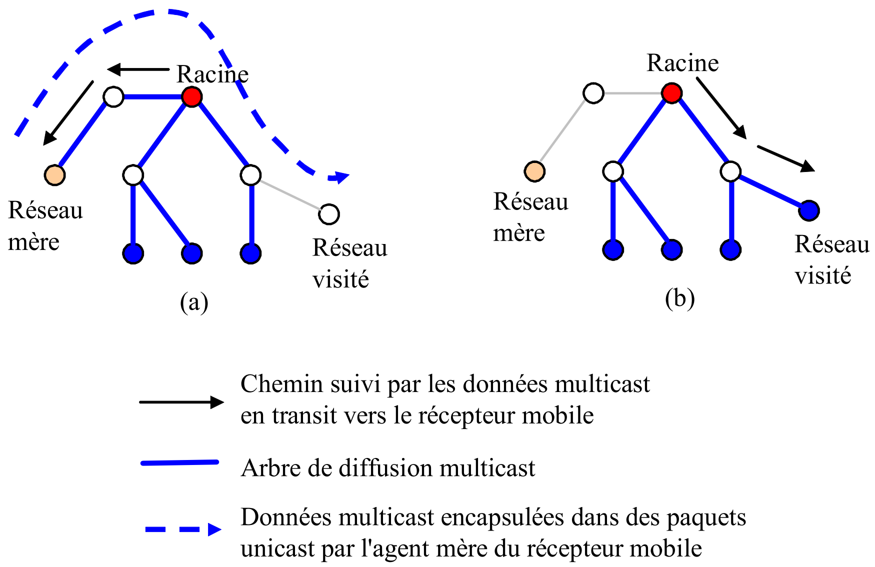

FIG. 1.4 – Tunnel bi-directionnel (a) et adhésion depuis le réseau visité (b)

Une alternative est l'utilisation d'un tunnel bi-directionnel entre un nœud mobile et son agent mère. La station mobile émet et reçoit toutes les données multicast à travers ce tunnel. De même, tous les messages MLD d'adhésion et de retrait sont envoyés par le mobile dans le tunnel à destination de son agent mère. Réciproquement, l'agent mère envoie les requêtes MLD (*queries*) dans le tunnel à destination de l'hôte mobile. Le principal avantage de cette approche est que les déplacements de la station mobile n'ont pas de répercussion sur le protocole de routage multicast. Chaque déplacement n'induit en effet qu'une mise à jour de l'une des extrémités du tunnel. Cette approche se heurte cependant à deux inconvénients majeurs. Premièrement, elle induit un routage triangulaire non optimal entre l'agent mère et la station mobile. Les données transitent en effet des sources vers l'agent mère, puis de l'agent mère jusqu'au membre mobile. Le deuxième problème lié à cette solution est connu sous le terme de *convergence des tunnels*. Si un réseau héberge plusieurs stations mobiles membres d'un même groupe multicast, plusieurs copies des mêmes données seront en effet envoyées vers ce réseau au travers de plusieurs tunnels. L'utilisation du multicast perd alors de son intérêt.

Le tableau 1.1 résume les avantages et les inconvénients de chacune des deux solutions envisagées au sein de l'IETF.

1.4.2 Mobile Multicast (MoM) Protocol

Le protocole MoM [54] a été défini afin de proposer une solution au problème de convergence des tunnels. Ce protocole est uniquement applicable en l'état avec Mobile IPv4 car il repose sur l'utilisation des agents relais (ou *foreign agent*) qui n'ont pas d'équivalents dans Mobile IPv6. On peut néanmoins envisager son utilisation avec Mobile IPv6 si les agents mère de différents réseaux peuvent collaborer. MoM propose une solution si plusieurs agents mère envoient, à travers plusieurs tunnels IP, le même trafic multicast vers un réseau dans lequel résident plusieurs stations mobiles ayant adhéré au même groupe. Avec Mobile IPv4, l'une des extrémités du tunnel mis en place entre un agent mère et un nœud mobile correspond en

	Avantages	Inconvénients
Adhésion depuis le réseau visité	- Meilleure résistance à la mise à l'échelle - Routage optimal	- Latence induite par chaque ré-adhésion - Nécessite le support du multicast dans le réseau visité
Utilisation d'un tunnel bi-directionnel	- Nécessite uniquement la mise à jour du tunnel - Autorise la réception du multicast dans un réseau ne proposant pas ce service	- Introduit un routage triangulaire - Augmente la charge de traitement pour l'agent mère - Mise à l'échelle limitée - Problème de convergence des tunnels

TAB. 1.1 – Avantages et inconvénients des solutions retenues dans Mobile IPv6

fait à l'agent relais du réseau visité par le mobile. Lorsque qu'un agent mère encapsule des données multicast *à destination* du mobile, le paquet encapsulé est en fait envoyé à l'agent relais qui retransmet les données multicast nativement sur le réseau visité. L'agent relais peut donc détecter l'arrivée de multiples copies de paquets multicast destinés à un même groupe multicast. Le protocole MoM propose que dans un tel cas, l'agent relais sélectionne un agent mère unique qui aura la charge de retransmettre les données multicast. Cet agent mère particulier est nommé DMSP pour Designated Multicast Service Provider. Les agents mère qui ne sont pas DMSP doivent stopper la retransmission des données multicast à destination du groupe et du réseau concernés. Le fonctionnement du protocole MoM est illustré par la figure 1.5.

Bien que cette proposition permettre de résoudre le problème de convergence de tunnels, plusieurs points restent en suspens. Premièrement, lorsqu'une station mobile dont l'agent mère a été choisi comme DMSP quitte le réseau visité courant, l'envoi du trafic multicast vers ce réseau visité est interrompu car l'agent mère va faire suivre les données multicast vers la nouvelle position du mobile. L'agent relais de l'ancien réseau visité doit donc choisir un nouveau DMSP parmi les agents mère des stations mobiles résidant toujours dans ce réseau. En pratique avec Mobile IPv4, l'agent relais n'est pas informé du départ d'une station mobile et la procédure de choix d'un nouveau DMSP sera donc déclenchée par l'expiration d'un temporisateur. En conséquence la diffusion du trafic multicast risque d'être interrompue pendant un laps de temps non négligeable [55]. De plus, MoM propose plusieurs méthodes de sélection du DMSP, mais en pratique cette procédure n'est pas triviale : dans tous les cas, le routage multicast reste triangulaire puisque les données transitent toujours via le DMSP. Enfin, les problèmes de sécurité et de relation de confiance entre les acteurs du protocole MoM, ainsi que la gestion des sources mobiles ne sont pas abordés dans cette proposition. Une extension du protocole MoM nommée RBMoM a fait suite à la publication du protocole MoM.

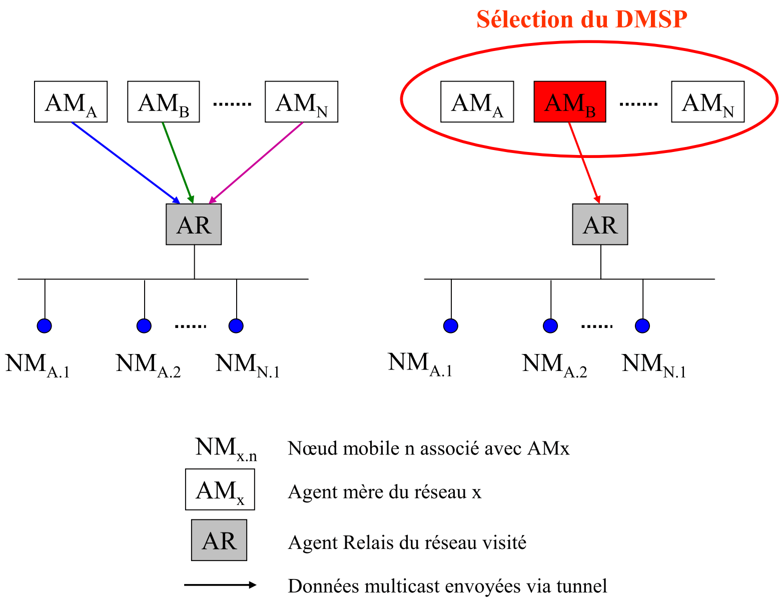

FIG. 1.5 – Le protocole MoM

1.4.3 RBMoM : une méthode généralisée

La particularité du protocole RBMoM [56] (Range-Based Mobile Multicast) est qu'il regroupe de manière subtile les propositions précédemment présentées : MoM, l'utilisation d'un tunnel bi-directionnel, et l'adhésion depuis le réseau visité. L'objectif principal de RBMoM est de proposer un compromis entre routage optimal (adhésion depuis le réseau visité) et surcharge engendrée par la reconstruction des arbres multicast (problème inexistant dans le cas du tunnel bi-directionnel). Comme dans MoM, le protocole RBMoM propose l'utilisation d'une entité particulière, le MHA pour Multicast Home Agent, qui a la charge de retransmettre les données multicast vers les nœuds mobiles. RBMoM introduit également la notion de portée (*range*) d'un MHA, c'est à dire la distance maximale (en nombre de sauts) entre un MHA et un mobile vers lequel le MHA retransmet des données multicast.

Lorsqu'une station mobile est intéressée par un groupe multicast, elle contacte son MHA qui va adhérer au groupe et lui encapsuler les données multicast. Lorsque le mobile se déplace, et tant qu'il reste à une distance inférieure à la valeur de la portée du MHA, son MHA reste inchangé. Cela induit un routage triangulaire proportionnellement borné par la valeur de la portée du MHA. Si le mobile se déplace dans un réseau visité à une distance supérieure à la valeur de la portée du MHA, le mobile doit changer de MHA et sélectionne l'agent relais du réseau visité comme nouveau MHA. Le routage triangulaire est alors supprimé jusqu'au prochain déplacement du mobile. De plus, pour éviter le problème de convergence des tunnels, chaque station mobile enregistre auprès de l'agent relais de son réseau visité l'adresse de son MHA et la liste des groupes multicast auxquels le MHA a adhéré. Comme avec MoM, l'agent relais peut alors choisir pour chaque groupe multicast un DMSP parmi les MHA. Il est intéressant de noter que pour les valeurs extrêmes de la portée, le protocole RBMoM se confond avec l'adhésion depuis le réseau visité et l'utilisation du tunnel bi-directionnel (à condition que tous les MHA utilisent la même valeur de portée). En effet, si la portée $R = 0$, alors à chaque déplacement la station mobile choisit comme MHA l'agent relais du réseau visité courant, et RBMoM devient identique à l'adhésion depuis le réseau visité. Au contraire, si la portée $R = \infty$, et comme le MHA initial d'un hôte mobile est son agent mère (avant tout déplacement), le mobile ne sera jamais au cours de ses déplacements à une distance supérieure à la portée de son MHA. Le MHA restera donc toujours l'agent mère du mobile, et dans ce cas RBMoM est identique à la solution du tunnel bi-directionnel par l'agent mère. Le protocole RBMoM est illustré par la figure 1.6.

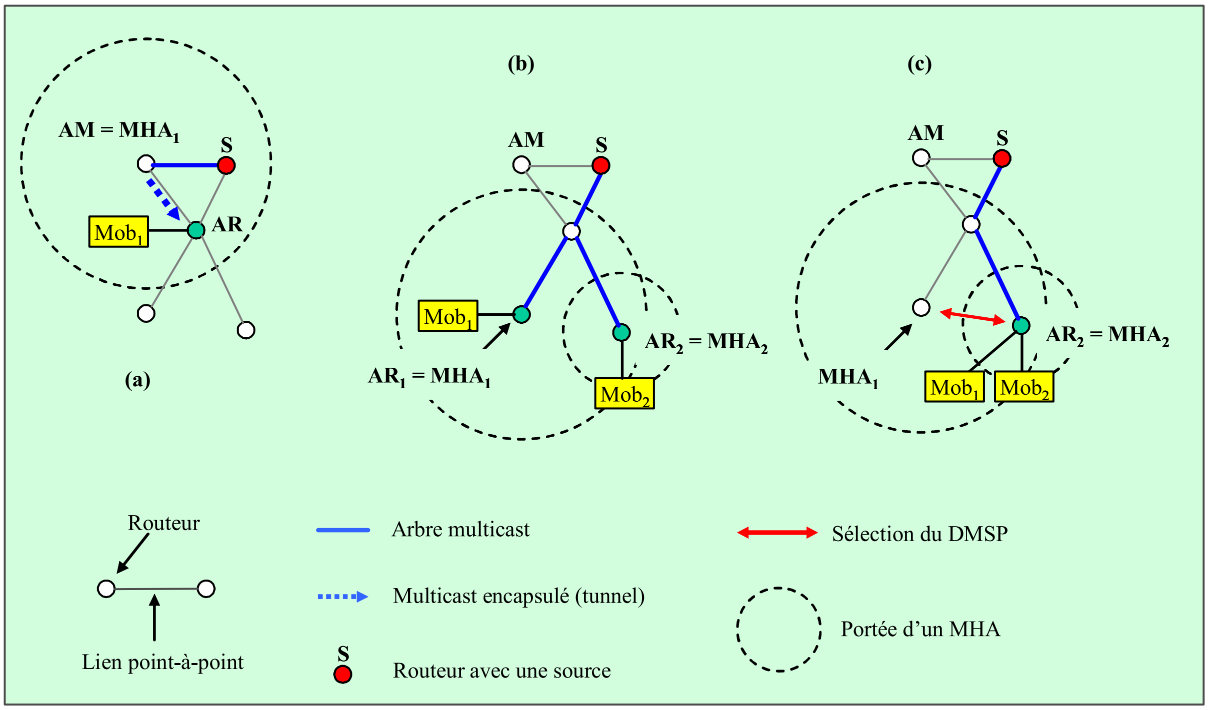

FIG. 1.6 – Le protocole RBMoM

Un mobile noté $\mathbf{Mob}_1$ sélectionne son agent mère (**AM**) comme MHA (noté $\mathbf{MHA}_1$). Les données sont envoyées de la source **S** vers $\mathbf{MHA}_1$ puis encapsulées vers la position courante du mobile (partie (a) de la figure 1.6). En (b), le mobile $\mathbf{Mob}_1$ se déplace dans un nouveau réseau hors de portée de l'agent mère. Le mobile $\mathbf{Mob}_1$ choisit donc comme nouveau MHA l'agent relais ($\mathbf{AR}_1$) du réseau visité. Dans le même temps, un second mobile $\mathbf{Mob}_2$ sélectionne l'agent relais ($\mathbf{AR}_2$) comme MHA. En (c), le mobile $\mathbf{Mob}_1$ se déplace dans le même réseau que $\mathbf{Mob}_2$, tout en restant dans la portée de $\mathbf{MHA}_1$. Il y a donc sélection du DMSP : RBMoM préconise de sélectionner le MHA le plus proche, c'est à dire $\mathbf{MHA}_2$.

L'avantage principal de RBMoM est d'unifier les solutions que sont MoM, l'utilisation d'un tunnel bi-directionnel et l'adhésion depuis le réseau visité. Ce protocole se heurte malheureusement à plusieurs difficultés pratiques liées principalement à des questions de sécurité et de relation de confiance entre les stations mobiles et les agents relais.

On citera également le protocole MMA [57] qui est quasiment identique à RBMoM avec une portée infinie et une procédure de sélection du DMSP différente. De plus, l'objectif de RBMoM qui est de trouver un compromis entre routage optimal et surcharge provoquée par la reconstruction des arbres multicast est également à la base d'une proposition visant à combiner routage triangulaire et adhésion depuis le réseau visité.

1.4.4 Routage triangulaire temporaire

La solution proposée par *Tseng et al.* [58] vise à tirer parti des avantages du routage triangulaire et de l'adhésion depuis le réseau visité. Cette proposition s'inspire fortement des protocoles MoM et RBMoM. Avec cette solution, une station mobile commence par adhérer à un groupe depuis le réseau visité tout en informant l'agent relais de cette adhésion. Lorsque le nœud mobile se déplace dans un nouveau réseau, il envoie un message particulier à l'agent relais de l'ancien réseau visité afin de lui demander de retransmettre temporairement le flux multicast encapsulé vers la nouvelle position du mobile. Cette technique utilise le fait que le flux multicast continue à arriver dans l'ancien réseau visité après le déplacement de l'hôte mobile. La durée pendant laquelle le trafic multicast continue à être délivré dans l'ancien réseau visité est déterminée par les temporisateurs utilisés dans le protocole d'adhésion au groupe (IGMP [59] pour IPv4 et MLD [19] pour IPv6). Dans le même temps, le mobile lance une procédure d'adhésion depuis le nouveau réseau visité afin de pouvoir bénéficier d'un routage optimal des données. Lorsque l'adhésion depuis le réseau visité est effectuée, la station mobile informe son ancien agent relais d'arrêter de retransmettre les données encapsulées. Une option permet également à un mobile, lors de son adhésion, de demander à son nouvel agent relais de maintenir l'abonnement pendant un certain laps de temps si le mobile se déplace. Cette option garantit ainsi que le mobile pourra recevoir, après son déplacement, le trafic multicast pendant un laps de temps assez long pour permettre l'adhésion depuis le nouveau réseau visité.

Cette proposition a l'avantage de tirer partie du routage optimal offert par l'adhésion depuis le réseau visité et de l'interruption de service minimale offerte par l'utilisation d'un tunnel. Une variante a également été proposée. Le protocole JWT [60] propose de différer l'adhésion depuis le nouveau réseau visité pendant une durée déterminée, ceci afin d'éviter de lancer une adhésion lorsqu'une station mobile a un déplacement rapide et donc une présence très courte dans un réseau visité. L'objectif est d'éviter la reconstruction inutile d'une branche vers l'arbre multicast. Cependant, le gain obtenu par cette restriction est difficilement quantifiable.

Les trois dernières approches que nous présentons n'utilisent pas le modèle multicast tel qu'il a été defini initialement par Deering. Ces propositions utilisent en effet des modèles multicast différents basés sur l'utilisation de messages unicast.

1.4.5 LAR : une architecture d'adressage et de routage logique

Le protocole LAR [48] (Logical Addressing and Routing) propose un système d'adressage et de routage logique permettant d'unifier les communications unicast et multicast avec des participants mobiles ou fixes [61]. L'architecture LAR se décompose en deux parties : un gestionnaire de communication présent uniquement sur les nœuds terminaux, et une couche LAR intercalée entre la couche transport et la couche réseau. La couche LAR est présente à la fois dans les nœuds terminaux et les routeurs intermédiaires, mais elle n'est pas obligatoire en ce qui concerne les routeurs intermédiaires. Cette couche supplémentaire se traduit notamment par l'utilisation d'adresses LAR. Une adresse destination LAR peut correspondre à une machine ou à une communication (de groupe par exemple) : elle est dans les deux cas indépendante de la localisation au niveau réseau. Une adresse source LAR correspond toujours à une machine.

Une communication LAR met en œuvre une connectivité logique, au dessus de la couche IP, entre un ensemble particulier de correspondants. L'accès à une communication LAR est contrôlé dynamiquement par le gestionnaire de communication assigné à chaque communication (par exemple l'initiateur d'une communication). En fait, l'ensemble des partenaires d'une communication est modifié par des opérations d'adhésions et de retraits. Ce type de communication est donc particulièrement adapté aux échanges entre des correspondants qui doivent être identifiés logiquement, c'est à dire indépendamment de leur adresse réseau courante. Le concept d'adressage et de routage logique proposé par LAR est donc particulièrement adapté aux communications de groupe et au support de la mobilité IP.

Le routage repose sur la construction d'un arbre LAR qui connecte les participants d'une communication LAR. Cet arbre est mis à jour par des messages d'adhésions, de retraits, et de déplacements envoyés par les acteurs d'une communication LAR. Les adresses LAR étant logiques et situées entre la couche transport et la couche réseau, les changements d'adresses réseau occasionnés par les déplacements de stations mobiles sont transparents pour les applications car les adresses LAR sont invariantes. Une communication LAR repose sur l'utilisation d'une table de commutation et d'un cache LAR. La table de commutation permet à chaque membre d'un arbre LAR de connaître l'adresse LAR de ses voisins dans l'arbre, et le cache LAR permet de faire l'association entre les adresses LAR et les adresses IP.

La mise en œuvre de l'architecture LAR dans une communication de groupe avec un participant mobile est illustrée par les figures 1.7 et 1.8. L'adresse LAR d'une machine x est notée @Lx, et son adresse réseau @Rx. L'arbre comprend quatre membres : deux récepteurs fixes *r1* et *r2*, un récepteur mobile m et une source fixe s. On peut noter au passage qu'avec LAR un nœud peut être indistinctement source ou récepteur. On peut voir sur la figure 1.7 qu'une table de commutation indique à chaque membre de l'arbre LAR les adresses LAR de ses voisins dans l'arbre (le nœud lui-même est également présent dans la table). Par exemple le routeur J a trois voisins dont les adresses LAR sont @Lr1, @Lr2 et @Lm. Le cache LAR du routeur J permet d'associer les adresses LAR avec les adresses réseau des voisins de J dans l'arbre LAR.

Lorsque la source s émet un datagramme multicast à destination de l'adresse LAR de groupe @LG, la couche LAR va encapsuler ce datagramme dans un paquet IP qui sera envoyé à destination de l'adresse réseau @RO (le routeur O est le seul voisin de s dans l'arbre LAR). Lorsque O va recevoir ce paquet, la couche LAR va faire le lien entre l'adresse destination @LG et les voisins de O dans l'arbre LAR à l'aide de sa table de commutation. Une copie des données sera donc envoyée à destination des adresses réseau @Rm et @RJ. Si le nœud mobile m se déplace comme illustré par la figure 1.8, l'envoi d'un message LAR_HELLO de m à O permettra de mettre à jour le cache LAR du routeur O avec la nouvelle adresse réseau de m. Le cas de mobilité illustré par la figure 1.8 est relativement simple dans le sens où il induit une simple mise à jour de l'adresse réseau de m dans le cache LAR du routeur O. Cette mise à jour est suffisante pour obtenir un nouvel arbre LAR optimal. Dans d'autres cas, le déplacement d'un membre d'un arbre LAR peut néanmoins nécessiter la reconfiguration d'une partie de l'arbre afin d'éviter de construire un routage non-optimal. Sans mécanismes de reconfiguration, les déplacements de stations mobiles aboutiraient en effet rapidement à un routage multicast sans rapport avec les positions courantes des nœuds mobiles. Un tel exemple est illustré par la figure 1.9.

Le mobile m s'est déplacé depuis son ancienne position vers le réseau de la source s. Dans

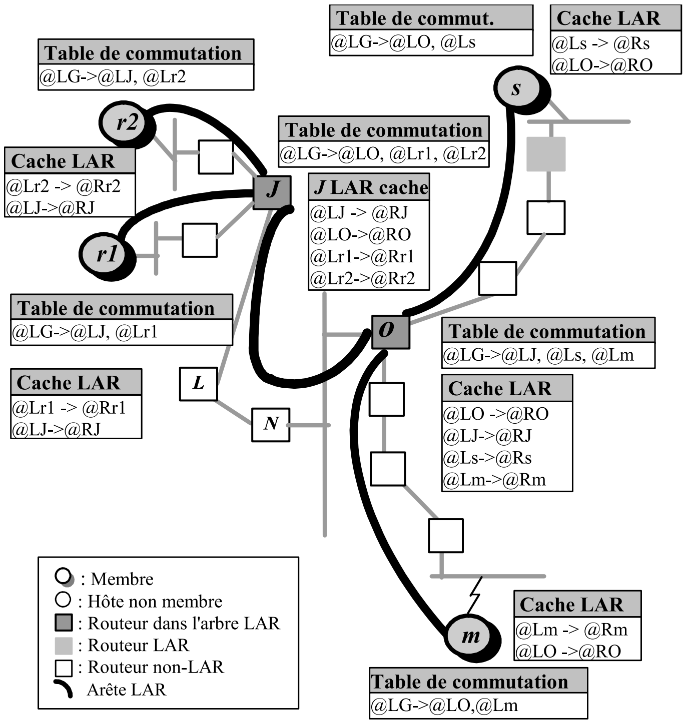

Fig. 1.7 – Une communication multicast avec LAR

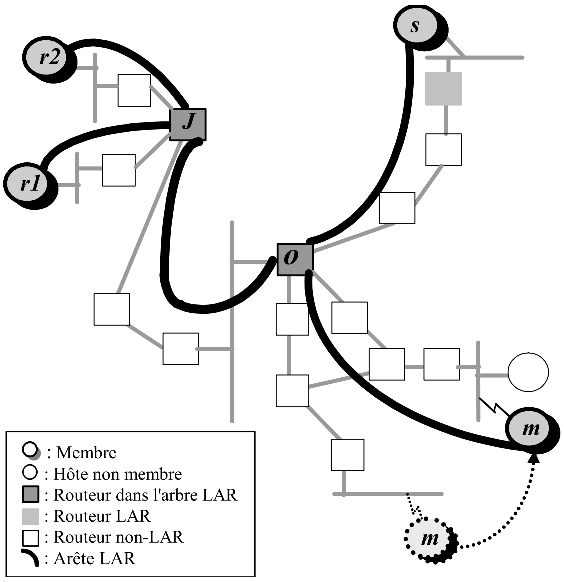

Fig. 1.8 – Déplacement d'un mobile dans une communication multicast avec LAR

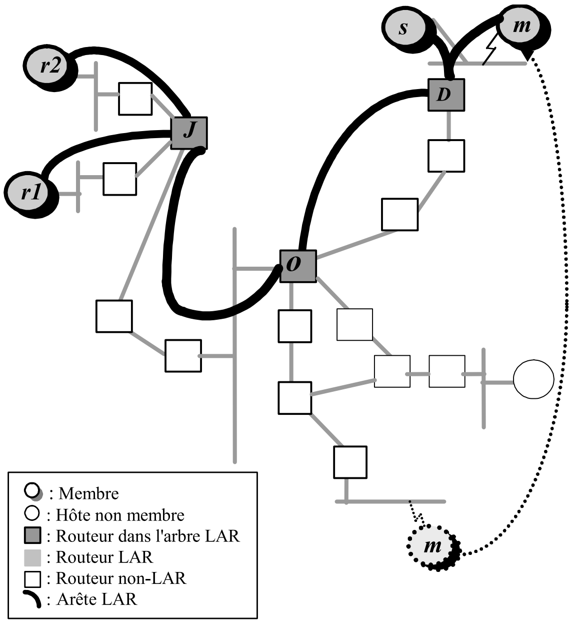

FIG. 1.9 – Fusion d'arêtes dans un arbre LAR

un premier temps, m envoie un message LAR_HELLO au routeur O afin qu'il mette à jour l'adresse réseau de m dans son cache LAR. Le routeur O détecte alors qu'il y a superposition de deux arêtes LAR : Om et Os. La détection de cette superposition déclenche la recherche d'un point de duplication en direction des deux membres s et m. Le protocole LAR supporte de telles fonctionnalités (messages LAR_DUP_SEARCH et LAR_UPDATE) qui vont permettre au routeur O de découvrir un nouveau point de duplication qui offrira un routage optimal vers la nouvelle position du mobile. Ce nouveau point de duplication, le nœud D, reconfigure l'arbre LAR en segmentant l'arête logique Os en deux arêtes : OD et Ds. Il reste alors à greffer l'arête Dm. On notera que cette reconfiguration est uniquement possible car il existe un routeur LAR par lequel transitent les deux arêtes superposées Os et Om.

L'architecture LAR a de nombreux avantages. Elle permet dans un premier temps de construire des arbres de diffusion optimisés sans avoir recours à un algorithme centralisé [62]. Ceci est valable avec des participants mobiles et fixes. De plus, le fonctionnement de l'architecture LAR ne nécessite pas que tous les routeurs soient modifiés pour supporter ce protocole. Un déploiement incrémental de LAR est donc possible. Enfin, cette architecture permet d'unifier les communications unicast et multicast et intègre de façon native la mobilité des participants d'une communication.

D'un autre côté, l'architecture LAR nécessite la mise en place d'une couche supplémentaire dans le modèle TCP/IP ainsi que la mise à jour des piles de communication des machines désirant utiliser cette architecture. Ces contraintes ont conduit à la proposition d'approches relativement proches de LAR mais qui ne nécessitent pas la mise en œuvre d'une couche d'adressage logique supplémentaire.

1.4.6 Le routage multicast de proche-en-proche avec des mobiles

Le routage multicast dit de *proche-en-proche* a été proposé afin de résoudre les problèmes liés au déploiement du multicast utilisant le modèle de Deering [63]. Le premier protocole de ce type est REUNITE [64] pour REcursive UNIcast TreE qui se démarque du modèle de Deering en utilisant uniquement des adresses unicast. Cette approche a été reprise par les protocoles Hop-by-Hop (HBH) [65] et XCast [66]. Ces deux protocoles ont ensuite été étendus pour supporter les membres mobiles ce qui a donné lieu à la proposition des protocoles Mobile-HBH (M-HBH) [67] et E^2MMP [68].

1.4.6.1 MHBH : principes et fonctionnement

Le protocole HBH est une variante de REUNITE qui simplifie la mise en œuvre du routage multicast de proche-en-proche. L'idée de base de HBH est l'utilisation d'une approche *récursive* utilisant uniquement des adresses unicast. Avec cette approche, les routeurs situés à des branchements de l'arbre de diffusion multicast sont responsables de la duplication des paquets de données : ils doivent renvoyer une copie des données à chacun de leurs fils dans l'arbre. Contrairement au routage multicast utilisant le modèle de Deering, la copie des données envoyée par un routeur HBH est transportée dans un paquet unicast. Chaque routeur de branchement envoie donc une copie des données avec autant de paquets unicast que le routeur a de fils dans l'arbre. De plus et contrairement à REUNITE, HBH construit les arbres de diffusion *dans le sens des données*, c'est à dire à partir d'une source vers les récepteurs. On notera également que HBH construit des arbres par source comme les protocoles EXPRESS et PIM-SSM.

Le protocole HBH utilise trois types de messages pour construire, maintenir et optimiser un arbre de diffusion : les messages *join*, *tree*, et *fusion*. De plus, ce protocole nécessite l'utilisation de deux tables qui permettent de maintenir les informations de retransmission : la MCT (Multicast Control Table) et la MFT (Multicast Forwarding Table). La table MCT est utilisée par les routeurs ayant un seul fils dans l'arbre, alors que la table MFT est utilisée lorsqu'un routeur doit retransmettre plusieurs copies des données à destination de ses fils dans l'arbre de diffusion. Les messages *join* sont périodiquement envoyés par les récepteurs en direction de la source du groupe afin de rafraîchir les entrées de retransmission contenues dans les tables MFT. Les routeurs de branchement dans l'arbre peuvent *intercepter* ces messages *join* et les combiner afin d'optimiser la diffusion dans l'arbre. La source émet elle-aussi des messages périodiques, les messages *tree*. Ces messages servent à mettre à jour la structure complète de l'arbre et servent également à détecter la superposition d'arêtes dans l'arbre. En cas de superposition d'arêtes, un message *fusion* est envoyé en direction de la source afin d'optimiser la distribution des données dans l'arbre. Tous ces mécanismes sont illustrés par un exemple représenté par la figure 1.10.

Dans cet exemple, l'on considère que la route entre S et r_2 est asymétrique : $S \rightarrow H_4 \rightarrow r_2$ et $r_2 \rightarrow H_3 \rightarrow H_1 \rightarrow S$. La route entre S et r_1 est elle-aussi asymétrique : $S \rightarrow H_1 \rightarrow H_3 \rightarrow r_1$ et $r_1 \rightarrow H_2 \rightarrow H_1 \rightarrow S$. La route entre S et r_3 est elle considérée comme étant symétrique : $S \leftrightarrow H_1 \leftrightarrow H_3 \leftrightarrow r_3$. Dans une première phase (a), le récepteur r_1 envoie un message *join* en direction de la source S. Une particularité du protocole HBH est qu'il impose que le premier message *join* envoyé par un récepteur doit obligatoirement atteindre la source et ne doit donc pas être intercepté par un routeur de l'arbre. Le message envoyé par r_1 traverse les routeurs H_2 et H_1 mais il ne crée aucun état dans les tables MCT et MFT car l'arbre est construit dans le sens des données : de la source vers les récepteurs. Lorsque la source S reçoit le message *join* envoyé par r_1 elle émet un message *tree* à destination de r_1. Ce message crée des états dans la table

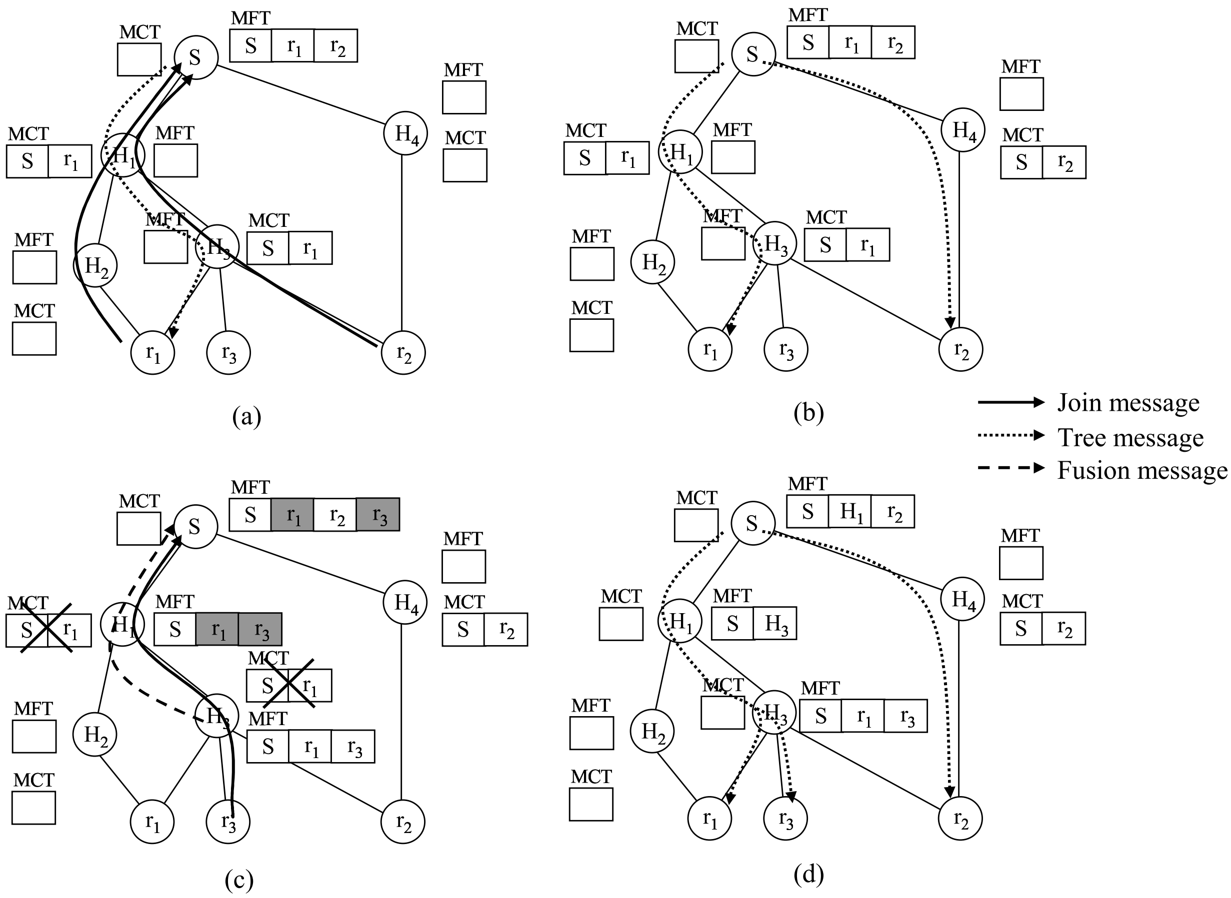

FIG. 1.10 – Mécanismes de construction d'un arbre avec HBH

MCT des routeurs intermédiaires H_1 et H_3. Le récepteur r_2 envoie ensuite un message *join* à destination de S : ce message n'est pas intercepté par les routeurs intermédiaires de l'arbre (H_3 et H_2) car le premier message *join* envoyé par un récepteur ne doit pas être intercepté. A la réception de ce message, la source S envoie un message *tree* à destination de r_2. Cette seconde phase (b) crée les états adéquats dans la table MCT du routeur H_4. On notera que dans les deux cas les branches de l'arbre ont bien été créées de la source vers les récepteurs. Dans la troisième phase (c), le récepteur r_3 envoie un message *join* à destination de la source ce qui déclenche l'envoi d'un message *tree* par la source. Le routeur H_1 détecte alors qu'il est sur le chemin des messages *tree* envoyés vers r_1 et de ceux envoyés vers r_3. Afin d'optimiser la diffusion dans l'arbre, le routeur H_1 envoie un message *fusion* à destination de son père dans l'arbre, c'est à dire S. A la réception du message *fusion*, la source S supprime les entrées pour r_1 et r_3 dans sa table MFT et les remplace par une unique entrée H_1. De même, le routeur H_3 envoie un message *fusion* à destination de H_1 ce qui remplace les entrées r_1 et r_3 par H_3. L'arbre final est illustré en (d).

Le protocole HBH a ensuite été étendu afin de gérer la présence de récepteurs et de sources mobiles. Cette extension est nommée Mobile-HBH (MHBH) [67]. M-HBH introduit dans un premier temps deux nouvelles entités : l'arbre de contrôle et l'arbre des données. L'arbre de contrôle se compose des routeurs échangeant des messages M-HBH alors que l'arbre de diffusion est composé des routeurs retransmettant les données. Lorsqu'il n'y a aucun participant mobile, ces deux arbres sont identiques. De plus, M-HBH se base sur les mécanismes de Mobile IP afin de faire le lien entre l'adresse mère d'un mobile et son adresse temporaire dans un réseau visité. Ainsi, chaque routeur M-HBH maintient notamment l'association entre l'adresse mère et l'adresse temporaire de la source de l'arbre auquel il appartient.

Les mécanismes de gestion de la mobilité dans M-HBH utilisent dans une première phase un tunnel bi-directionnel entre chaque station mobile et son agent mère. Lorsque la source d'un arbre M-HBH se déplace, elle utilise un tunnel bi-directionnel vers son agent mère afin de continuer à envoyer les messages *tree* dans l'arbre de contrôle via son agent mère. Réciproquement, les récepteurs continuent à envoyer les messages *join* à destination de l'adresse mère S de la source. Ces messages sont interceptés par l'agent mère de la source et encapsulés à destination de l'adresse temporaire S' de celle-ci. L'arbre de contrôle utilisé pour échanger les messages M-HBH reste donc identique. Pour l'envoi des données, la source n'utilise par contre pas le tunnel bi-directionnel mis en place avec son agent mère. Les données sont en effet simplement envoyées à destination des nœuds présents dans la table MFT de la source, c'est à dire à destination des routeurs fils de la source dans l'arbre des données. Ces données sont cependant envoyées avec l'adresse temporaire S' de la source alors que les routeurs de l'arbre de données et les récepteurs attendent des paquets envoyés avec comme adresse source S. Afin de permettre à ces derniers de faire le lien entre l'adresse mère S de la source et son adresse temporaire S', la source rajoute à tous les paquets de données une option destination IPv6 *adresse mère* positionnée à S. Cependant, le routage n'est pas optimal car l'arbre a été construit à partir du réseau mère de la source alors que celle-ci s'est déplacée. Pour optimiser le routage, les récepteurs initient la reconstruction des arbres de contrôle et de données en envoyant des messages *join* vers la nouvelle position de la source. Les anciens arbres sont graduellement supprimés lorsque les états présents dans les routeurs expirent. Enfin, la mobilité des récepteurs est gérée de manière similaire à celle des sources. Un tunnel bi-directionnel est dans un premier temps mis en place entre le récepteur et son agent mère : il est utilisé pour relayer les messages de contrôle (*join* et *tree*). Une nouvelle option destination *adresse temporaire* proposée par M-HBH vise de plus à permettre la ré-adhésion du récepteur à partir de sa nouvelle position courante. L'option est en particulier utilisée par les routeurs de l'arbre afin de trouver le point duplicateur optimal permettant d'atteindre le récepteur mobile dans le nouveau réseau visité.

Le principal avantage du protocole M-HBH est qu'il permet de gérer de manière équivalente et quasiment similaire la mobilité des sources et des récepteurs. De plus, l'influence néfaste du routage triangulaire est minimisée dans un premier temps car la source émet directement les données vers ses routeurs fils dans l'arbre sans avoir à passer par un tunnel via l'agent mère. Un mécanisme de reconstruction permet également d'optimiser les arbres de contrôle et des données en initiant la ré-adhésion des récepteurs vers la nouvelle position de la source. La gestion de la mobilité des récepteurs se base sur des mécanismes similaires. Le protocole M-HBH souffre cependant du principal désavantage introduit par le protocole HBH. En effet, alors que l'objectif principal de HBH est de permettre le déploiement de communications de groupe sans avoir à mettre en œuvre une infrastructure de routage multicast [65], ce protocole nécessite l'implantation de HBH dans les routeurs de l'Internet. En effet, si tous les routeurs n'implémentent pas HBH, le routage des arbres de diffusion créés par ce protocole n'est pas optimal car les branchements ne pourront pas être faits aux endroits topologiques les plus stratégiques. De plus, le protocole HBH n'évite pas le maintien de tables de forwarding dans les routeurs de l'arbre de diffusion. Il nécessite également la mise en œuvre de mécanismes liés à la construction, à la maintenance et à l'optimisation des arbres de diffusion. Ces contraintes ont eu pour principale conséquence de motiver la proposition d'une méthode alternative basée sur l'utilisation d'une liste explicite des destinataires.

1.4.6.2 E²MMP : utilisation d'une liste explicite des destinataires

Le protocole E²MMP (Efficient Explicit Mobile Multicast routing Protocol) [68] est basé sur l'utilisation du protocole XCast qui a été spécifié pour IPv4 [66] et IPv6 [69]. XCast (Explicit Multicast) propose une méthode basée sur l'utilisation d'une liste spécifiant explicitement la liste des destinataires d'un paquet. De par sa nature même le protocole XCast est uniquement destiné aux communications de groupe entre quelques participants (entre trois et six selon [66]). La liste spécifiant les adresses des destinataires est en effet incorporée aux paquets envoyés par la (ou les) source(s) d'une communication de groupe : la longueur de la liste est donc limitée par la taille maximale d'un paquet (il faut également laisser une place suffisante aux données). Avec XCast, chaque source maintient la liste de tous les participants d'une communication de groupe. La méthode permettant à chaque récepteur de s'enregistrer auprès de la source n'est cependant pas précisée.

Le routage des données avec XCast fonctionne de la manière suivante. A l'aide de sa table de routage, une source détermine le prochain saut (routeur) pour chacun des récepteurs. La source partitionne alors la liste des destinataires selon le prochain saut utilisé pour atteindre chaque destinataire. Enfin, la source envoie une copie des données à chaque prochain saut identifié dans la procédure précédente : elle inclut alors dans le paquet la sous-liste des destinataires dont la route passe par ce prochain saut. Xcast ne nécessite donc aucun mécanisme de construction et de maintenance des arbres de diffusion. De plus, avec XCast les routeurs intermédiaires ne maintiennent aucun état relatif aux arbres de diffusion.

Le protocole E²MMP se base sur le protocole XCast en ce qui concerne la méthode de diffusion des données. E²MMP est une extension de XCast capable de gérer la présence de participants mobiles dans une communication de groupe. De plus, le protocole E²MMP utilise les mécanismes du protocole Mobile IP pour faire le lien entre l'adresse mère et l'adresse temporaire d'un hôte mobile. Lorsqu'un récepteur mobile se déplace dans un nouveau réseau, il doit se ré-enregistrer auprès de chaque source de données : il utilise pour cela sa nouvelle adresse temporaire et son adresse mère. La source peut alors trouver le meilleur prochain saut relatif à la nouvelle position du récepteur mobile. Réciproquement, lorsqu'une source mobile se déplace, elle envoie un message de mise à jour de l'association à chaque récepteur afin qu'ils puissent faire le lien entre la nouvelle adresse temporaire de la source et son adresse mère.

Le protocole E²MMP est intéressant en raison de sa simplicité. De plus, il est basé sur le protocole XCast qui supprime les difficultés liées à la construction et à la maintenance d'arbres de diffusion. Cependant, ces deux protocoles souffrent d'inconvénients majeurs. Premièrement, XCast ne supprime pas le problème du déploiement incrémental puisqu'il doit être implanté dans tous les routeurs afin de proposer un routage optimal. Deuxièmement, XCast concentre toute la complexité localement : chaque routeur intermédiaire doit en effet analyser la liste des destinataires, trouver le prochain saut pour chaque destinataire, et renvoyer des copies vers chacun des prochains sauts ayant été identifiés. Cette procédure est très pénalisante car chaque saut intermédiaire induit une latence supplémentaire dans la retransmission des données. En outre et comme nous l'avons déjà mentionné, ce protocole est uniquement applicable à des groupes de petite taille. Enfin la gestion de la mobilité introduite par E²MMP est très sommaire car elle se contente d'appliquer les concepts de Mobile IP au protocole XCast.

1.4.7 Conclusion

Les protocoles que nous avons présentés ont chacun leurs avantages et leurs inconvénients. En général, il est difficile de concilier optimisation de routage et diminution de la surcharge liée au déplacement de membres mobiles (ex : reconstruction d'arbres). De plus, il est difficile de minimiser l'interruption de diffusion et de réception des données multicast lors de déplacements. Chaque solution est donc un compromis entre optimisation de routage et réduction de l'impact induit par le déplacement d'un récepteur ou d'une source mobile. Cependant, les deux seules solutions [48, 67] qui permettent la gestion du déplacement de la source d'un arbre par source sont des propositions qui ne suivent pas le modèle classique des communications multicast (le modèle de Deering). De plus, la troisième alternative qui consiste à mettre en place un tunnel bi-directionnel entre la source et son agent mère souffre d'un problème de routage triangulaire. Le routage multicast à arbre par source [30] est pourtant un modèle intéressant qui permet de déployer facilement un routage multicast inter-domaines [32]. De plus, ce modèle est particulièrement adapté aux applications émergentes telles que la télévision et la radio par Internet.

Nous avons donc proposé une solution originale permettant de gérer les déplacements de la source d'un arbre par source dans le cadre du protocole de routage multicast PIM-SSM [30]. Afin d'évaluer les performances de notre proposition, nous avons décidé de réaliser des simulations. On sait cependant qu'il est difficile de simuler l'Internet et ses protocoles de manière réaliste pour diverses raisons [70]. L'une des principales est que l'Internet contient des millions de routeurs et cet immense réseau est donc très difficile à cartographier. Il est de plus quasiment impossible d'utiliser une topologie d'une telle taille dans un simulateur pour des raisons évidentes de performance. Enfin, il n'existe pas de données permettant de simuler de manière réaliste la mobilité des nœuds dans l'Internet. Nous avons donc dans un premier temps validé un environnement de simulation pour nous permettre de réaliser des simulations aussi réalistes que possible. Nous avons en particulier étudié la forme des arbres multicast obtenus avec cet environnement et nous avons comparé ces résultats avec des études similaires. Enfin, nous avons utilisé notre environnement de simulation afin de comparer les deux solutions proposées à l'IETF afin de gérer la mobilité de récepteurs mobiles adhérant à des groupes multicast.

Chapitre 2

Mise en place et validation d'un environnement de simulation

2.1 Introduction

Depuis environ une dizaine d'années, la simulation est devenue la méthode la plus utilisée pour évaluer et mettre au point des protocoles de communication [70]. En effet, l'Internet est devenu un réseau d'une telle ampleur qu'il est devenu quasi impossible de tester des protocoles en situation réelle. De plus, il est parfois indispensable de valider des protocoles avant de les déployer à grande échelle afin de garantir qu'ils n'auront aucun effet néfaste sur le fonctionnement de l'Internet. La simulation est devenue un outil d'une telle popularité qu'on l'utilise dans tous les domaines liés aux communications dans l'Internet. On citera par exemple les simulations des différentes versions du protocole TCP [71], les simulations de trafic de données sur les liens de l'Internet [72], ou encore les simulations du protocole applicatif HTTP utilisé pour les communications liées au web [73].

La simulation a bien entendu des limites puisqu'elle permet uniquement d'étudier un modèle de l'Internet. Dans un premier temps, il est généralement nécessaire de simplifier les modèles afin de les rendre utilisables. On est donc amené à étudier une *version simplifiée* de l'Internet. De même, une simulation se base sur des hypothèses qui restreignent généralement le cadre d'application des résultats. Dans ces conditions, on peut risquer d'obtenir des résultats ne reflétant pas la réalité de l'Internet [74]. En particulier, la superposition de simplifications et d'hypothèses peut amener à ignorer certains facteurs clés ayant une influence majeure dans l'évaluation du protocole étudié. En effet, le modèle d'un protocole de la couche n du modèle TCP/IP dépend très fortement des modèles des couches inférieures à n. On est donc toujours dépendant de l'aptitude des modèles à simuler de manière réaliste les protocoles des couches inférieures à celle à laquelle appartient le protocole étudié.

Dans le cadre de nos travaux sur les communications multicast avec des participants mobiles, il est ainsi indispensable de se baser sur des topologies de réseau réalistes à partir desquelles vont être construits les arbres de diffusion multicast. La topologie physique d'un réseau est en effet le facteur dont l'influence est la plus importante lors de la construction d'arbres multicast. Nous avons donc utilisé un outil permettant de générer des topologies ayant des caractéristiques similaires à la topologie de l'Internet (dans la limite des connaissances actuelles). Afin de valider cet outil, nous avons simulé la construction d'arbres multicast afin d'analyser certaines de leurs caractéristiques topologiques. Nous avons ainsi pu comparer ces données avec les études de référence existantes dans le but de valider notre environnement de simulation.

2.2 Arbres multicast et topologie de l'Internet

Plusieurs études ont montré que la topologie de l'Internet semble suivre un certain nombre de modèles mathématiques pouvant être modélisés par des lois de puissance [75]. Ainsi, le degré des nœuds du graphe de l'Internet ou la distribution des distances dans l'Internet peuvent par exemple être représentés par ces modèles. Il semble que ces lois mathématiques résultent de la mise en œuvre pratique des réseaux, c'est à dire des méthodes et des équipements modernes qui sont utilisés pour créer et interconnecter les infrastructures des réseaux qui forment l'Internet [76]. Or beaucoup de travaux ont eu pour objectif de proposer des méthodes permettant de générer des topologies simulées de l'Internet [77]. En général ces techniques se basent sur des modèles purement mathématiques qui ne parviennent pas à *capturer* certaines des caractéristiques fondamentales de la topologie de l'Internet. De plus, certains générateurs de topologie ont été rendus obsolètes par la découverte des relations de lois de puissance dans la topologie de l'Internet. Afin de baser notre travail sur des topologies réalistes, nous avons utilisé un générateur de topologie récent qui se base sur une méthode originale.

Ce générateur appelé *nem* (pour NEtwork Manipulator) [78], intègre dans un premier temps toutes les découvertes récentes sur les relations mathématiques de la topologie de l'Internet. De plus, *nem* utilise une méthode originale afin de générer des topologies de tailles utilisables dans des simulations. Cet outil utilise en effet une méthode basée sur l'extraction d'un graphe à partir d'une topologie existante de l'Internet. Le graphe extrait de la topologie de l'Internet a en particulier une taille utilisable dans un simulateur. L'algorithme sur lequel est basé *nem* modifie alors le graphe extrait afin que celui-ci vérifie les lois de puissance des topologies de l'Internet et afin qu'il présente des caractéristiques équivalentes à la topologie initiale (comme par exemple la distance moyenne dans le graphe). Cette méthode permet d'obtenir des topologies de taille modeste (quelques milliers de routeurs) ayant des caractéristiques similaires à la topologie de l'Internet [78].

Afin de valider les topologies générées par *nem*, nous avons simulé la construction d'arbres multicast avec des topologies obtenues à partir de cet outil. Plusieurs études ont en effet montré que les arbres multicast présentent des caractéristiques qui peuvent être modélisées par des modèles mathématiques. Ces études se basent notamment sur des arbres résultant de l'utilisation du MBone qui est le réseau expérimental utilisé pour étudier différents aspects des communications multicast. Notre démarche consiste donc à comparer les arbres multicast construits à partir des topologies générées avec *nem* avec ceux obtenus dans d'autres conditions. Notre principal objectif est de valider notre environnement de simulation et en particulier l'utilisation de *nem* comme générateur de topologies. Cet environnement est en effet à la base des tests que nous désirons réaliser avec des participants mobiles.

2.3 Taille des arbres multicast

De nombreux travaux ont été menés sur le routage multicast et en particulier sur la forme des arbres multicast. Suite à une étude visant à estimer le gain obtenu avec le multicast par rapport aux communications unicast, Chuang et Sirbu [79] ont découvert que la taille des arbres peut être représentée par une loi mathématique lorsque le nombre de sites (ou réseaux) avec des membres est connu. Cette étude a notamment considéré différentes topologies dont en particulier un graphe du MBone de 1996 et des topologies générées avec des outils développés avant les découvertes des lois de puissance dans l'Internet. La loi de Chuang-Sirbu est donnée par l'équation

$$L_m = \overline{L_u} N^k \tag{2.1}$$

dans laquelle L_m est le nombre de liens de l'arbre multicast (ou le nombre de nœuds moins 1), $\overline{L_u}$ est la distance moyenne dans la topologie, et N est le nombre de sites avec des membres du groupe multicast. Le facteur k est appelé facteur de mise à l'échelle (scaling factor) et est compris dans l'intervalle [0 ;1]. On note que N n'est pas directement le nombre de membres du groupe, mais le nombre de sites (ou de réseaux) ayant des membres. En terme de routage multicast, un site fait en effet partie de l'arbre de diffusion quel que soit le nombre de membres (du groupe multicast considéré) présents dans le réseau (il faut bien entendu au moins un membre). Dans leur étude, Chuang et Sirbu ont estimé que k était constant et égal à 0,8 dans un grand nombre de situations, c'est à dire indifféremment de la localisation des membres du groupe (ou plutôt des sites avec des membres). Cette découverte a ensuite été confirmée par une analyse mathématique rigoureuse utilisant les mêmes topologies de réseau [80]. Ce travail a également considéré des arbres l-aire (l-ary) dans lesquels un nœud de l'arbre a au plus l fils (on note généralement ces arbres k-aire mais nous voulons évité toute confusion avec le facteur k de la loi de Chuang-Sirbu). Les auteurs de cette étude ont dérivé une forme différente de l'équation 2.1 qui donne cependant des résultats identiques. Enfin, cette étude conclut que la loi de Chuang-Sirbu s'applique à tout réseau dont la fonction "d'atteignabilité" notée $S(r)$ présente une forme exponentielle (i.e. $S(r) \simeq e^{\lambda r}$ pour une certaine valeur de λ). La fonction $S(r)$ indique le nombre de routeurs pouvant être atteints à partir d'un nœud du graphe en exactement r sauts.

Une étude plus récente a cependant remis en question la valeur généralement admise de 0,8 pour le facteur de mise à l'échelle k. Bien que cette étude [81] confirme la forme de la loi de Chuang-Sirbu, ses auteurs ont calculé que k était plus proche de 0,7 que de 0,8. Chalmers et Almeroth ont notamment utilisé des topologies plus récentes du MBone ainsi que des graphes générés avec des outils plus récents vérifiant les lois de puissance de l'Internet. Pour illustrer l'influence de la valeur de k dans la loi de Chuang-Sirbu nous avons calculé le terme N^k pour différentes valeurs de k. Ces résultats sont présentés par la figure 2.1. On remarque qu'une faible différence dans la valeur de k induit une différence notable dans la valeur de N^k. Enfin, une dernière étude [82] à partir d'arbres l-aire conteste le fait que k est stable mais qu'au contraire sa valeur varie de 0,71 à 1 lorsque N augmente. Cette étude précise néanmoins que pour des tailles de topologie de l'ordre de celles utilisées dans les études précédemmment citées ([79, 80, 81]), la valeur de k varie très peu car l'influence de N n'est pas assez forte.

A partir des études existantes il est donc diffile de savoir quelle est la valeur la plus cor-

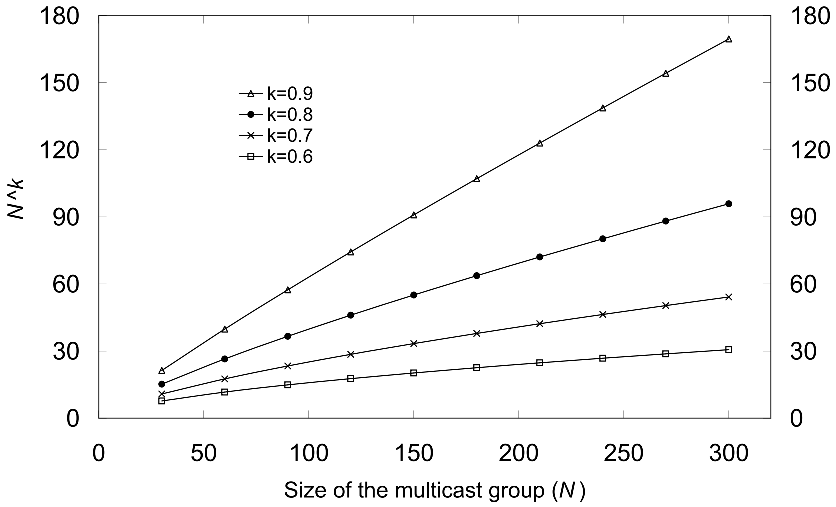

FIG. 2.1 – Influence du facteur k de la loi de Chuang-Sirbu

recte pour k. Toutes ces études semblent en effet également valides. Chalmers et Almeroth [81] indiquent cependant qu'ils utilisent une définition de N différente de celle utilisée par Chuang et Sirbu [79]. Ils considèrent en effet N comme le nombre de membres d'un groupe alors que Chuang et Sirbu définissent N comme le nombre de sites avec des membres. Cette dernière définition semble en effet mieux adaptée au routage multicast puisque l'on considère les routeurs de l'Internet et, comme nous l'avons mentionné plus haut, un routeur fait partie d'un arbre multicast qu'il y ait dans son réseau un seul ou plusieurs membres du groupe multicast correspondant. De plus, la définition de $\overline{L_u}$ est différente selon les études car Chalmers et Almeroth définissent ce paramètre comme la distance moyenne entre les sources et les récepteurs alors que les autres études le définissent comme la distance moyenne dans la topologie étudiée. Il est donc difficile de comparer les résultats obtenus entre d'une part Chalmers et Almeroth et d'autre part Chuang et Sirbu. Néanmoins, la loi de Chuang-Sirbu semble faire l'unanimité et nous avons donc voulu valider notre environnement de simulation en vérifiant si cette loi est présente dans les arbres multicast construits à partir des topologies construites avec *nem*. De plus, un deuxième objectif est de préciser la valeur de k sachant que *nem* génère des topologies respectant les lois de puissance présentes dans la topologie de l'Internet.

2.4 Simulations et mesures

Afin de simuler la construction d'arbres multicast à partir de topologies générées par *nem*, nous avons utilisé le simulateur NS (pour Network Simulator) [83]. Ce simulateur est très largement utilisé par les chercheurs en réseaux et protocoles car il propose une plate-forme ouverte permettant à cette communauté de recherche d'utiliser un outil commun pour évaluer leurs travaux (voir [84] pour plus de détails sur NS). Nous avons utilisé *nem* pour générer cinq topologies différentes de 3000 routeurs chacune. Cette taille est en effet semblable à celles utilisées dans les études similaires et est également un compromis entre les contraintes de fonctionnement du simulateur NS et les performances de *nem*. Ce dernier a en effet des performances optimales lorsque des topologies de 3000 à 4000 routeurs sont générées. Etant donné la complexité de ces topologies (elles contiennent un très grand nombre de liens), nous avons décidé d'utiliser une taille de 3000 routeurs afin que le temps d'exécution d'une simulation avec NS reste inférieur à 1 heure. Enfin nous avons utilisé avec *nem* la carte topologique de l'Internet la plus grande possible. Cette carte a été obtenue d'une part par les laboratoires Lucent Bell et d'autre part dans le cadre d'un projet appelé SCAN. La carte finale contient environ 285000 routeurs et près de 450000 liens. Elle représente la topologie d'une partie de l'Internet de novembre 1999. Il est important de noter que *nem* génère des topologies ayant des caractéristiques proches de la carte tout en rajoutant artificiellement des liens afin de garantir le respect des lois de puissance de la topologie de l'Internet. On obtient donc bien des topologies avec les caractéristiques recherchées. Quelques détails des cinq topologies générées sont présentés par le tableau 2.1.

	Topologie (3000 routeurs)				
Topologie	1	2	3	4	5
Degré moyen	2,6				
Distance moyenne	9,717	8,618	9,29	8,102	8,138
Diamètre (en sauts)	24	19	23	21	19

TAB. 2.1 – Paramètres des topologies générées

A partir de ces topologies, nous avons procédé à la construction d'arbres multicast. Etant donné que nous nous intéressons uniquement à l'arbre formé par les routeurs, nous ne simulons pas l'adhésion de membres. Nous définissons de plus les termes suivants pour un arbre multicast donné. Le *routeur racine* est le routeur vers lequel toutes les branches de l'arbre sont construites. Les *routeurs récepteurs* sont les routeurs à partir desquels les constructions de branches de l'arbre sont initiées. Enfin les *routeurs non membres* sont les routeurs faisant partie de l'arbre mais qui ne sont pas des routeurs récepteurs. Au contraire des routeurs récepteurs, les routeurs non membres n'ont pas de membres (du groupe correspondant à l'arbre observé) dans les réseaux auxquels ils sont connectés. Enfin, les simulations de construction d'arbres ont été effectuées de la façon suivante.

Pour chaque topologie, nous sélectionnons aléatoirement un routeur racine parmi tous les routeurs de la topologie (avec une probabilité de sélection uniforme pour tous les routeurs). De la même manière nous sélectionnons un premier jeu de 30 routeurs récepteurs ce qui correspond à 1% des routeurs de la topologie. Nous construisons alors l'arbre enraciné au routeur racine et analysons ses propriétés topologiques. Nous étendons alors l'ensemble des routeurs récepteurs avec 30 routeurs supplémentaires (pour atteindre une proportion de routeurs récepteurs égale à 2% du nombre total de routeurs), et nous mettons à jour l'arbre et analysons à nouveau ses propriétés. Nous effectuons cette opération jusqu'à atteindre un ensemble de routeurs récepteurs égal à 300 routeurs (soit 10 % du nombre total de routeurs). On note cependant que la procédure de sélection des routeurs récepteurs ne reflète pas forcément la réalité des réseaux. En pratique les récepteurs appartiennent en effet à des réseaux qui sont généralement des feuilles de l'Internet : on devrait donc privilégier ces routeurs lors de la sélection des routeurs récepteurs. Malheureusement, *nem* ne permet pas d'identifier les routeurs feuilles de la topologie et nous avons donc ignoré cette particularité. On note néanmoins qu'avec la procédure de sélection uniforme que nous utilisons, on sélectionnera un nombre majoritaire de routeurs feuilles car ils sont prépondérants dans la topologie. Nous avons enfin effectué autant de simulations qu'il était nécessaire pour obtenir des résultats significatifs, et ce pour chacune des cinq topologies. En général, 95% des résultats que nous présentons sont compris entre la valeur moyenne m présentée plus ou moins 10%, c'est à dire dans l'intervalle $[0,9m; 1,1m]$.

2.4.1 Taille des arbres

Nous avons dans un premier temps vérifié si la loi de Chuang-Sirbu s'applique aux topologies que nous utilisons. Nous avons pour cela mesuré le nombre de routeurs T (avec $T = L_m + 1$) dans chacun des arbres observés pour différentes valeurs du nombre N de routeurs récepteurs. Les résultats de ces observations sont présentés par la figure 2.2. Nous avons pu vérifier que la taille mesurée peut en effet être estimée à partir de la loi de Chuang-Sirbu avec $\overline{L_u} = 8,77$ et $k = 0,71$. Nous utilisons la définition de Chuang et Sirbu pour $\overline{L_u}$ (la distance moyenne dans la topologie). On précise également que le coefficient de corrélation entre les mesures et la loi de puissance est égal à 0,97.

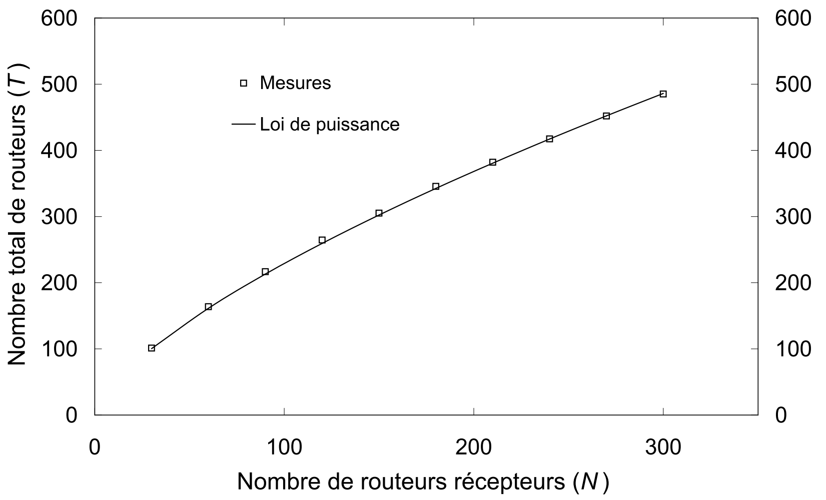

FIG. 2.2 – Taille des arbres multicast

On remarque donc que la valeur de k que nous obtenons (0,71) est plus proche de celle obtenue par Chalmers et Almeroth (0,7) que celle mesurée par Chuang et Sirbu (0,8). Elle est cependant strictement égale à la borne inférieure calculée dans [82] qui correspond à des topologies de quelques milliers de routeurs. Nous pensons que la différence vient du fait que les topologies du MBone utilisées par Chuang et Sirbu sont relativement anciennes. De plus leurs topologies simulées ont été obtenues avec des générateurs développés avant la découverte des lois de puissance dans la topologie de l'Internet. En particulier ces générateurs ont tendance à créer des topologies faiblement maillées ce qui semble diminuer la probabilité de superposition de branches de l'arbre multicast. Ce sont en effet les superpositions de branches qui sont à l'origine de la loi de Chuang-Sirbu et en particulier du facteur k. En effet sans superposition on aurait simplement $L_m = \overline{L_u}N$. Nous pensons donc que la valeur de k que nous obtenons provient du fait que les topologies générées par *nem* sont plus fortement maillées que celles utilisées par Chuang et Sirbu. Enfin, nous avons voulu vérifier l'affirmation de [82] selon laquelle la loi de Chuang-Sirbu est valable lorsque la fonction "d'atteignabilité" a une forme exponentielle. Pour cela nous avons étudié la fonction $T(r)$ telle que

$$T(r) = \sum_{i=1}^{r} S(i) \tag{2.2}$$

Cette fonction représente donc le nombre de routeurs atteignables en r sauts ou moins à partir d'un nœud. La figure 2.3 représente la fonction $T(r)$ obtenue avec les topologies générées avec *nem* et celle obtenue avec une fonction $S(r)$ exponentielle telle que $S(r) = e^{0.3r}$ et de telle façon que les deux fonctions $T(r)$ convergent vers le nombre total de routeurs dans le réseau. On remarque que la croissance de $T(r)$ n'est pas exponentielle ce qui est cohérent avec les connaissances actuelles de la topologie de l'Internet : le voisinage d'un nœud est en effet très dense rapidement ($5 < r < 15$) alors qu'avec une courbe exponentielle la croissance est lente au départ et très rapide par la suite. La principale conclusion de cette comparaison est que la loi de Chuang-Sirbu est non seulement applicable aux topologies ayant une fonction $T(r)$ (strictement $S(r)$) exponentielle, mais qu'elle est également valide avec les topologies récentes de l'Internet qui ne présentent pas cette caractéristique.

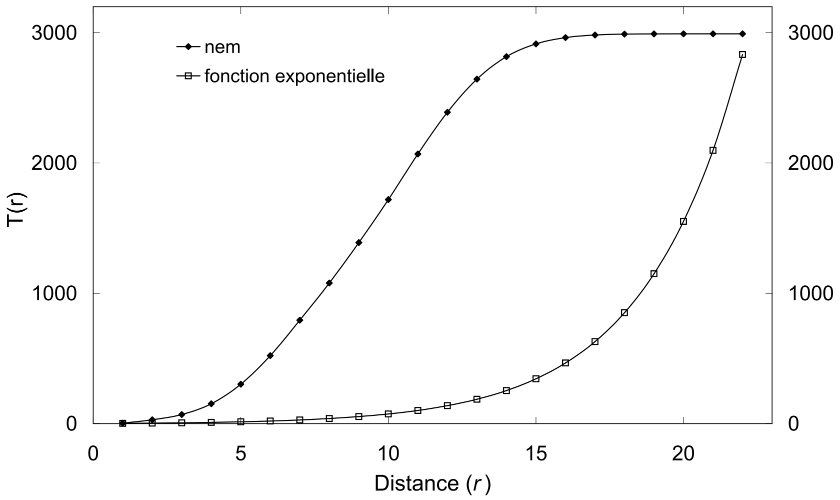

FIG. 2.3 – $T(r)$ avec *nem* et avec une loi exponentielle

Enfin nous terminons cette section en illustrant une conséquence de la loi de Chuang-Sirbu. En effet, lorsque le nombre de routeurs récepteurs N augmente linéairement, le nombre total de routeurs dans l'arbre n'augmente pas aussi rapidement. En particulier, les proportions de routeurs récepteurs et de routeurs non membres varient de manière singulière. On peut en effet calculer la proportion de routeurs récepteurs $p(N)$ dans l'arbre à partir de l'équation 2.1 si on considère que $T = L_m$ (en fait $T = L_m + 1$). On trouve alors que $p(N)$ est égal à

$$p(N) = \frac{N}{\overline{L_u} N^k} * 100 = \frac{100}{\overline{L_u}} N^{1-k}. \tag{2.3}$$

On trouve alors facilement la proportion de routeurs non membres $p(E)$ dans l'arbre puisque $p(E) = 100 - p(N)$. La figure 2.4 illustre ce phénomène. On remarque que les proportions s'inversent lorsque N augmente. Lorsque N est faible, il y a une majorité de routeurs non membres dans l'arbre. Il faut en effet construire une branche entre chaque routeur récepteur et le routeur racine de l'arbre. Si il n'y pas de superposition de branches, on rajoute en moyenne $\overline{L_u}$ routeurs dans l'arbre à chaque construction de branche (si l'on ne compte pas la racine). Au fur et à mesure que le nombre de routeurs récepteurs augmente, le nombre de routeurs non membres rajoutés dans l'arbre à chaque étape décroît. En effet, les superpositions de branches deviennent plus fréquentes et il peut de plus arriver au cours du processus d'agrandissement de l'arbre que l'on sélectionne un routeur non membre comme routeur récepteur ce qui n'a aucune conséquence pour l'arbre. Ce phénomène d'inversion des proportions (qui n'est en fait qu'une conséquence directe de la loi de Chuang-Sirbu) est très positif pour le déploiement futur du routage multicast dans l'Internet. En effet, des groupes très attractifs comme des stations de radio ou de télévision n'induisent pas la construction d'arbres multicast dont la taille croît exponentiellement avec le nombre de récepteurs : la croissance est en effet franche au départ mais perd de son intensité par la suite. On note qu'il serait cependant intéressant de pouvoir réaliser des simulations avec des topologies de plusieurs centaines de milliers de routeurs afin d'étudier la loi de Chuang-Sirbu à très grande échelle. De telles simulations sont cependant difficiles car les logiciels de simulation actuels ne sont pas adaptés à une telle échelle topologique.

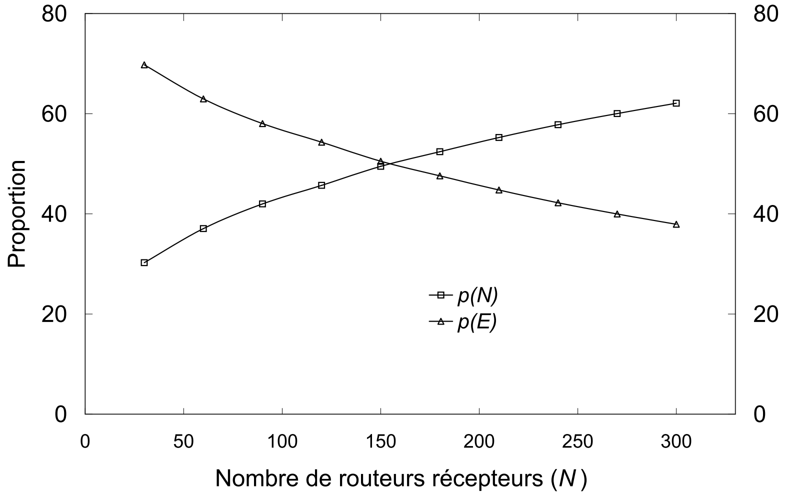

FIG. 2.4 – Proportion $p(E)$ et $p(N)$ par rapport à N

Bien que nous ayons validé notre méthode de simulation en montrant que la loi de Chuang-Sirbu est respectée avec les topologies générées avec *nem*, nous avons décidé d'étendre notre étude en analysant le degré et la profondeur des arbres multicast dont nous avons simulé la construction. Ces indicateurs nous permettent en effet d'étudier ces arbres avec plus de précision.

2.4.2 Degré des arbres

Dans cette section, nous appelons *degré* d'un nœud A d'un arbre multicast le nombre de voisins de A dans l'arbre. Nous utiliserons avec un sens identique les termes *degré dans l'arbre* et *degré*. Lorsque nous voulons mentionner le degré topologique, c'est à dire le nombre de voisins d'un routeur dans la topologie physique, nous utiliserons le terme *degré topologique*. Au cours de cette étude, nous avons constaté qu'il est plus pertinent d'analyser le degré des membres d'un arbre multicast en différenciant les routeurs récepteurs des routeurs non membres. De cette manière il est plus facile de comprendre l'évolution du degré des membres d'un arbre lorsque le nombre de routeurs récepteurs augmente. Nous présentons donc dans un premier temps le degré des routeurs récepteurs puis celui des routeurs non membres. Nous présentons alors l'analyse du degré de tous les routeurs d'un arbre multicast en nous appuyant sur les observations des deux catégories de routeurs considérées.

2.4.2.1 Routeurs récepteurs

Par nature, le degré d'un routeur récepteur est initialement égal à 1. Ces routeurs sont en effet généralement des feuilles de l'arbre multicast. Cependant, quand la taille de l'arbre augmente, le degré de certains routeurs récepteurs est amené à croître. Certains routeurs récepteurs sont en effet appelés à participer à la retransmission des données dans l'arbre à la condition qu'ils n'aient pas un degré topologique égal à 1 (c'est à dire qu'ils ne soient pas des routeurs feuilles de la topologie). Nous avons mesuré la proportion de routeurs récepteurs ayant un degré égal à D lorsque le nombre N de routeurs récepteurs augmente. Ces résultats sont présentés par la figure 2.5. Cette figure indique le pourcentage de routeurs récepteurs ayant un degré dans l'arbre égal à D. On remarque en particulier qu'il y a bien une majorité de routeurs récepteurs avec un degré égal à 1 et que le degré augmente légèrement avec N.

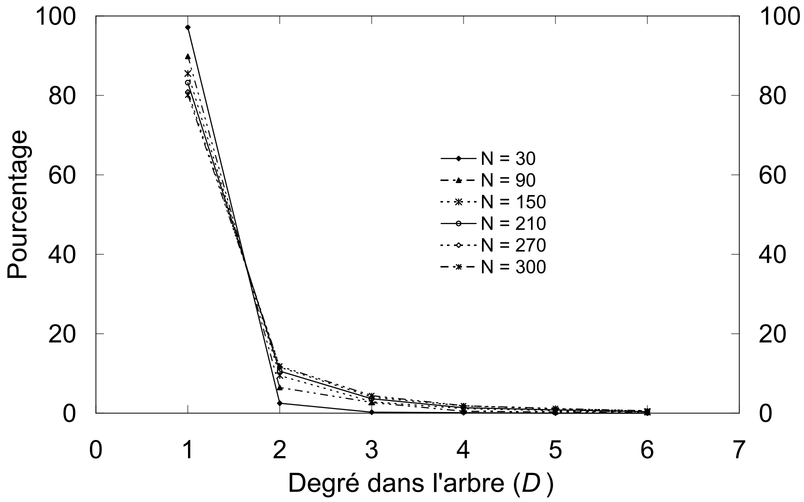

FIG. 2.5 – Pourcentage de routeurs récepteurs avec un degré égal à D

2.4.2.2 Routeurs non membres

Une des premières études sur le degré des arbres multicast a montré que les routeurs de ces arbres ont en grande majorité un degré égal à 2 [85]. Ces *routeurs relais* apparaissent en effet lors de la construction d'une branche entre un routeur récepteur et la racine de l'arbre. Les routeurs intermédiaires sur ce chemin se contentent en effet de faire suivre les données entre la racine et les feuilles de l'arbre. La majorité de ces routeurs relais sont donc des routeurs non membres rajoutés dans l'arbre lors de la construction de celui-ci. Comme précédemment, nous avons mesuré la proportion de routeurs non membres ayant un degré dans l'arbre égal à D lorsque le nombre N de routeurs récepteurs augmente. La figure 2.6 présente ces résultats. On remarque clairement que les routeurs non membres sont en grande majorité des routeurs relais. On note également que lorsque N augmente, le degré des routeurs non membres augmentent. En effet, la construction de nouvelles branches à partir d'un nouveau routeur récepteur peut conduire jusqu'à des routeurs faisant déjà partie de l'arbre ce qui augmente le degré de ces derniers.

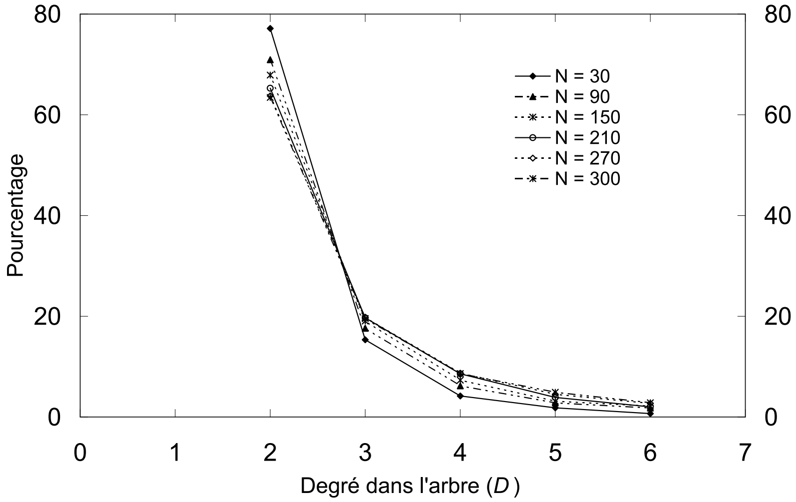

FIG. 2.6 – Pourcentage de routeurs non membres avec un degré égal à D

2.4.2.3 Arbre dans sa globalité

Comme nous l'avons indiqué précédemment, il est pertinent d'analyser le degré des membres d'un arbre multicast en considérant d'un côté les routeurs récepteurs et de l'autre les routeurs non membres. La figure 2.7 présente en effet le degré des membres d'un arbre multicast lorsque le nombre N de routeurs récepteurs augmente. Si nous n'avions pas procédé à l'analyse séparée des deux types de routeurs, il aurait été difficile de comprendre l'évolution des courbes de la figure 2.7. De plus, il est important d'intégrer à cette analyse les conclusions relatives à la proportion des deux types de routeurs dans l'arbre (voir la seconde partie de la section 2.4.1). On remarque en effet que lorsque $N = 30$, 55% des routeurs de l'arbre ont un degré égal à 2 alors que 30% ont un degré égal à 1. Or on sait que dans ce cas les routeurs récepteurs ont majoritairement un degré égal à 1 alors que les routeurs non membres ont majoritairement un degré égal à 2. De plus, on sait que lorsque $N = 30$ les routeurs non membres sont prépondérants dans l'arbre : les résultats de la figure 2.7 sont donc cohérents avec les conclusions précédentes. Au contraire lorsque $N = 300$, les routeurs récepteurs sont prépondérants dans l'arbre et comme leur degré est majoritairement égal à 1 on assiste à une inversion des proportions de routeurs avec des degrés 1 et 2. Il est intéressant de noter que cette analyse a posteriori est directement liée à la compréhension des effets induits par la loi de Chuang-Sirbu : lors de l'analyse des caractéristiques d'un arbre multicast, il est donc pertinent d'étudier séparément les routeurs récepteurs et les routeurs non membres.

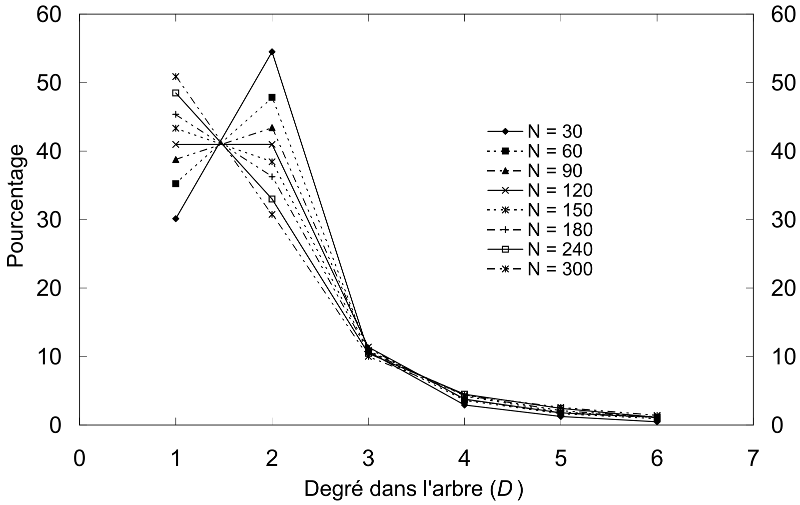

FIG. 2.7 – Pourcentage des routeurs d'un arbre avec un degré égal à D

Pour conclure cette section, nous avons également calculé le degré moyen de tous les routeurs d'un arbre multicast lorsque le nombre N de routeurs récepteurs augmente. Ces résultats sont présentés par la figure 2.8. On remarque que l'effet de la loi de Chuang-Sirbu est toujours aussi important. A priori, il peut en effet paraître surprenant que le degré moyen de tous les routeurs reste stable alors que celui de chacune des deux catégories de routeurs augmente. En fait, lorsque $N = 30$, le degré moyen est principalement influencé par le degré moyen des routeurs non membres qui sont prépondérants dans l'arbre. Au contraire lorsque $N = 300$ l'influence des routeurs récepteurs est la plus importante. L'effet d'inversion des proportions induit par la loi de Chuang-Sirbu modifie donc l'influence globale de chacune des deux catégories de routeurs. Cette observation confirme la nécessité de réaliser des observations séparées de chaque catégorie de routeurs lors de l'étude d'un arbre multicast.

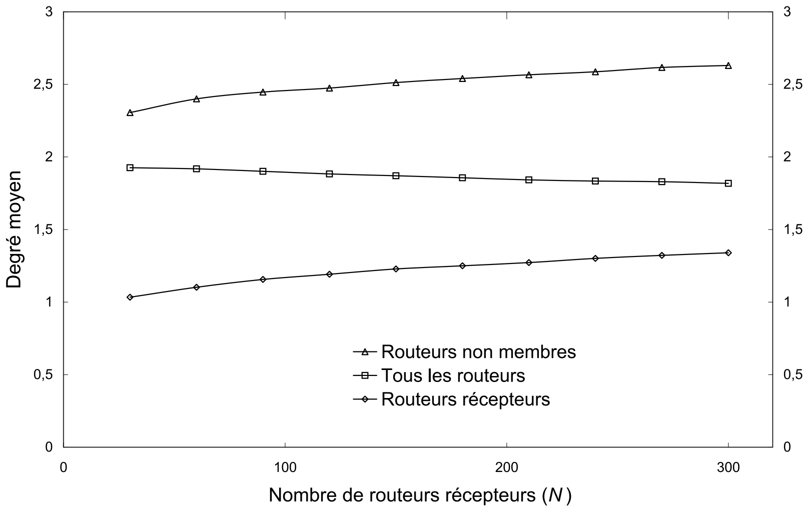

FIG. 2.8 – Degré moyen de tous les routeurs d'un arbre

2.4.3 Profondeur des arbres

Au cours de nos simulations nous avons également analysé la profondeur des arbres multicast. Nous avons de plus analysé le degré des routeurs par rapport à leur profondeur dans l'arbre. Ces observations nous permettent en effet d'étudier avec quelle intensité et à quelle profondeur se produisent les branchements dans l'arbre. Les figures 2.9 et 2.10 présentent ainsi le nombre de routeurs présents à une certaine profondeur dans l'arbre.

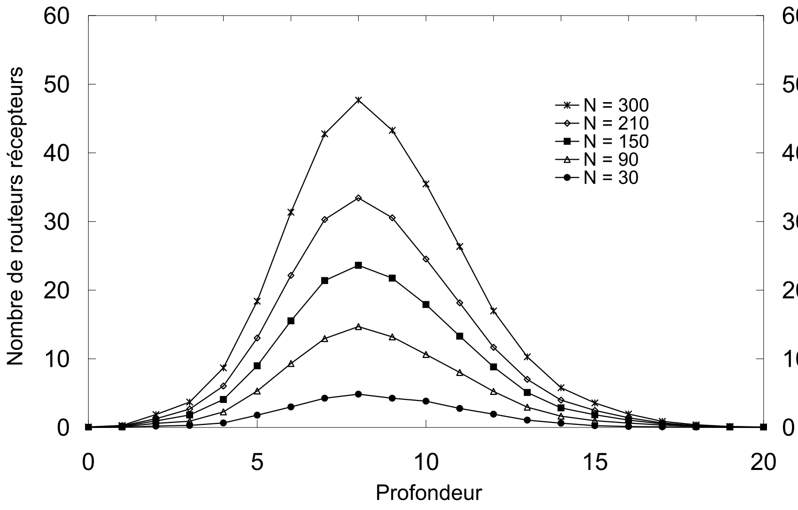

FIG. 2.9 – Profondeur des routeurs récepteurs

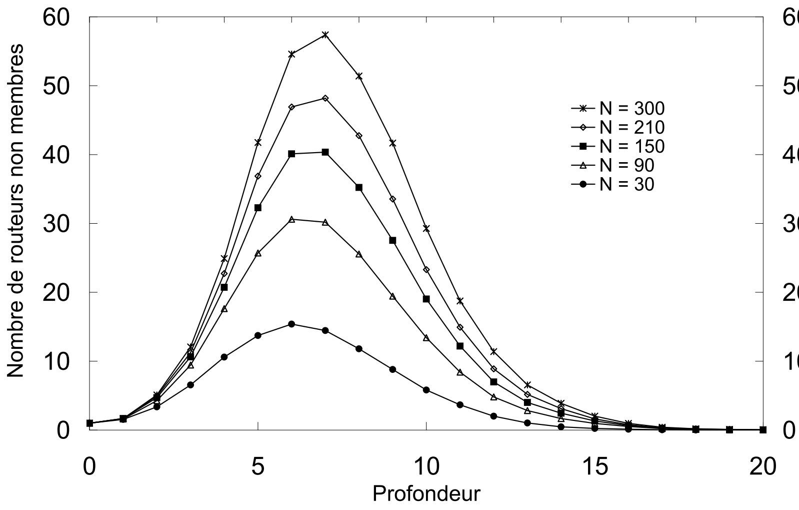

FIG. 2.10 – Profondeur des routeurs non membres

On remarque que les courbes représentant la profondeur des routeurs récepteurs sont des Gaussiennes centrées à une profondeur environ égale à 8,5. Cette observation est tout à fait cohérente avec la méthode utilisée pendant les simulations. Les routeurs récepteurs sont en effet choisis aléatoirement et uniformément parmi tous les routeurs de la topologie. Le routeur racine est sélectionné de la même manière. Statistiquement, la distance moyenne entre le routeur racine et les routeurs récepteurs est donc égale à la distance moyenne des topologies étudiées soit 8,77. La nature uniforme du processus de sélection ainsi que la forme de la fonction "d'atteignabilité" conduisent naturellement à la distribution Gaussienne. On constate également que les courbes représentant la profondeur des routeurs non membres ont également une forme Gaussienne mais celle-ci est légèrement déformée. De plus suivant la valeur de N ces courbes sont centrées à une profondeur comprise entre 6 et 7. Les routeurs non membres sont en effet toujours situés entre un routeur récepteur et le routeur racine et leur profondeur est donc par nature inférieure à celle des routeurs récepteurs. Il est cependant intéressant de noter que la profondeur moyenne des routeurs non membres est comprise entre 6 et 7 sauts. Il y a donc une forte superposition des branches lorsque l'on est proche du routeur racine. En fait cette observation est une conséquence de la forme de la fonction "d'atteignabilité" des topologies que nous utilisons (voir figure 2.3). En effet, un routeur de la topologie a en moyenne peu de voisins à une distance inférieure à 5 sauts : il ne peut donc pas y avoir beaucoup de routeurs non membres autour du routeur racine puisque ceci est topologiquement impossible. Il est alors intéressant d'observer le degré des routeurs par rapport à leur profondeur dans l'arbre multicast. Ces mesures sont présentées par la figure 2.11

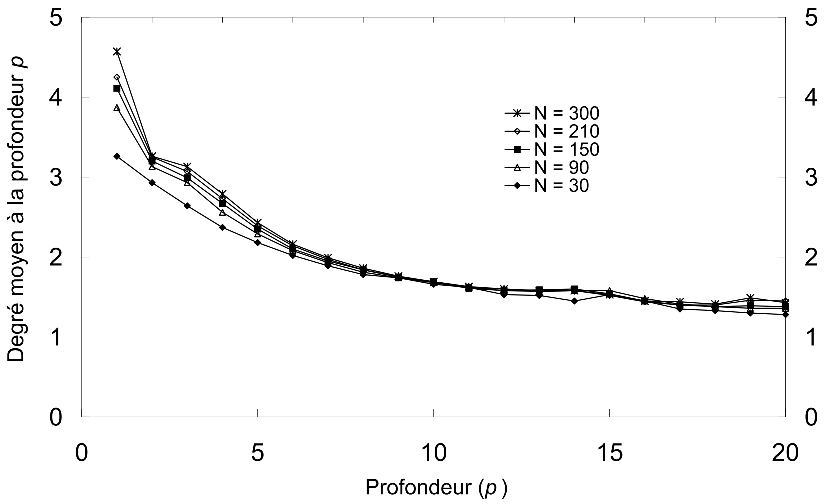

FIG. 2.11 – Degré moyen par rapport à la profondeur

On remarque que la majorité des branchements se font dans le voisinage du routeur racine et que ce nombre diminue lorsque l'on s'éloigne de celui-ci. On constate également que la majorité des routeurs, c'est à dire ceux situés à une profondeur comprise entre 6 et 9, ont un degré égal à 2 : ce sont donc majoritairement des routeurs relais. De manière plus surprenante, on voit que les courbes de la figure 2.11 sont très proches quelle que soit la valeur de N. Enfin, à une profondeur supérieure à 9 le degré moyen est inférieur à 2 car le nombre de routeurs feuilles de degré 1 augmente avec la profondeur. Toutes ces observations nous permettent d'affirmer que les arbres multicast sont construits autour d'un squelette qui est formé très rapidement. Les routeurs proches du routeur racine sont en effet très rapidement présents dans l'arbre même lorsque N est faible. De plus ces routeurs *centraux* supportent la majorité des branchements. La partie supérieur de l'arbre (si la racine est le sommet de l'arbre) que nous appelons squelette de l'arbre, est donc formée très rapidement avec relativement peu de routeurs mais beaucoup de branchements. Lorsque le nombre N de routeurs récepteurs augmente, le degré des routeurs centraux augmente : des branches sont donc greffées au squelette de l'arbre tant que cela est topologiquement possible. Etant donné que la majorité des routeurs situés à une profondeur supérieure à 7 a en moyenne un degré égal ou inférieur à 2, on en déduit que les routeurs récepteurs sont reliés au squelette par de longues branches avec peu de branchements.

2.5 Conclusion

Dans ce chapitre, nous avons dans un premier temps présenté et validé notre environnement de simulation. Nous avons en particulier établi que la loi de Chuang-Sirbu est applicable aux topologies générées par *nem* : les arbres multicast que nous avons simulés respectent donc bien les caractéristiques généralement acceptées dans la littérature. Nous avons de plus étudié le degré et la profondeur des routeurs des arbres multicast en distinguant les routeurs récepteurs des routeurs non membres. Il est en effet pertinent d'étudier ces deux catégories de routeurs de manière séparée car la loi de Chuang-Sirbu modifie la proportion des routeurs de ces deux catégories lorsque le nombre de routeurs récepteurs augmente : leurs influences respectives dépendent donc de la taille des arbres.

Dans le chapitre suivant nous allons dans un premier temps étudier les performances des deux solutions retenues à l'IETF pour gérer la mobilité des récepteurs de groupes multicast dans le cadre du protocole Mobile IPv6. Nous proposons ensuite une solution permettant d'optimiser le routage multicast avec un arbre spécifique à une source lorsque celle-ci est mobile et se déplace. Nous évaluons notre proposition et la comparons avec l'utilisation d'un tunnel bi-directionnel via l'agent mère de la source.

Chapitre 3

Multicast et mobilité

3.1 Introduction

Les spécifications du protocole Mobile IPv6 proposent deux solutions afin de gérer la mobilité des récepteurs d'un groupe multicast. La première alternative propose la mise en place d'un tunnel bi-directionnel entre un récepteur mobile et son agent mère. Avec cette solution, les données multicast sont reçues par le récepteur mobile via son agent mère. L'agent mère adhère donc au groupe multicast du mobile dont il maintient l'association courante. La deuxième alternative est que le récepteur mobile adhère au groupe multicast depuis le réseau visité, c'est à dire comme n'importe quel hôte statique de ce réseau. Les avantages et les inconvénients de ces deux méthodes sont résumés dans la section 1.4.1. Nous illustrons leur fonctionnement par la figure 3.1.

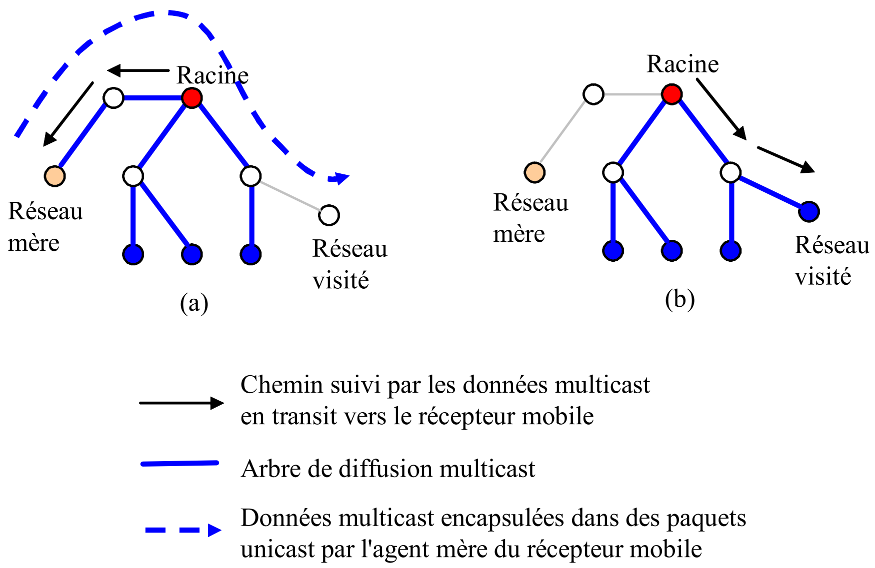

FIG. 3.1 – Gestion des récepteurs mobile avec Mobile IPv6

Avec l'utilisation d'un tunnel bi-directionnel (a), les données multicast transitent nativement vers le réseau mère du récepteur mobile. Ces données sont alors encapsulées dans des paquets unicast vers la position courante du récepteur mobile. Avec l'adhésion depuis le réseau visité (b), le récepteur mobile adhère simplement au groupe multicast depuis sa position courante.

Afin de pallier aux problèmes introduits pas ces deux solutions, nous avons proposé l'utilisation d'un nouveau message MLD avec une méthode d'adhésion utilisant conjointement les deux solutions présentées ci-dessus. Nous décrivons cette proposition dans la prochaine section. Nous évaluons ensuite les deux solutions retenues à l'IETF ainsi que l'alternative que nous proposons. Dans une dernière partie, nous proposons un protocole permettant de gérer la mobilité de la source d'un arbre de type SSM. Un tel arbre est en effet enraciné à la position courante de la source (ou plutôt au réseau hébergeant cette source) : il est donc primordial de pouvoir gérer de manière optimale les déplacements de la source lorsque celle-ci est mobile. Nous évaluons notre proposition en la comparant à la solution du tunnel bi-directionnel mentionnée dans les spécifications du protocole Mobile IPv6.

3.2 Utilisation d'un proxy MLD

En pratique, la mise en œuvre de la solution du tunnel bi-directionnel peut se faire de deux manières. Avec la première méthode, l'agent mère du récepteur mobile est également routeur multicast, et le tunnel d'encapsulation vers le récepteur est vu comme une interface virtuelle. Lorsqu'un nœud mobile envoie un message d'adhésion MLD (MLD report) dans le tunnel vers son agent mère, le protocole de routage multicast de l'agent mère réagit en contruisant la branche appropriée pour le routage des données multicast. Bien que cette méthode soit identique au processus d'adhésion dans un réseau filaire (mise à part l'utilisation du tunnel), elle peut être problématique d'un point de vue fonctionnel. En effet, l'agent mère doit également être routeur multicast. Or un opérateur peut vouloir séparer ces deux fonctionnalités afin d'éviter qu'un agent mère ne participe au routage multicast. On peut par exemple vouloir éviter de surcharger un agent mère avec les fonctions de routeur multicast. Une alternative consiste alors à mettre en œuvre un *proxy MLD* au niveau de l'agent mère [52, 86]. Elle permet en particulier à l'agent mère d'éviter de participer au routage multicast. Les fonctionnalités d'un proxy MLD sont illustrées par la figure 3.2 : l'agent mère effectue les fonctions de routeur MLD pour chaque interface tunnel, et les fonctions de récepteur MLD pour l'interface vers son réseau local courant [19].

Comme avec la méthode précédente, un nœud mobile (NM) envoie son message d'adhésion MLD (MLD report) dans le tunnel vers son agent mère. Le proxy MLD note alors de quelle interface est arrivé ce message afin de maintenir une table de propagation des données multicast. Sur la figure 3.2, le récepteur NM_1 adhère au groupe G_1 via le proxy MLD. Celui-ci note qu'il doit retransmettre les données envoyées à destination de G_1 vers l'interface tunnel i_2 menant au récepteur NM_1. Cette interface est rajoutée à la liste des interfaces de sortie (oif pour output interfaces) pour le groupe G_1 dans la table de propagation. Le proxy MLD adhère alors au groupe G_1 en envoyant un message d'adhésion MLD sur son réseau local. Lorsqu'il y a plusieurs récepteurs mobiles pour un groupe donné, le proxy MLD adhère une seule fois au groupe et retransmet les données vers les interfaces tunnel conduisant aux récepteurs mobiles. Enfin les états de propagation sont périodiquement mis à jour par l'envoi des messages d'adhésion MLD (des récepteurs mobiles vers l'agent mère et du proxy MLD sur son réseau local). On note également qu'en pratique le proxy MLD envoie des messages de requêtes d'appartenance (MLD query) dans les tunnels à destination des nœuds mobiles dont il gère l'association d'adresses. Ces messages sont en effet nécessaires car ils déclenchent l'envoi des messages d'adhésion. Enfin, nous avons validé le fonctionnement d'un proxy MLD en implémentant cette solution sur une plate-forme Linux avec la version du protocole Mobile IPv6 développée dans le cadre du projet MIPL [1].

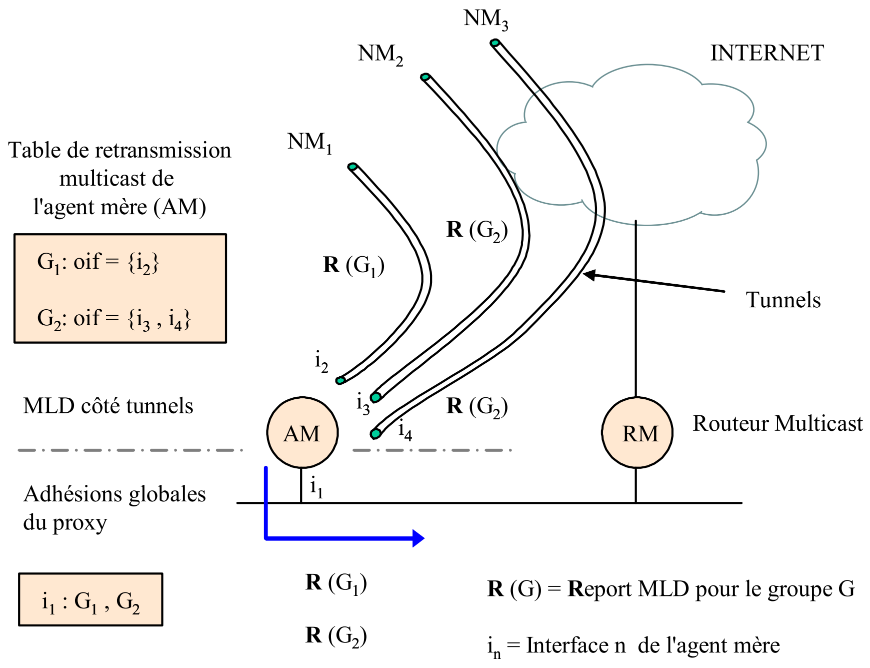

FIG. 3.2 – Fonctionnement d'un proxy MLD

Nous avons de plus proposé l'utilisation d'un nouveau message MLD afin de tirer parti des avantages des deux solutions de l'IETF. La solution du tunnel bi-directionnel permet en effet de minimiser le temps d'interruption de diffusion des données vers un récepteur mobile après que celui-ci se soit déplacé. L'adhésion depuis le réseau visité permet quant à elle d'optimiser le routage des données en évitant le routage triangulaire. Il est donc intéressant pour un récepteur mobile de recevoir les données via son agent mère après un changement de point d'attachement (ou handoff), puis de déclencher l'adhésion depuis le réseau visité. Afin de permettre de mettre en œuvre ce double processus d'adhésion lors de déplacements consécutifs, nous proposons l'utilisation d'un nouveau message MLD : le message MLD d'attente (MLD hold) [87].

Après un déplacement, nous proposons qu'un récepteur mobile adhère toujours au groupe multicast depuis le réseau visité. Lorsque cette adhésion est effective, le routage des données multicast est optimal. Le récepteur mobile envoie alors un message MLD hold à son agent mère afin de lui demander d'arrêter de retransmettre les données multicast sans pour autant quitter le groupe. Si le mobile se déplace à nouveau, il enverra un message Mobile IPv6 de mise à jour de l'association à son agent mère afin de l'informer de sa nouvelle adresse temporaire. Ce message déclenche alors la retransmission des données multicast vers la nouvelle position courante du récepteur. Ce dernier reçoit ainsi les données multicast très rapidement après son déplacement. Le récepteur déclenche de plus l'adhésion depuis le réseau visité afin de profiter d'un meilleur routage des données. Lorsque les données arrivent depuis le réseau visité, le

[1]http ://www.mobile-ipv6.org

récepteur envoie à nouveau un message MLD hold à son agent mère. L'utilisation de ce message permet d'éviter que l'agent mère ait besoin de se réabonner au groupe multicast à chaque fois que le récepteur mobile se déplace après que celui-ci soit resté longtemps dans le réseau visité précédent. L'inconvénient est que l'agent mère reste membre du groupe multicast lorsque le récepteur mobile reçoit les données directement depuis le réseau visité. En pratique, si un agent mère gère la mobilité de plusieurs centaines de mobiles, on peut limiter l'utilisation de ce mécanisme aux groupes multicast les plus populaires. De plus, on limitera également le temps pendant lequel l'agent mère reste membre d'un groupe sans retransmettre les données : cette durée doit être un compromis entre la surcharge induite par l'adhésion de l'agent mère et la latence de retransmission des données après le déplacement d'un récepteur mobile. Si un récepteur mobile a des déplacements fréquents (par exemple si celui-ci se déplace en voiture), un tel mécanisme permettra de minimiser le temps d'interruption de réception des données multicast induit par chaque changement de point d'accès. De plus si un protocole de détection de handoff est utilisé [88], l'agent mère pourra retransmettre les données multicast vers la nouvelle position courante du récepteur dès que celui-ci se sera déplacé.

3.3 Evaluation des solutions pour récepteurs mobiles

Dans cette section nous évaluons les performances des solutions proposées à l'IETF pour la gestion de la mobilité des récepteurs d'un groupe multicast. Nous comparons de plus ces solutions avec l'optimisation apportée par l'utilisation du message MLD hold et analysons également les effets de concentration d'états multicast induits par chaque solution [89]. Nous utilisons en particulier l'environnement de simulation présenté et validé dans le chapitre 2. Nous utilisons ainsi les topologies présentées par la table 2.1.

3.3.1 Méthodologie

Nous avons effectué une première série de simulations afin d'étudier la longueur des routes multicast ainsi que leur latence. Nous avons également illustré l'influence de la taille des arbres multicast sur ces résultats. Dans une deuxième série de simulations, nous avons étudié les effets de concentration des états de propagation dans les routeurs lorsque l'on considère plusieurs arbres multicast. Ces effets sont en effet différents pour chacune des solutions étudiées.

Dans la première expérimentation et ce pour chaque topologie, nous sélectionnons un routeur racine ainsi qu'un routeur qui sera l'agent mère du récepteur mobile. Nous sélectionnons également un routeur pour représenter le réseau visité du récepteur mobile. Le récepteur mobile adhère alors à un groupe multicast de trois manières. Avec la première méthode, le récepteur utilise le tunnel bi-directionnel avec son agent mère pour adhérer au groupe. Ce dernier rejoint le groupe à la réception du message d'adhésion envoyé par le récepteur mobile. Avec la deuxième méthode, l'adhésion se fait depuis le réseau visité. Enfin avec la troisième méthode le tunnel est utilisé mais l'agent mère est déjà membre du groupe multicast : on simule donc l'utilisation du message MLD hold. Pour les trois alternatives nous mesurons la latence avec laquelle les données arrivent vers le récepteur mobile ainsi que la longueur du chemin emprunté par les données. Pour cette dernière mesure, la longueur du tunnel entre l'agent mère et le récepteur mobile est prise en compte. De plus afin d'étudier l'impact de la taille de l'arbre, nous effectuons ces mesures en modifiant le nombre de récepteurs statiques ayant adhéré à l'arbre. Nous utilisons ainsi de 0 à 30 récepteurs (par tranches de 5 récepteurs). Enfin nous avons effectué autant de simulations qu'il était nécessaire afin d'obtenir la convergence des données mesurées en générant notamment différentes combinaisons entre routeur racine, agent mère, et routeur du réseau visité.

Au cours de la deuxième expérimentation nous avons voulu étudier la concentration des états multicast dans les routeurs de la topologie. En effet, si l'on construit plusieurs arbres multicast, certains routeurs vont faire partie de plusieurs arbres et vont donc potentiellement devoir propager un grand volume de données. Afin d'étudier cet effet, nous avons procédé aux simulations suivantes. Nous sélectionnons un routeur de la topologie comme étant l'agent mère de R récepteurs mobiles. Nous sélectionnons alors R routeurs afin de représenter les R réseaux visités hébergeant chacun un récepteur mobile. Enfin nous sélectionnons M routeurs qui seront les racines de M arbres multicast. Chaque récepteur mobile adhère alors aux M groupes multicast : la première fois en utilisant le tunnel bi-directionnel et la seconde fois en adhérant aux groupes à partir du réseau visité. Comme précédemment, nous avons effectué autant de simulations qu'il était nécessaire afin d'obtenir la convergence des données mesurées. Nous avons notamment utilisé différentes combinaisons entre routeurs racines, agent mère, et routeurs des réseaux visités.

3.3.2 Diffusion des données

Dans cette section, nous présentons le gain obtenu en utilisant l'adhésion depuis le réseau visité par rapport à l'utilisation du tunnel bi-directionnel. En effet, l'adhésion depuis le réseau visité offre un routage optimal des données multicast. Au contraire, l'utilisation du tunnel introduit un routage triangulaire des données à partir de la racine de l'arbre via l'agent mère. Nous présentons donc le gain obtenu en utilisant l'adhésion depuis le réseau visité. Nous mesurons le gain en terme de longueur (nombre de sauts) et de latence des routes par lesquelles transitent les données de la racine de l'arbre vers les différents récepteurs. Le gain observé par les récepteurs est présenté par la figure 3.3 sous la forme d'une fonction cumulative inverse.

Ainsi, pour 50% des récepteurs mobiles, la route utilisée par les données multicast est 45% plus courte avec l'adhésion depuis le réseau visité qu'avec le tunnel via l'agent mère. En fait la valeur moyenne des gains est de 47,6% pour la longueur des routes et de 46,5% pour la latence. Avec l'adhésion via l'agent mère les routes multicast sont donc en moyenne deux fois plus longues que celles mesurées avec l'adhésion depuis le réseau visité. De même, la latence moyenne de ces routes est en moyenne le double de celle observée avec l'adhésion depuis le réseau visité. Ces observations sont tout à fait cohérentes avec la nature de l'expérimentation. Avec l'adhésion depuis le réseau visité, la longueur moyenne des routes multicast est égale à la distance moyenne entre chaque récepteur et la racine de l'arbre. Etant donné que nous avons effectué des simulations jusqu'à convergence des données, cette distance est donc égale à la distance moyenne $\overline{L_u}$ entre deux nœuds de la topologie (nous utilisons la même notation que dans la section 2.3). En comparaison, la longueur moyenne des routes multicast avec l'utilisation du tunnel est égale à la distance moyenne entre chaque récepteur et son agent mère plus la distance moyenne entre l'agent mère et la racine de l'arbre. La distance totale est donc en moyenne égale à $2 * \overline{L_u}$. Le gain obtenu par l'utilisation de l'adhésion depuis le réseau visité est donc égal à environ 50%. Le même raisonnement permet d'expliquer le gain d'environ 50% pour la latence. On confirme donc qu'il est bien plus intéressant d'utiliser l'adhésion depuis le réseau visité car les données transitent via un chemin plus court et plus rapide.

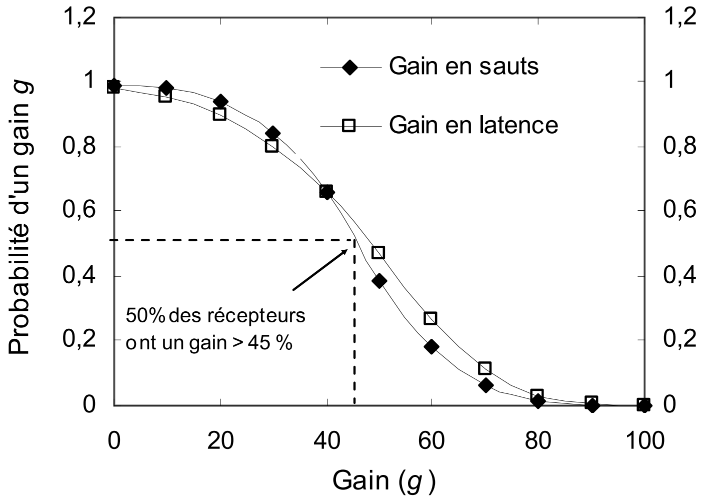

FIG. 3.3 – Gain observé par un récepteur utilisant l'adhésion depuis le réseau visité

3.3.3 Distance à l'arbre multicast

Nous avons également mesuré la distance h (en sauts) entre chaque récepteur et l'arbre multicast de la manière suivante. Avec l'adhésion depuis le réseau visité, h est la distance entre le récepteur et le premier membre de l'arbre multicast sur le chemin du récepteur vers la racine de l'arbre. En pratique, h est donc par exemple la distance que devra parcourir une requête d'adhésion du protocole PIM afin de construire une branche reliant le réseau du récepteur à l'arbre multicast. Avec l'utilisation du tunnel, h représente la distance entre le récepteur et son agent mère plus la distance entre l'agent mère et un membre de l'arbre multicast. Enfin si l'on considère l'adhésion basée sur l'utilisation du message MLD hold, h représente la distance entre le récepteur et son agent mère. En fait la distance h indique avec quelle vitesse l'adhésion au groupe multicast sera effectuée (même si strictement h est une distance). Nous rappelons que nous avons fait varier la taille *statique* de l'arbre en modifiant le nombre de récepteurs non mobiles de l'arbre. On peut ainsi étudier l'influence de la taille de l'arbre sur le processus d'adhésion. La figure 3.4 présente ces résultats.

Sur la figure 3.4, la courbe "Mobile via AM" représente la valeur moyenne de h dans le cas de l'utilisation du tunnel entre le récepteur mobile et son agent mère (AM). De plus la courbe "AM vers arbre" précise la distance moyenne entre l'agent mère et l'arbre avec cette même solution. On voit sur la figure 3.4 qu'avec l'adhésion par le tunnel, la distance h est au maximum égale à environ 16,8 alors qu'elle est égale à environ 8 avec l'adhésion depuis le réseau visité et avec la méthode utilisant le message MLD hold. En effet lorsqu'il n'y a aucun récepteur statique, ces distances correspondent environ à $2 * \overline{L_u}$ et $\overline{L_u}$ (et $\overline{L_u} = 8,77$). Lorsqu'on augmente la taille de l'arbre, toutes les courbes (sauf la courbe "MLD hold") décroissent de la même manière. Etant donné que nous avons effectué des simulations jusqu'à convergence des mesures, la distance moyenne entre le récepteur et l'arbre, et celle entre l'agent mère et l'arbre sont statistiquement égales. Sur la figure 3.4, les deux courbes sont séparées artificiellement pour améliorer la lisibilité de la figure mais elles sont en fait superposées. La seule distance n'évoluant pas est la distance moyenne entre un récepteur et son agent mère : cette distance n'est en effet pas influencée par la taille de l'arbre. Les performances de la technique utilisant les messages MLD hold sont donc directement proportionnelles à la distance séparant un récepteur de son agent mère. Elle est cependant toujours inférieure à celle résultant de l'utilisation sans optimisation du tunnel.

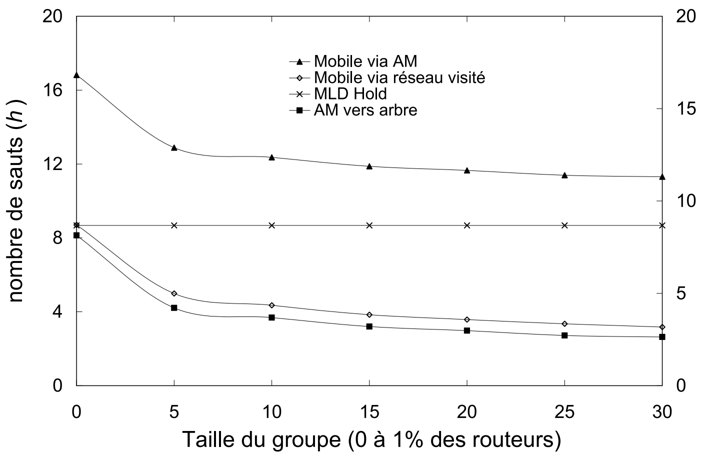

FIG. 3.4 – Distance moyenne (h) entre un récepteur et l'arbre multicast

3.3.4 Concentration des états multicast

Dans cette dernière partie nous avons voulu étudier la concentration des états multicast dans les routeurs lorsque plusieurs arbres multicast sont superposés. En effet, avec l'adhésion via le tunnel, un agent mère peut être amené à adhérer à un grand nombre d'arbres multicast. Ces adhésions peuvent en particulier avoir un impact sur le voisinage de l'agent mère car les routeurs dans la périphérie de l'agent mère seront également amenés à adhérer à plusieurs arbres multicast. Il va donc y avoir une concentration d'états de propagation dans le voisinage de l'agent mère. Alors que l'agent mère est prêt à supporter cette surcharge dans sa gestion des nœuds mobiles, les routeurs de son voisinage ne sont pas forcément disposés à participer à cette diffusion de données multicast. Nous avons donc mesuré le nombre d'états dans ces routeurs lorsque l'adhésion se fait à travers le tunnel (E_T) et lorsqu'elle est faite depuis le réseau visité (E_V). Nous avons ensuité calculé le ratio E_T/E_V afin d'évaluer la surcharge induite par l'utilisation du tunnel via l'agent mère. Nous rappelons que nous avons simulé la construction de plusieurs dizaines de groupes multicast avec un nombre variable de récepteurs mobiles partageant tous le même agent mère. Tous les récepteurs mobiles adhèrent à tous les groupes : la première fois via l'agent mère, puis une deuxième fois à partir du réseau visité de chaque mobile.

La figure 3.5 présente le ratio E_T/E_V calculé au cours des simulations pour les routeurs du voisinage de l'agent mère. Nous avons de plus distingué ces routeurs suivant la distance (en sauts) à laquelle ils se trouvent de l'agent mère. En effet l'influence des adhésions multiples de l'agent mère dépend fortement de la distance par rapport à ce dernier. On remarque dans un premier temps que le ratio E_T/E_V est au maximum égal à 9,5 et qu'il décroît lorsque le nombre de récepteurs augmente. Parallèlement, le rapport est plus faible lorsque le nombre de groupes est plus important. Ce phénomène peut être expliqué de la façon suivante. Lorsqu'on utilise le tunnel, le nombre de récepteurs n'a aucune influence sur le nombre d'états E_T dans les routeurs du voisinage de l'agent mère. En effet, l'agent mère adhère à un groupe multicast quel que soit le nombre de récepteurs mobiles lui ayant envoyé la demande d'adhésion via le tunnel. Seul le nombre de groupes multicast a donc une influence sur E_T. Au contraire avec l'adhésion depuis le réseau visité et pour un groupe donné, la probabilité pour un routeur de la topologie d'avoir un état multicast pour le groupe augmente avec le nombre de récepteurs. Ces derniers sont en effet dispersés dans la topologie et à chaque nouvelle adhésion la taille de l'arbre augmente. De plus la probabilité pour un routeur d'appartenir à plusieurs groupes multicast augmente également avec le nombre de groupes. Nous avons ainsi remarqué que la croissance de E_V est supérieure à celle de E_T, et le ratio diminue donc lorsque le nombre de récepteurs et le nombre de groupes augmentent. On note cependant que ce ratio tend vers 1 ce qui indique, en terme du nombre d'états dans les routeurs proches de l'agent mère, que l'utilisation du tunnel n'est jamais plus intéressante que l'adhésion depuis le réseau visité.

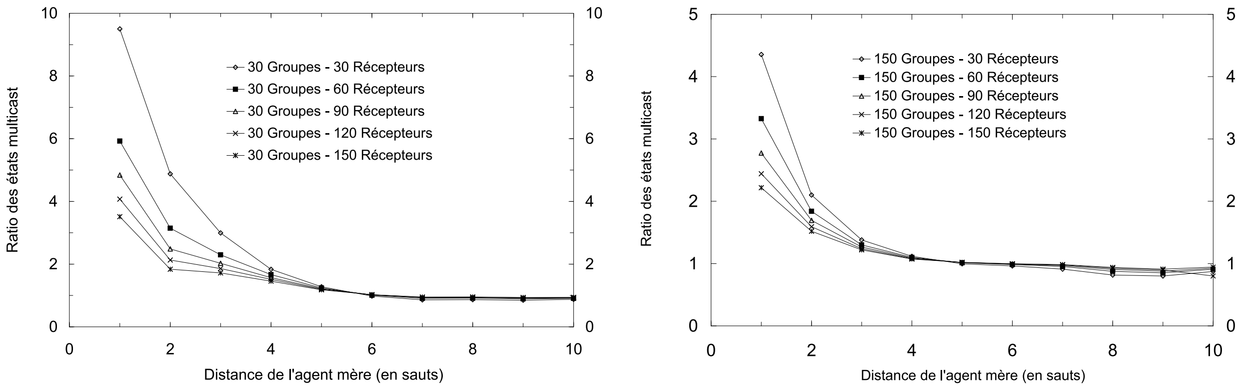

FIG. 3.5 – Ratio E_T/E_V des états multicast

3.3.5 Conclusion

Les résultats que nous venons de présenter nous permettent de tirer un certain nombre de conclusions. Premièrement, la diffusion des données est plus performante lorsqu'un récepteur mobile adhère à un groupe depuis le réseau visité. Le routage des données est en effet optimal alors qu'en moyenne on double la longueur de la route lorsqu'on utilise un tunnel via l'agent mère. De plus, l'utilisation du tunnel via l'agent mère induit une forte concentration d'états multicast dans les routeurs proches de l'agent mère. En pratique, cet impact peut être négatif pour ces routeurs si l'agent mère (ou un groupe d'agents mère) gère un très grand nombre de récepteurs mobiles et de groupes multicast. Ces constatations nous ont amenés à proposer une solution permettant d'optimiser le routage d'un arbre multicast spécifique à une source lorsque celle-ci est mobile. Il est en effet primordial d'éviter le routage triangulaire puisque dans ce cas la mobilité de la source affecte tous les récepteurs alors que précédemment la méthode d'adhésion d'un récepteur n'avait aucune conséquence sur les autres récepteurs. L'utilisation d'un tunnel via l'agent mère présente néanmoins des avantages puisque lors du déplacement d'un récepteur, l'envoi d'un message de mise à jour de l'association permet à l'agent mère de faire suivre les données très rapidement vers la nouvelle position du récepteur mobile. Nous proposons donc une méthode permettant de bénéficier de cet avantage lorsque la source d'un arbre multicast est mobile.

3.4 Support des arbres multicast spécifiques à une source mobile

Dans cette section nous considérons la mobilité de la source d'un arbre multicast spécifique à une source. En particulier nous étudions le cas de la source d'un groupe SSM lorsque celle-ci est mobile. Nous rappelons que le modèle SSM (voir section 1.2.3) introduit la notion de canal (S,G) pour lequel seule une source S émet des données à destination d'une adresse de groupe G. L'arbre multicast associé au canal (S,G) est enraciné au réseau dont fait partie la source. Dans la suite de cette section, cet arbre sera appelé arbre (S,G). La mobilité de la source S d'un canal SSM (S,G) introduit deux problèmes majeurs. Premièrement, si la source se déplace et acquiert une nouvelle adresse temporaire S' à l'aide des mécanismes de Mobile IPv6, l'arbre (S,G) existant devient obsolète. En effet les routeurs de l'arbre (S,G) retransmettent uniquement les paquets de données dont l'adresse source est S. De plus, cet arbre est enraciné au réseau dont fait partie l'adresse S alors que la source s'est déplacée dans un nouveau réseau. Une solution permettant de gérer ces deux problèmes est d'utiliser un tunnel avec l'agent mère de la source. Lorsque la source se déplace, elle encapsule les données multicast dans un tunnel vers son agent mère. Afin de permettre la diffusion des données, l'adresse source des datagrammes multicast doit être S. Nous illustrons ce procédé par la figure 3.6.

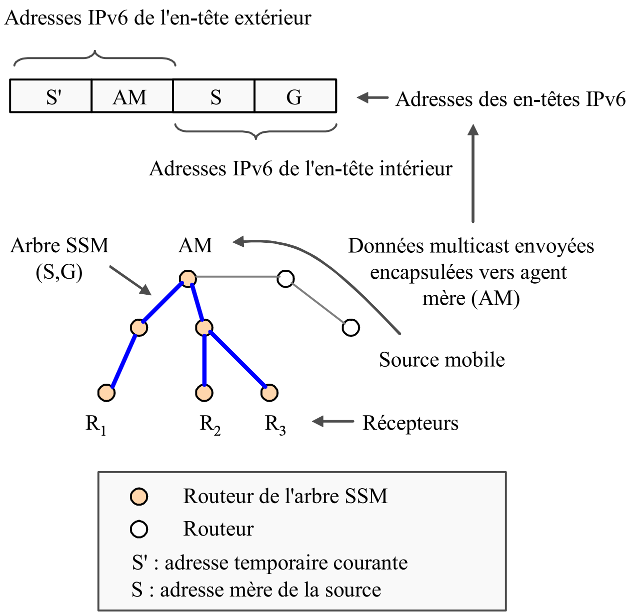

FIG. 3.6 – Utilisation d'un tunnel avec une source SSM mobile

Cette solution a l'avantage d'être très simple. Elle permet de plus de *masquer* la mobilité de la source, c'est à dire que les récepteurs ne savent pas quelle est la position courante de la source, ni même si celle-ci est mobile. Cependant, l'utilisation d'un tunnel introduit un routage triangulaire et nous avons montré dans la section précédente qu'un tel routage est très pénalisant. L'effet du routage triangulaire est ici d'autant plus négatif que tous les récepteurs sont pénalisés. Enfin, si un agent mère gère la mobilité de plusieurs sources mobiles, l'utilisation de tunnels aura pour conséquence une concentration des états multicast dans le voisinage de l'agent mère (voir résultats précédents).

On note également que la solution de l'adhésion depuis le réseau visité n'est pas utilisable avec PIM-SSM. En effet dans ce cas particulier, l'adhésion depuis le réseau visité n'a pas de sens puisque la source est la racine de l'arbre de diffusion. Lorsque que la source se déplace, l'adhésion depuis le réseau visité ne permet en effet pas de reprendre la diffusion des données vers les récepteurs car ils maintiennent leur adhésion à l'ancien canal. Cette solution n'est donc pas applicable à la mobilité de la source d'un canal SSM.

Dans la suite de ce chapitre, nous proposons une solution permettant de reconstruire un arbre des plus courts chemins lors du déplacement de la source d'un canal SSM tout en minimisant le temps d'interruption de la diffusion des données. Nous évaluons ensuite notre solution en la comparant avec l'utilisation d'un tunnel qui la seule alternative possible.

3.4.1 Le protocole MSSMSv6

Le protocole MSSMSv6 (Mobile SSM Sources for IPv6) [90] que nous proposons est basé sur l'introduction de la notion de session SSM et sur l'utilisation d'un mécanisme inspiré par le protocole Mobile IPv6. Nous définissons dans un premier temps la notion de session SSM comme suit : une session SSM (I,G) identifie de manière unique une communication de groupe de type SSM par un identifiant I et une adresse de groupe G. L'identifiant I est en particulier une adresse IPv6 invariante. Dans le cas particulier d'une communication SSM avec une source mobile, la session (I,G) est donc invariante alors que le canal (S,G) varie si S est l'adresse courante (topologique) de la source. En pratique, l'invariant I est l'adresse mère de la source mobile. Lorsque la source envoie des données au canal (S,G), elle rajoute une option destination "adresse mère" à chaque datagramme multicast. Cette option, positionnée à I, permet donc aux récepteurs de reconstruire la session (I,G) correspondant au canal (S,G). En particulier, la couche transport des récepteurs utilise la session (I,G) afin d'identifier la communication. Enfin avant de décrire en détails le fonctionnement de notre proposition, nous rappelons le fonctionnement du mécanisme d'annonces publiques des sessions utilisé actuellement dans l'Internet.

Il existe en effet dans l'Internet un mécanisme permettant d'annoncer publiquement l'existence d'une communication de groupe. Ce mécanisme se base sur l'utilisation d'une adresse de groupe particulière afin de propager des *annonces de session*. L'adresse de groupe utilisée pour propager est en particulier définie par le protocole SAP (Session Announcement Protocol) [91]. Les messages d'annonces de session utilisent un format particulier spécifié par le protocole SDP (Session Description Protocol) [92]. En pratique, une application spécifique appelée SDR (Session Directory) est utilisée : SDR s'abonne simplement au groupe public servant aux annonces et affiche à l'utilisateur la liste des sessions actives. Si un utilisateur décide de créer un groupe, il utilise de même SDR afin d'émettre son annonce de session. Dans la suite de ce chapitre nous considérons qu'un outil de type SDR est utilisé pour l'annonce des sessions multicast. Enfin nous utilisons la terminologie introduite par le protocole Mobile IPv6, en particulier au niveau du nommage des adresses.

3.4.1.1 Fonctionnement du protocole

Lors du déplacement de la source d'un canal SSM, la seule manière d'éviter le routage triangulaire est de reconstruire un arbre des plus courts chemins enraciné à la nouvelle position de la source. Or dans les communications multicast l'adhésion est initiée par les récepteurs : il est donc nécessaire de trouver une méthode permettant d'informer les récepteurs de la nouvelle adresse de la source. Nous utilisons pour cela un nouveau message inspiré du message de mise à jour de l'association utilisé avec Mobile IPv6. Ce message que nous appelons "notification de déplacement d'une source SSM" permet à une source d'informer les récepteurs de sa nouvelle adresse. Le format de ce message utilise celui d'une sous-option Mobile IPv6 comme illustré par la figure 3.7. Ce message contient uniquement la nouvelle adresse de la source. Le champ TBD (To Be Defined) est le numéro attribué à la sous-option qui est à définir. Le champ suivant indique la longueur des données soit 16 octets.

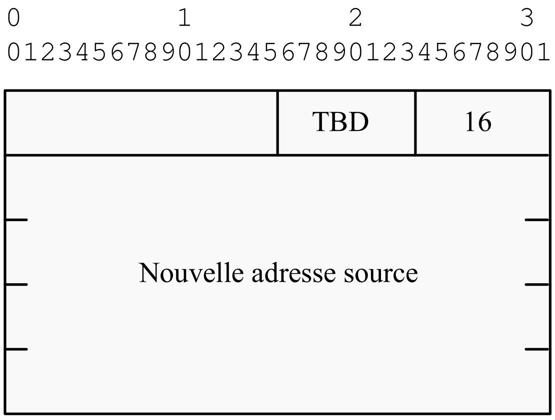

Fig. 3.7 – Sous-Option utilisée par MSSMSv6

Le fonctionnement du protocole MSSMSv6 est illustré par la figure 3.8. Lorsque la source se déplace, elle acquiert une nouvelle adresse temporaire (nCoA pour new Care-of-Address) dans le réseau visité. On note que l'ancien canal SSM est noté (oCoA,G) : il utilise l'ancienne adresse temporaire de la source (oCoA pour old Care-of-Address). On rappelle également que la session SSM est définie par la paire (I,G) c'est à dire avec l'adresse mère I de la source. Après le déplacement, la source met en place un tunnel avec un routeur d'accès de son ancien réseau : ce routeur est noté oAR pour old Access Router. Comme dans des travaux utilisant l'ancien routeur d'accès [88], nous considérons qu'il existe une relation de confiance et de sécurité entre la source et le routeur oAR afin de permettre la mise en place de ce tunnel. Une fois que le tunnel est établi, la source envoie un message contenant l'option permettant d'annoncer sa nouvelle adresse. Le message encapsulé est envoyé à destination du groupe G et avec comme adresse source oCoA. Lorsque le routeur oAR décapsule ce message, il envoie le message sur son réseau local. Ce message est alors propagé sur l'arbre (oCoA,G) et est donc reçu par tous les récepteurs ayant adhéré au canal (oCoA,G). De plus le paquet contient une option destination "adresse mère" qui contient l'adresse mère I de la source. Chaque récepteur peut donc associer le message à la session SSM (I,G) à laquelle il a initialement adhéré. Chaque récepteur peut alors initier une nouvelle adhésion au canal (nCoA,G) : il reste cependant membre du canal (oCoA,G) jusqu'à ce qu'il reçoive les données à partir de (nCoA,G).

Après son déplacement, la source continue en effet à envoyer les données à travers le tunnel vers le canal (oCoA,G). Ces datagrammes multicast ont comme adresse source oCoA et comme adresse destination G : ils sont donc retransmis sur le canal (oCoA,G). De plus la source émet les mêmes données à destination du canal (nCoA,G) dans des datagrammes ayant comme adresse source nCoA. Lorsqu'un récepteur commence à recevoir les données sur le nouveau canal (nCoA,G), il peut informer le routeur multicast de son réseau qu'il souhaite se désabonner de l'ancien canal (oCoA,G). L'arbre multicast (oCoA,G) est donc graduellement élagué et dans le même temps l'arbre (nCoA,G) est progressivement agrandi par l'adjonction de nouvelles branches. La migration des récepteurs vers le nouveau canal est donc progressive et induit une perte minimale de données qui est uniquement liée au temps nécessaire au handoff de la source. De plus, la construction des branches du nouvel arbre est initiée par chaque récepteur et ne nécessite donc aucune synchronisation. Enfin lorsque tous les récepteurs ont rejoint le nouvel arbre, la source peut arrêter d'envoyer une copie des données vers l'ancien canal. Nous suggérons l'utilisation d'un protocole tel que MSNIP (Multicast Source Notification of Interest Protocol) [93] afin de permette à la source de savoir si il existe toujours des récepteurs ayant adhéré au canal (oCoA,G). On précise également que tant que des récepteurs adhèrent au canal (oCoA,G), la source envoie régulièrement le message de "notification de déplacement d'une source SSM" afin de s'assurer que les récepteurs adhèrent bien au nouvel arbre.

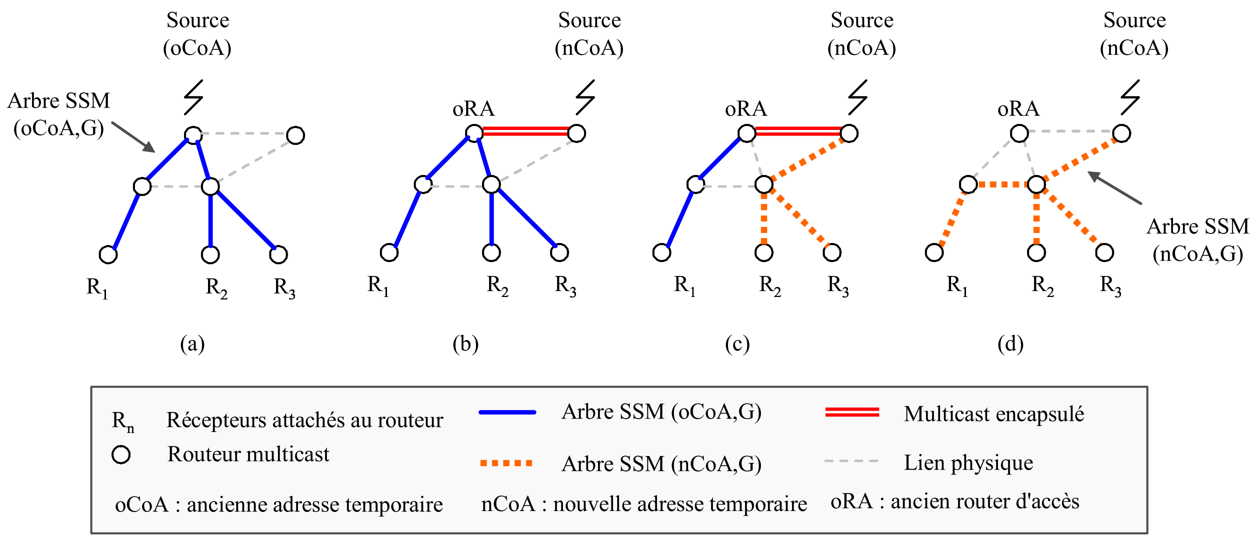

FIG. 3.8 – Fonctionnement de MSSMSv6

Enfin, la source est responsable de l'annonce de la communication. Nous proposons d'utiliser une annonce particulière contenant à la fois la session et le canal SSM. La session (I,G) annoncée est invariante alors que le canal (S,G) est mis à jour à chaque handoff de la source avec la nouvelle adresse temporaire de la source. Une telle annonce permet donc à de nouveaux récepteurs d'adhérer au canal courant. Cependant, le protocole SDP ne permet actuellement pas de réaliser une telle annonce. Il est néanmoins possible d'utiliser un champ optionnel de description (noté 'c=' dans SDP) afin d'inclure l'adresse mère I de la source lors de l'annonce.

3.4.1.2 Déplacements multiples de la source

Dans le cas particulier où une source se déplace *rapidement* et change plusieurs fois de point d'attachement, le fonctionnement de notre proposition reste inchangé. La source doit mettre en place un tunnel avec chacun des anciens routeurs d'accès des réseaux qu'elle a visités. Elle envoie des messages de "notification de déplacement d'une source SSM" vers chacun des anciens arbres multicast afin d'annoncer sa nouvelle position courante. Dans un fonctionnement sans perte de paquets, chaque récepteur va donc *suivre* les déplacements de la source avec un certain retard. Si ce retard est supérieur à la vitesse de déplacement de la source, il est possible qu'un récepteur soit en train d'adhérer à un canal SSM alors que la source s'est déjà à nouveau déplacée. Dans ce cas, le récepteur recevra un nouveau message de "notification de déplacement d'une source SSM" à travers l'arbre auquel il est effectivement connecté. Le récepteur pourra donc déclencher l'adhésion vers le nouveau canal SSM de la source et annuler son adhésion précédente (par l'envoi d'un message MLD done). Il est important de noter que dans le cas d'un déplacement rapide de la source les récepteurs pourront toujours recevoir les données via l'un des anciens canaux de la source. On assure ainsi une migration des récepteurs vers le canal courant tout en garantissant une perte minimale des données.

3.4.1.3 Surcharge, complexité, et déploiement

Le protocole MSSMSv6 induit une très faible surcharge au niveau des routeurs de la topologie sur laquelle il est déployé. En effet, la construction d'un arbre multicast avec un protocole de type PIM-SSM génère l'envoi périodique d'un message par lien de l'arbre. Selon la loi de Chuang-Sirbu, on a donc environ $\overline{L_u} * N^k$ messages envoyés lorsque l'on considère N sites avec des récepteurs. Avec MSSMSv6, le déplacement d'une source induit donc l'envoi de $\overline{L_u} * N^k$ messages afin de construire l'arbre enraciné à la nouvelle position de la source. De plus, chaque message est uniquement envoyé sur le lien séparant un routeur de son routeur père dans l'arbre. La surcharge est donc très faible. Enfin, notre proposition ne nécessite aucune modification des protocoles de routage multicast de l'Internet contrairement à d'autres solutions telles que [53]. Notre solution nécessite en effet uniquement des modifications de la source et des récepteurs et plus précisément des piles protocolaires Mobile IPv6 et MLD.

3.4.2 Evaluation du protocole MSSMSv6

Dans cette section nous évaluons notre proposition en la comparant avec la solution utilisant un tunnel via l'agent mère de la source mobile. Comme précédemment, nous étudions la longueur des routes entre la source et les récepteurs. Nous précisons cette mesure en comparant la longueur des routes induites par les deux solutions en prenant en compte la distance à laquelle se trouve la source de son agent mère. Enfin comme dans l'étude sur les récepteurs mobiles, nous étudions la concentration des états multicast dans les routeurs du voisinage de l'agent mère.

3.4.2.1 Méthodologie

Nous considérons les mêmes topologies que celles utilisées dans les simulations précédentes. Pour chaque topologie, nous sélectionnons 100 routeurs récepteurs de manière aléatoire ainsi qu'un *routeur mère* qui symbolise le réseau mère de la source mobile. Nous sélectionnons enfin un *routeur visité* qui symbolise le réseau visité vers lequel va se déplacer la source. La source étant initialement dans son réseau mère, nous construisons dans un premier temps le canal

(I,G), c'est à dire le canal enraciné au réseau mère de la source. La source se déplace alors pour arriver dans le réseau visité. Nous simulons tout d'abord l'utilisation du tunnel via l'agent mère puis répétons cette expérience en utilisant notre proposition MSSMSv6. Dans chaque cas nous effectuons plusieurs mesures que nous présentons dans les prochaines sections. De plus, nous avons effectué autant de simulations qu'il était nécessaire afin d'obtenir la convergence des données mesurées. Nous avons notamment utilisé différentes combinaisons entre routeur mère, routeur visité, et routeurs récepteurs. Enfin nous avons également choisi les routeurs mère et visité en mesurant la distance d séparant ces deux routeurs afin de présenter des résultats pour une distance d donnée.

Dans une deuxième série de simulations, nous avons voulu étudier la concentration des états multicast dans les routeurs du voisinage du routeur mère lorsque plusieurs canaux SSM sont superposés. Les sources de chaque canal partagent le même réseau mère mais il n'y a qu'une source dans un réseau visité donné. Nous sélectionnons alors un nombre variable de récepteurs pour chaque canal SSM et construisons dans un premier temps les arbres enracinés au réseau mère des sources, puis les arbres enracinés aux positions courantes des sources dans les réseaux visités. Dans chaque cas nous mesurons le nombre d'états multicast dans les routeurs du voisinage du routeur mère. Nous pouvons ainsi comparer les conséquences de l'utilisation du tunnel par rapport à notre proposition MSSMSv6. Comme précédemment, nous avons effectué autant de simulations qu'il était nécessaire afin d'obtenir la convergence des données mesurées. Nous avons notamment fait varier le nombre de récepteurs par canal ainsi que le nombre total de canaux.

3.4.2.2 Diffusion des données

Dans cette section nous présentons le gain obtenu en utilisant le protocole MSSMSv6 par rapport à l'utilisation du tunnel via l'agent mère. Après le déplacement d'une source et une fois que le nouvel arbre enraciné à la nouvelle position de la source est construit, le protocole MSSMSv6 permet en effet d'avoir un routage optimal entre les récepteurs et la source du canal SSM. Afin d'évaluer le gain apporté par MSSMSv6, nous avons mesuré la longueur ainsi que la latence des routes entre la source et les récepteurs. On précise que les temps de handoff de niveaux 2 et 3 (couches réseau et liaison de données) de la source sont ignorés car ils sont identiques quelle que soit la solution considérée. Ils dépendent en effet uniquement des conditions d'accès au réseau (association avec le nouveau point d'accès) et de la latence nécessaire à la création de l'adresse temporaire dans le nouveau réseau (délai avant réception d'un message d'annonce de routeur dans le nouveau réseau).

Dans le cas où le tunnel via l'agent mère est utilisé, nous avons pris en compte la longueur du tunnel mis en place entre la source et son agent mère. Après le déplacement de la source, les récepteurs réceptionnent à nouveau les données multicast en moyenne en 730 ms de temps simulé lorsque le tunnel via l'agent mère est utilisé. On note qu'avec la solution MSSMSv6, le temps d'interruption de réception des données est identique puisque les récepteurs reçoivent dans un premier temps les données via l'ancien canal SSM (oCoA,G). De plus avec MSSMSv6, le nouvel arbre (nCoA,G) est entièrement construit en moyenne en 2 secondes de temps simulé. Pour que le protocole MSSMSv6 permette aux récepteurs de recevoir les données de manière optimale, il faut donc que la source reste au minimum 2 secondes (de temps simulé) dans le réseau visité. En pratique, il faut qu'une source reste dans le réseau visité pendant un temps au minimum deux fois supérieur au temps nécessaire à la construction d'un arbre multicast. Dans le cas contraire, l'utilisation du protocole MSSMSv6 n'est pas optimale car les récepteurs ne profitent pas forcément de l'adhésion au nouveau canal de diffusion.

Lorsqu'avec le protocole MSSMSv6 les récepteurs adhèrent au nouveau canal SSM, le gain observé par chaque récepteur est très important. La figure 3.9 présente ainsi le gain moyen observé par chaque récepteur suivant la distance d entre le routeur mère et le routeur visité de la source : cette distance représente en effet l'influence du routage triangulaire via l'agent mère. Nous avons mesuré le gain en terme de longueur des routes (gain en sauts) et de latence de réception des données (gain en latence). On voit par exemple sur la figure 3.9 que pour $d = 6$, les routes entre la source et les récepteurs sont en moyenne 45 % plus courtes avec MSSMSv6 qu'avec l'utilisation du tunnel via l'agent mère. Le protocole MSSMSv6 conduit en effet à la construction d'arbres de plus courts chemins. On observe de plus sur la figure 3.9 que le gain augmente avec d et tend vers une valeur limite d'environ 45% : lorsque d augmente, l'influence néfaste du tunnel est accrue. Nous ignorons en effet les résultats obtenus pour $d > 8$ car ils sont faussés par des effets de bord car dans ces conditions on approche des bordures physiques de la topologie. De plus, ces données ne sont pas significatives car les handoffs avec $d > 8$ sont très rares.

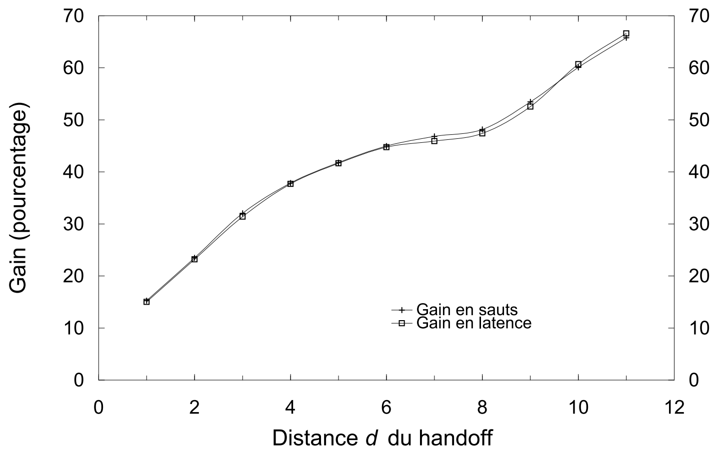

FIG. 3.9 – Gain moyen observé par chaque récepteur avec MSSMSv6

La figure 3.10 présente sous la forme d'une fonction cumulative inverse le gain observé par les récepteurs (en terme de longueur de la route entre chaque récepteur et la source). On voit par exemple sur la figure de gauche que quelle que soit la valeur de d, 50% des récepteurs ont un gain supérieur à 40%. Ces résultats sont relativement identiques à ceux obtenus lors des simulations avec des récepteurs mobiles (voir section 3.3.2) : le routage triangulaire double en moyenne la longueur des routes pour la diffusion multicast. On observe également que même lorsque la distance entre le routeur mère et le routeur visité est faible ($d = 1$), seuls 25% des récepteurs n'observent aucun gain (sur la figure de droite, 75% des récepteurs ont un gain non nul). La construction des arbres de plus courts chemins par le protocole MSSMSv6 génère donc toujours un gain non négligeable quelle que soit la valeur de la distance d.

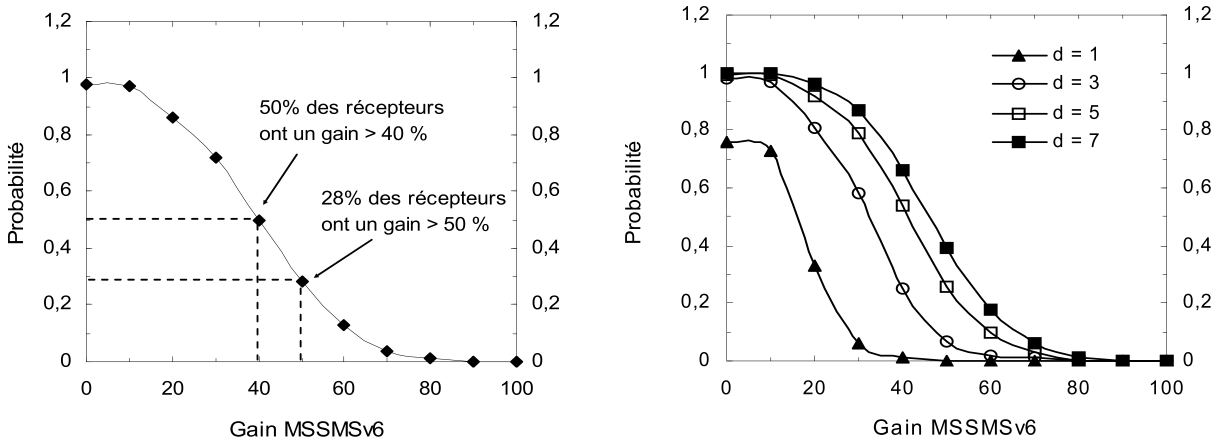

FIG. 3.10 – Gain observé par les récepteurs avec MSSMSv6

3.4.2.3 Concentration des états multicast

Dans cette section nous avons évalué l'effet induit par l'utilisation de tunnels sur la concentration des états multicast dans les routeurs du voisinage du routeur mère. Avec les tunnels, si un agent mère a la charge de retransmettre les données de plusieurs sources mobiles, tous les arbres SSM seront enracinés au niveau du routeur mère. Il va donc y avoir une concentration des états de propagation autour du routeur mère. Au contraire avec MSSMSv6, les arbres sont enracinés aux positions courantes des sources et on évite donc l'effet de concentration dans les routeurs du voisinage du routeur mère. Nous avons donc mesuré le nombre d'états dans ces routeurs lorsque l'envoi des données se fait via l'agent mère (E_T) et lorsqu'il est fait depuis la position courante de chaque source (E_V). Nous avons ensuite calculé le ratio E_T/E_V afin d'évaluer la surcharge induite par l'utilisation du tunnel via l'agent mère. Nous rappelons que nous avons simulé la construction de plusieurs dizaines de groupes multicast avec un nombre variable de récepteurs et de sources mobiles dispersés dans la topologie. Tous les récepteurs adhèrent à tous les groupes : la première fois via le canal (I,G), puis une deuxième fois via le canal (nCoA,G). Les résultats de ces mesures sont présentés par la figure 3.11.

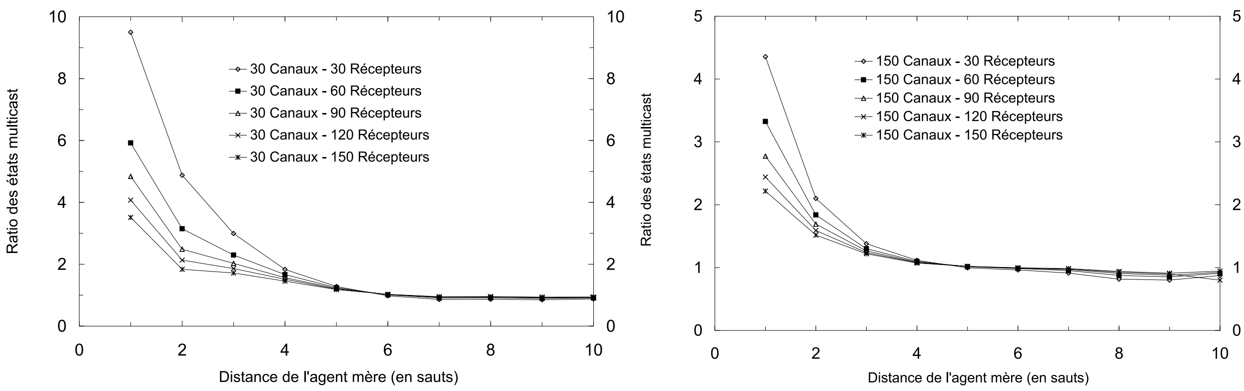

FIG. 3.11 – Ratio E_T/E_V des états multicast

On remarque dans un premier temps que ces résultats sont identiques à ceux obtenus dans la section 3.3.4 (figure 3.5). En effet, si l'on ignore les spécificités des simulations de la section 3.3.4 et celles de la section actuelle, on note que dans les deux cas on effectue des simulations équivalentes. On simule ainsi dans un premier temps la construction d'arbres enracinés au même endroit (utilisation de tunnels enracinés à l'agent mère d'une source ou d'un récepteur mobile), et dans un deuxième temps la construction d'arbres enracinés à des endroits différents dans la topologie (adhésion depuis le réseau visité ou utilisation de MSSMSv6). Etant donné qu'il y a convergence des mesures au cours des simulations, on obtient des mesures identiques dans les deux cas, c'est à dire lors de la superposition d'arbres avec des récepteurs mobiles ou avec des sources mobiles. L'évolution des courbes de la figure 3.11 s'explique avec le même raisonnement que celui utilisé dans la section 3.3.4. Avec l'adhésion au canal (I,G), tous les arbres sont enracinés au réseau mère des sources mobiles et les routeurs du voisinage de l'agent mère ont donc généralement un grand nombre d'états multicast même lorsque le nombre de récepteurs est faible. Les routeurs proches de l'agent mère forment en effet le squelette de l'arbre de diffusion (comme décrit dans la section 2.4.3) et le nombre de récepteurs a une faible influence sur E_T. D'un autre côté, le nombre de canaux a une influence forte sur E_T dans le voisinage de l'agent mère car les routeurs proches de ce dernier sont dans le squelette de presque tous les arbres. Au contraire avec le protocole MSSMSv6, les arbres sont enracinés aux positions courantes des sources mobiles ce qui supprime l'effet de concentration d'états autour de l'agent mère. Le ratio E_T/E_V est donc élevé. Cependant lorsque le nombre de récepteurs augmente, la probabilité pour qu'un routeur proche de l'agent mère fasse partie d'un arbre multicast s'accroît car chaque arbre implique un plus grand nombre de routeurs. On observe donc une diminution du ratio E_T/E_V car la croissance de E_V est supérieure à celle de E_T (qui est en fait presque nulle lorsque le nombre de récepteurs augmente). Enfin lorsque l'on augmente le nombre de canaux, la probabilité pour qu'un routeur proche de l'agent mère fasse partie de plusieurs arbres de diffusion croît considérablement. Le rapport E_T/E_V est donc plus faible lorsque le nombre de canaux est élevé car E_T croît linéairement alors que E_V augmente plus rapidement.

La conclusion de cette section est que le protocole MSSMSv6 permet de minimiser l'effet de concentration des états multicast dans les routeurs du voisinage du routeur mère. D'après la loi de Chuang-Sirbu, on sait qu'un groupe multicast de taille N génère en moyenne un arbre multicast de taille $\overline{L_u}N^k$. Comme chaque routeur de l'arbre a un état de propagation, la construction d'un arbre consiste donc à *distribuer* $\overline{L_u}N^k$ états multicast dans le réseau. Si l'on superpose par exemple G arbres, on va donc distribuer $G * \overline{L_u}N^k$ états multicast dans le réseau. Avec le protocole MSSMSv6 ces états seront donc dispersés car les arbres sont enracinés à divers endroits de la topologie alors qu'avec l'utilisation de tunnels via l'agent mère les arbres sont tous enracinés au réseau mère des sources mobiles. Le protocole MSSMSv6 ne diminue donc pas le nombre d'états multicast mais les distribue de manière uniforme dans le réseau.

3.4.3 Conclusion

Les résultats que nous avons présentés nous ont permis de montrer que le protocole MSSMSv6 que nous proposons permet d'optimiser le routage multicast avec des sources mobiles. Ce protocole permet en effet de construire des arbres de plus courts chemins enracinés à la position courante d'une source mobile. Le routage des données est donc optimal et on évite également la concentration d'états multicast dans les routeurs proches d'un agent mère.

De plus le protocole MSSMSv6 est très simple à mettre en œuvre et induit une surcharge de trafic négligeable.

3.5 Conclusion

Dans ce chapitre, nous avons considéré la mobilité de membres de groupes multicast dans des réseaux sans-fil à infrastructure. Nous avons en particulier évalué et comparé plusieurs solutions permettant de gérer ce type particulier de mobilité. Nous avons également proposé une solution simple permettant de gérer de manière dynamique et décentralisée la mobilité de la source d'un arbre SSM. Enfin, nous avons montré que l'utilisation de tunnels via un agent mère dégrade fortement les performances de la diffusion multicast, et ce avec des récepteurs ou des sources mobiles. Enfin, notre proposition MSSMSv6 s'intègre au modèle de gestion de la mobilité proposé par le protocole Mobile IPv6 dans lequel la mobilité est gérée par les terminaux et non pas par les routeurs du réseau.

Dans la seconde partie de ce mémoire, nous étendons notre travail en considérant les réseaux mobiles ad hoc. La spécificité de ces réseaux est qu'ils ne reposent pas sur une infrastructure de points d'accès. Ces réseaux sont en effet basés sur la collaboration spontanée de nœuds mobiles qui coopèrent activement afin de retransmettre les données entre tous les membres d'un réseau. La mobilité est en particulier une composante intrinsèque de ces réseaux car les déplacements des nœuds modifient de manière permanente la topologie d'un réseau ad hoc.

Deuxième partie

Mobilité, multicast, et continuité de préfixe dans les réseaux ad hoc

Chapitre 4

Contexte de recherche

4.1 Le routage unicast dans les réseaux ad hoc

4.1.1 Introduction

Un réseau ad hoc est un réseau sans-fil constitué d'entités qui collaborent les unes avec les autres afin de faire transiter les données à travers le réseau. Les stations constituant un tel réseau ne sont en effet pas nécessairement à portée radio les unes des autres : il n'est donc pas forcément possible pour une station de pouvoir communiquer directement avec tous les autres hôtes du réseau. La particularité d'une telle architecture est que les stations qui constituent le réseau doivent collaborer afin de permettre à deux machines de communiquer même si elles ne sont pas à portée radio l'une de l'autre. Les données peuvent en effet transiter de *proche-en-proche* d'une station à une autre en utilisant des nœuds intermédiaires du réseau. Cette collaboration mutuelle est indispensable afin de permettre à toutes les stations de communiquer entre elles. Les réseaux ad hoc ont de plus la particularité intrinsèque d'être topologiquement variables dans le temps : les nœuds du réseau sont en effet libres de se déplacer. A tout instant, la position géographique des nœuds du réseau et les caractéristiques du support radio (propagation des ondes, interférences, puissance de transmission/réception) crée un graphe de connectivité entre toutes les entités communicantes du réseau.

La communication multi-sauts dans un réseau ad hoc est donc conditionnée par la mise en place d'un protocole de routage, fonctionnalité *sine qua non* sans laquelle la communication entre deux nœuds n'étant pas à portée radio l'un de l'autre n'est pas possible. Or, les protocoles de routage *classiques* à états de lien ou à vecteurs de distance [94] tels que OSPF [24] et RIP [95] n'ont pas été conçus pour fonctionner dans un environnement topologique aussi variable que celui rencontré dans un réseau ad hoc [96]. Ces protocoles ont en effet été développés pour réagir à des changements topologiques *mineurs* tels que des ruptures de liens ou des pannes de nœuds du réseau. Ils ne sont pas optimisés pour réagir à de fréquents changements topologiques et ne prennent pas en compte les spécificités des réseaux ad hoc. Un groupe de travail a donc été mis en place à l'IETF afin de proposer des solutions de routage adaptées aux particularités des réseaux ad hoc. Le groupe de travail MANET a ainsi vu le jour à la fin des années 90 : son objectif principal est la spécification de protocoles de routage unicast pour les réseaux ad hoc. De plus, la spécificité des réseaux ad hoc impose un certain nombre de contraintes liées à la dynamicité de la topologie et, dans une moindre mesure, à la nature des nœuds du réseau (ex : énergie et puissance limitées). Un protocole de routage unicast pour les réseaux ad hoc doit donc intégrer plusieurs hypothèses et/ou contraintes [97] dont les principales sont que :

- la topologie d'un réseau ad hoc est dynamique et les déplacements des nœuds sont imprévisibles
- les liens physiques (radio) entre deux nœuds peuvent être uni-directionnels ou bi-directionnels
- le fonctionnement du protocole de routage doit être distribué.

La problématique du routage dans les réseaux ad hoc a conduit à la proposition de nombreuses solutions [96, 98]. Cependant, seules quatre propositions ont été retenues par le groupe de travail MANET de l'IETF. Ces quatre protocoles sont en effet les quatre candidats les plus matures pour le routage unicast dans les réseaux ad hoc. Les protocoles AODV [99] et DSR [100] sont *réactifs* et découvrent les routes *à la demande*, c'est à dire lorsqu'un nœud du réseau désire communiquer avec une station pour laquelle il n'a aucune information de routage. Au contraire, les protocoles OLSR [101] et TBRPF [102] sont *proactifs*, dans le sens où chaque station maintient des routes pour tous les nœuds du réseau ad hoc même si cette station ne communique pas avec certains de ces nœuds.

4.1.2 Le protocole DSR

Le protocole DSR (Dynamic Source Routing) [100, 103] est l'un des premiers protocoles ayant été proposés pour le routage dans les réseaux ad hoc. Il s'agit d'un protocole de routage réactif qui repose sur un mécanisme de découverte de routes *à la demande* et qui utilise un modèle de routage *dirigé par la source* : lorsqu'une station du réseau ad hoc émet des données, tous les paquets contiennent la liste de tous les sauts intermédiaires qui composent la route vers la destination. La route utilisée pour transférer les données est donc imposée par la source. Au niveau du routage, le protocole DSR peut être décomposé en deux phases : la phase de découverte des routes et la phase de maintenance.

4.1.2.1 Découverte des routes

Lorsqu'un nœud du réseau veut communiquer avec une autre station du réseau ad hoc, il consulte sa table de routage afin de vérifier si il a une route pour cette destination. Si une route existe, le nœud l'utilise pour transmettre les données. Par contre, si il n'existe aucune route pour cette destination dans la table de routage, le nœud va initier la découverte d'une route en émettant une requête de découverte de route (*route request*) à destination de tous les nœuds du réseau. Cette requête contient en particulier l'adresse du destinataire, un identifiant unique permettant d'identifier la requête, et la liste de tous les nœuds intermédiaires ayant retransmis la requête. Elle se propage en effet de la source vers tous les hôtes du réseau. Chaque nœud qui reçoit cette requête vérifie dans sa table de routage si il a une route pour cette destination. Si le nœud n'a pas de route pour la destination, il retransmet la requête à tous ses voisins et rajoute son adresse dans la liste des nœuds intermédiaires. Afin d'éviter une propagation excessive des requêtes et pour éviter de créer des boucles de routage, chaque nœud retransmet la requête une seule fois grâce à l'utilisation de l'identifiant. Lorsque la requête atteint la destination ou un nœud ayant une entrée pour cette destination dans sa table de routage, une réponse de routage (*route reply*) est envoyée à l'initiateur de la requête. Cette réponse contient en particulier la liste de tous les nœuds intermédiaires ayant propagé la requête initiale ce qui permet à la source de connaître le détail de la route vers la destination. Dans le cas où un nœud autre que la destination renvoie la réponse, celui-ci rajoute la liste des nœuds formant la portion de route située entre lui-même et la destination. Le message renvoyé à l'initiateur de la requête de découverte contiendra donc la route complète entre ce dernier et la destination.

Lors de l'envoi de la réponse de routage, il est possible que le nœud émettant cette réponse n'ait pas de route pour joindre l'initiateur de la requête. Dans ce cas, le nœud émettant la réponse initie une nouvelle procédure de découverte de route et intègre dans la nouvelle requête la réponse à la requête initiale. Cette requête/réponse doit alors impérativement arriver jusqu'à l'initiateur de la première requête et aucun nœud intermédiaire ne peut y répondre : l'objectif premier de toute cette procédure est en effet que l'émetteur de la requête initiale découvre la route vers la destination. Enfin, si les liens entre tous les nœuds du réseau sont bi-directionnels, la réponse peut être envoyée en utilisant l'information de route contenue dans la requête (en fait en utilisant le chemin inverse).

4.1.2.2 Maintenance des routes

Le fonctionnement du protocole DSR dépend de l'aptitude des nœuds à découvrir la rupture de liens dans le réseau ad hoc. En l'occurence, cette détection est uniquement locale [100] : chaque nœud sur une route vérifie si le paquet en transit peut être transmis au prochain saut. En cas de détection d'une rupture de lien, un nœud envoie un message d'erreur (*route error*) à destination de la source des données. Ce message permet à tous les nœuds le retransmettant de supprimer les routes qui utilisaient le lien défaillant.

Les spécifications de DSR [103] proposent d'utiliser différents mécanismes afin de détecter la rupture d'un lien. Un nœud peut ainsi se baser sur les mécanismes de la couche liaison de données (ex : IEEE 802.11) qui permettent d'acquitter une trame (si un tel mécanisme existe). Une deuxième possibilité est d'utiliser un système d'acquittements passifs : lorsqu'un nœud A retransmet des données vers son voisin B, il peut en théorie *entendre* si B réémet bien les données à destination du saut suivant. En pratique, cette technique est cependant difficile à mettre en œuvre car elle implique que chaque nœud puisse détecter l'envoi de données vers des destinations autres que lui-même (ex : passage de l'interface réseau en mode *promiscious*). Enfin avec la troisième méthode, le protocole DSR lui-même peut être utilisé afin d'envoyer des acquittements de proche-en-proche.

4.1.2.3 Optimisations

Le protocole DSR propose également des optimisations intéressantes. Un nœud peut par exemple apprendre de nouvelles routes en retransmettant des données ou des réponses de routage puisque ces deux types de paquets contiennent la liste complète de tous les sauts intermédiaires d'une route. De plus, un mécanisme optionnel permet de déclarer un *flot* de données entre une source et une destination. Avec ce système, la source n'a plus besoin de mettre la liste de tous les sauts intermédiaires dans chaque paquet de données. Un identifiant de flot est en effet attribué à la communication, et est connu par chaque nœud intermédiaire sur la route. La liste des sauts intermédiaires est alors remplacée par un simple identifiant de flot.

4.1.3 Le protocole AODV

Le protocole réactif AODV (Ad hoc On-Demand Distance Vector Routing) [99, 104] est dérivé du protocole DSDV (Dynamic Destination-Sequenced Distance Vector Routing [105]) et il utilise des mécanismes de découverte de routes inspirés du protocole DSR. En effet, alors que DSDV découvre périodiquement les routes pour tous les nœuds du réseau en utilisant une version modifiée d'un protocole de routage à vecteurs de distance, AODV découvre les routes *à la demande*. Le protocole AODV se différencie de DSR par l'utilisation de mécanismes différents pour la découverte et la maintenance des routes. De plus, AODV n'utilise pas de routage *dirigé par la source*.

4.1.3.1 Découverte des routes

Comme avec DSR, la découverte de routes avec AODV se base sur l'envoi d'une requête de routage (*route request ou RREQ*) suivie par la réception d'une réponse de routage (*route reply ou RREP*). Cependant, et contrairement à DSR, la requête de routage ne contient pas la liste des sauts intermédiaires. Avec AODV, chaque nœud intermédiaire retransmettant une requête de routage mémorise l'adresse du nœud duquel il a reçu la requête de routage. AODV introduit également l'utilisation de numéros de séquence qui permettent d'éviter la formation de boucles de routage et qui permettent aux nœuds de connaître *l'état de fraîcheur* des routes, c'est à dire leur validité potentielle. Lorsque la requête atteint la destination ou un nœud ayant une route récente pour la destination, une réponse de routage est renvoyée de *proche-en-proche* à l'initiateur de la requête. Chaque nœud retransmet ainsi la réponse de routage au nœud par lequel il a reçu la requête correspondante. Tous les nœuds sur le chemin emprunté par la réponse de routage créent alors une entrée dans leur table de routage pour la destination en spécifiant comme prochain saut l'adresse du nœud leur ayant envoyé la réponse de routage. Contrairement à DSR, le protocole AODV ne nécessite donc pas l'existence d'une route entre la destination et la source. Cependant, le mécanisme de découverte de routes dans AODV nécessite la présence de liens bi-directionnels entre tous les nœuds du réseau ad hoc.

4.1.3.2 Maintenance des routes

Comme DSR, AODV repose sur l'aptitude des nœuds du réseau ad hoc à détecter la rupture de liens. Comme pour DSR, cette détection est locale avec AODV : chaque nœud vérifie si le prochain saut utilisé pour retransmettre un paquet est joignable ou non. AODV se démarque de DSR en utilisant un mécanisme de découverte des voisins basé sur l'envoi périodique de messages HELLO permettant à chaque nœud de connaître ses voisins. La réception d'un tel message permet en effet à un nœud de savoir que le lien entre lui-même et l'émetteur du message HELLO est actif. De plus afin d'éviter de surcharger le réseau, seuls les nœuds ayant des routes actives envoient des messages HELLO. Si un nœud détecte une rupture de lien, il envoie un message d'erreur (*route error*) à tous ses voisins utilisant des routes dont fait partie le lien rompu. Chaque nœud maintient donc une table contenant la liste de ses voisins qui l'utilisent comme prochain saut pour certaines destinations.

4.1.4 Le protocole OLSR

OLSR (Optimized Link-State Routing) [101, 106] est un protocole de routage proactif à états de lien optimisé pour les réseaux ad hoc. Contrairement aux protocoles de routage réactifs, un protocole proactif maintient en permanence des routes pour toutes les destinations du réseau. Les routes sont donc immédiatement disponibles dès qu'un nœud désire envoyer des données.

OLSR est particulièrement adapté à la nature volatile des réseaux ad hoc car il propose l'utilisation d'un certain nombre d'optimisations par rapport à un protocole *classique* de routage à états de lien. OLSR repose en particulier sur le concept de nœud relais multipoint (MultiPoint Relay ou MPR). Avec un protocole classique de routage à états de lien, tous les nœuds innondent le réseau avec des informations topologiques. Avec OLSR, seuls les nœuds relais multipoints (MPR) réalisent cette opération. L'utilisation des MPR réduit donc fortement la surcharge liée à l'inondation du réseau nécessaire au fonctionnement d'un protocole de routage à états de lien. De plus, une seconde optimisation permet de réduire la taille des messages de contrôle car chaque MPR ne diffuse qu'un sous-ensemble de ses liens avec ses voisins.

4.1.4.1 Découverte du voisinage

Tout protocole de routage à états de lien dépend de l'aptitude des nœuds du réseau à découvrir dynamiquement leur voisinage. Avec OLSR, tout nœud envoie périodiquement un message HELLO qui contient les adresses des nœuds avec lesquels il maintient un lien bi-directionnel, et les adresses des nœuds dont il reçoit des messages HELLO. Lorsqu'un nœud A reçoit d'un nœud B un message HELLO ne contenant pas l'adresse de A, A considère dans un premier temps son lien avec B comme uni-directionnel. Dans son message HELLO suivant, A indique simplement qu'il reçoit des messages HELLO de B. Lorsque B reçoit le nouveau message HELLO envoyé par A, il voit son adresse dans la liste des nœuds entendus par A et en déduit donc que le lien entre A et B est bi-directionnel. B pourra donc annoncer A comme voisin bi-directionnel dans ses prochains messages HELLO, ce qui permettra à A de savoir que son lien avec B est bi-directionnel. Si un nœud ne reçoit pas de messages HELLO d'un de ses voisins pendant une certaine période de temps, il le supprime de sa liste de voisins. Ce mécanisme permet donc à chaque nœud du réseau ad hoc de découvrir son voisinage *à deux sauts* car chaque paquet HELLO contient la liste des voisins de l'émetteur du message : un nœud peut donc apprendre quels sont les voisins de ses propres voisins.

4.1.4.2 Relais multipoints (MPR)

Le protocole OLSR repose sur la notion de relais multipoints [107] définie comme suit. Chaque nœud sélectionne dans son voisinage direct ses MPR de telle sorte que l'ensemble des MPR de ce nœud couvre (en termes de couverture radio) tous ses voisins à deux sauts.

En conséquence, un nœud peut joindre tous ses voisins à deux sauts en utilisant uniquement ses MPR comme sauts intermédiaires. Avec OLSR, plus l'ensemble des MPR d'un nœud est petit, et plus la surcharge liée à l'inondation périodique du réseau est petite. En effet, seuls les MPR initient l'envoi d'informations topologiques. De plus, seuls les MPR retransmettent ces messages. On notera également que la sélection optimale des MPR est un problème NP-complet : les spécifications de OLSR proposent donc des heuristiques permettant de trouver un ensemble quasi-optimal [108].

De plus, lorsqu'un nœud A sélectionne l'un de ses voisins B comme MPR, il annonce ce choix dans ses messages HELLO en indiquant que B est MPR pour A. Le nœud B peut donc apprendre qu'il est MPR pour A. Chaque MPR maintient ainsi une table récapitulant la liste des nœuds l'ayant choisi comme MPR. En particulier, cette liste est utilisée pour construire les messages de découverte de topologie (Topology Control ou TC).

La figure 4.1 illustre l'ensemble des MPR pour un nœud d'un réseau ad hoc. On note que tous les nœuds situés à deux sauts du nœud considéré sont joignables grâce à trois nœuds relais multipoints. Afin d'atteindre tous les voisins situés à deux sauts du nœud considéré, un message sera donc retransmis trois fois lorsqu'on utilise le mécanisme des MPR. En comparaison, le message sera réémis cinq fois si tous les voisins à un saut retransmettent le message (inondation sans optimisation).

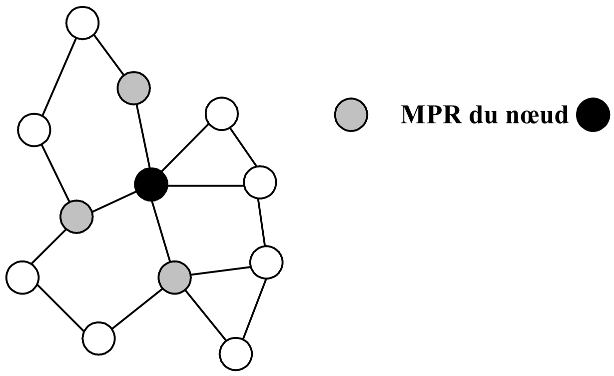

Fig. 4.1 – Ensemble des MPR pour un nœud donné

4.1.4.3 Découverte de topologie et routage

Afin de permettre à chaque nœud d'avoir une vision complète de la topologie du réseau, un protocole de routage à états de lien passe par une phase d'inondation des informations de voisinage. OLSR utilise un mécanisme similaire qui ne permet cependant pas aux nœuds d'avoir une vision complète du réseau. En effet avec OLSR, seuls les MPR envoient des messages de découverte de la topologie (Topology Control ou TC). De plus, ces messages contiennent uniquement la liste des nœuds ayant choisi l'émetteur comme MPR. Les messages TC permettent néanmoins à chaque nœud du réseau de construire une topologie contenant tous les nœuds du réseau (il manquera uniquement certaines arêtes). Chaque nœud sera donc capable de calculer à partir de cette topologie (partielle) une table de routage complète pour toutes les destinations possibles du réseau ad hoc.

4.1.5 Le protocole TBRPF

Comme OLSR, le protocole TBRPF (Topology Dissemination Based on Reverse-Path Forwarding) [102, 109] est une variante du protocole de routage à états de lien. Contrairement à un protocole de routage classique à états de lien, les informations topologiques sont propagées en utilisant un arbre de diffusion minimal enraciné à la source envoyant ces données. Cette méthode permet en particulier de minimiser la surcharge liée à l'inondation du réseau car les feuilles de l'arbre de diffusion ne retransmettent pas inutilement ces données [102]. De plus, la surcharge est minimale dans le sens où seuls les nœuds *stratégiquement* situés retransmettent les données : la méthode est donc en principe similaire à l'utilisation des MPR (OLSR) mais en pratique les deux techniques sont très différentes.

4.1.5.1 Découverte du voisinage

Contrairement à OLSR, tout nœud TBRPF a uniquement besoin de connaître son voisinage direct à un saut. Cette restriction permet notamment à TBRPF d'utiliser un mécanisme minimal pour la découverte des voisins. En effet, les messages HELLO périodiquement envoyés par chaque nœud contiennent uniquement des informations lorsque l'émetteur du message HELLO détecte des changements dans son voisinage. Ce mécanisme est nécessaire car le protocole TBRPF utilise uniquement les liens bi-directionnels du réseau. Lorsqu'un nœud détecte la présence d'un nouveau voisin il doit donc initier une procédure afin de valider le lien comme étant bi-directionnel. Néanmoins, ce mécanisme réduit fortement la taille des messages HELLO quand on le compare à un mécanisme standard comme celui utilisé par OLSR.

Ainsi, si un nœud A reçoit un message HELLO du nœud B pour lequel il n'a aucune information de voisinage, A va rajouter l'adresse de B dans le champ NEIGHBOR REQUEST de son prochain message HELLO. Lors de la réception de ce nouveau message, B trouve son adresse dans le message HELLO envoyé par A et va rajouter l'adresse de A dans le champ NEIGHBOR REPLY de son prochain message HELLO. B aura donc déjà détecté que son lien avec A est bi-directionnel. A la réception du nouveau message HELLO envoyé par B, A détecte également que le lien est bi-directionnel. Dans leurs messages HELLO suivants, les nœuds A et B n'ont donc plus besoin d'échanger d'informations particulières ce qui réduira la taille de leurs futurs messages HELLO. Cependant, si par exemple le nœud A ne reçoit plus de messages HELLO de B pendant un certain laps de temps, il indique ce changement en mettant l'adresse de B dans le champ NEIGHBOR LOST de son prochain message HELLO. Ainsi si le lien est devenu uni-directionnel, B pourra apprendre que A ne reçoit plus ses propres messages HELLO et pourra supprimer A de sa table des voisins bi-directionnels.

4.1.5.2 Diffusion par arbre de recouvrement

L'originalité de TBRPF est qu'il utilise un arbre de recouvrement par nœud afin de diffuser les informations topologiques dans le réseau ad hoc. Cette technique induit en effet une surcharge minimale par rapport à la méthode d'inondation utilisée généralement dans les protocoles à états de lien non optimisés. La technique par arbre de recouvrement tire en effet parti de la nature du support radio qui permet en particulier à un nœud d'envoyer un même paquet à tous ses voisins. De plus lorsqu'un arbre de recouvrement a atteint une taille suffisante permettant de couvrir tous les nœuds du réseau, les feuilles de l'arbre ne réémettent plus les données ce qui diminue fortement la surcharge associée à la diffusion des informations topologiques.

Avec TBRPF, chaque nœud commence par diffuser la liste de ses voisins : chaque nœud va donc apprendre la liste de ses voisins à deux sauts. A partir de là, chaque station va trouver le prochain saut p pour chacun de ses voisins v à deux sauts, et va informer p de ce choix en lui envoyant un message NEW PARENT en indiquant en particulier l'adresse de v. Un arbre dont la racine est v est donc graduellement créé : il va permettre de diffuser les informations de voisinage de v. Il y aura donc un arbre de diffusion par nœud du réseau. Lorsque les informations topologiques commencent à être diffusées dans un arbre, les nœuds proches des feuilles de l'arbre peuvent adhérer à l'arbre de recouvrement en choisissant l'une des feuilles comme père dans l'arbre. Tous les arbres s'étendent donc à chaque étape de diffusion des informations topologiques. Enfin, lorsque l'arbre atteint une taille suffisante permettant de couvrir tous les nœuds du réseau, les feuilles peuvent arrêter de propager les informations. En pratique, une feuille arrête de propager les données pour une source s si elle n'a pas reçu de message NEW PARENT pour s pendant un certain laps de temps. Si une feuille détecte un changement dans son voisinage, elle recommence à propager les informations topologiques pendant un laps de temps prédéfini afin de permettre d'éventuelles nouvelles adhésions à l'arbre.

Avec TBRPF, chaque nœud émet uniquement dans son arbre de recouvrement les informations sur son voisinage direct à un saut. Chaque station du réseau peut déduire de la réception de ces informations la topologie complète du réseau et va en particulier pouvoir réagir à une rupture de lien. Comme pour OLSR, la détection de la rupture d'un lien est locale : chaque nœud détecte en effet la perte de ses voisins. Si une station détecte la rupture d'un lien, elle va utiliser son graphe topologique afin de trouver (si nécessaire) un nouveau père pour tous les nœuds (et arbres) du réseau. La ré-adhésion aux arbres peut donc être très rapide. Cependant, si un nœud ne dispose pas d'informations topologiques suffisantes pour trouver un nouveau père pour un arbre donné, la convergence est plus lente et est proportionnelle à la distance entre ce nœud et le membre de l'arbre le plus proche.

4.1.6 Conclusion

La difficulté du routage unicast dans les réseaux ad hoc est liée principalement à la nature dynamique et imprévisible de la topologie de ces réseaux. Les deux *philosophies* de routage dans un réseau ad hoc, i.e. proactive et réactive, ont chacune leurs avantages et inconvénients. Les protocoles proactifs tels que OLSR et TBRPF maintiennent en effet en permanence des routes vers tous les nœuds du réseau ce qui implique un échange permanent d'informations de routage dans le réseau. L'avantage est que les routes sont toujours disponibles au moment où une station en a besoin. Au contraire, les protocoles réactifs tels que AODV et DSR découvrent uniquement les routes à la demande ce qui diminue substantiellement le volume de trafic lié au routage. Le principal inconvénient de ces protocoles est que la procédure de découverte d'une route induit un délai qui retarde la transmission des données. Il est cependant difficile d'évaluer objectivement les performances précises de ces protocoles [110, 111] pour diverses raisons. La première est que les facteurs influant les performances de ces protocoles sont nombreux [112] et leur influence est parfois difficile à évaluer. Un deuxième problème est lié aux techniques de simulation utilisées et à leur exactitude [113] qui rendent difficile la comparaison entre plusieurs études indépendantes.

Le routage dans les réseaux ad hoc ne s'est bien entendu pas limité aux communications unicast. De nombreuses solutions ont en effet été proposées afin de mettre en place un routage multicast dans les réseaux ad hoc. Dans ce cas, la gestion de la mobilité des nœuds devient une contrainte essentielle contrairement aux cas des réseaux sans-fil à infrastructure qui ne sont que le prolongement de réseaux filaires. Avec les réseaux ad hoc, la mobilité des participants d'une communication de groupe est au cœur des propositions de routage multicast. Nous présentons un certain nombre de solutions qui illustrent les mécanismes utilisés pour le routage multicast dans les réseaux ad hoc.

4.2 Le routage multicast dans les réseaux ad hoc

Dans les réseaux filaires, le routage multicast conduit à la création d'arbres de diffusion qui peuvent être partagés ou spécifiques à une source. Les solutions de routage multicast dans les réseaux filaires sont de plus généralement basées sur un mécanisme d'inondation-élagage ou un mécanisme d'adhésion explicite. La classification de ces protocoles est donc généralement basée sur ces deux critères : le type d'arbre généré et la technique utilisée pour mettre en place les états de retransmission dans les routeurs.

Les protocoles de routage multicast dans les réseaux ad hoc sont classifiés différemment. Premièrement, dans un réseau ad hoc et grâce aux propriétés du support radio, il est possible de créer des arbres de diffusion mais également, sans difficulté supplémentaire, des maillages. Il est en effet possible pour un nœud d'envoyer le même paquet de données à tous ses voisins : il est donc très facile de transformer un arbre en structure maillée sous certaines conditions topologiques. En effet, une station faisant partie d'un arbre de diffusion uni-directionnel ne retransmet généralement que les données transmises par son père dans l'arbre. Dans le cas d'un arbre bi-directionnel, les données sont retransmises par un nœud si elles sont envoyées par son père ou par l'un de ses fils dans l'arbre. Dans les deux cas, les données sont uniquement retransmises si elles ont été envoyées par un voisin du nœud dans l'arbre. Or dans un réseau ad hoc, il suffit de lever cette restriction pour conduire à la formation d'une structure maillée : on autorise un nœud à réémettre les données même si elles n'ont pas été envoyées par l'un de ses voisins dans l'arbre. Un nœud peut en effet recevoir des paquets multicast de l'un de ses voisins topologiques sans que ce nœud soit l'un de ses voisins dans l'arbre de diffusion. Ce procédé est illustré par la figure 4.2. Dans le cas du maillage, le nœud 2 peut retransmettre (et recevoir) les informations envoyées par le nœud 1 même si celui-ci n'est pas l'un de ses voisins dans l'arbre. En cas de rupture du lien entre les nœuds 2 et 3, les données multicast seront toujours reçues par la station 2. La seule difficulté induite par l'utilisation d'un maillage est que chaque nœud doit uniquement retransmettre un paquet de donnée si il ne l'a pas déjà retransmis (pour éviter la formation de boucles de routage). Chaque station doit donc maintenir un cache temporaire des paquets déjà retransmis.

De plus, les arbres ou maillages créés dans un réseau ad hoc peuvent être partagés ou spécifiques à une source. Par contre, le mécanisme permettant la mise en place des états de retransmission dans les routeurs est presque toujours basé sur un système d'annonce et

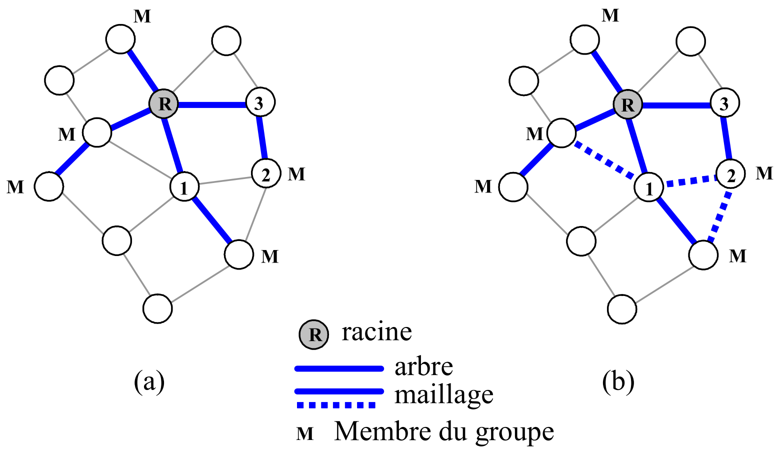

FIG. 4.2 – Arbre (a) et maillage (b) multicast dans un réseau ad hoc

d'adhésion explicite. Les sources et/ou les récepteurs s'identifient en envoyant un message particulier à tous les nœuds du réseau ad hoc, puis l'adhésion est initiée de la source vers les récepteurs ou dans le sens inverse.

Nous présentons donc les solutions de routage multicast selon la classification adoptée généralement [114, 115, 116] en présentant d'une part les protocoles utilisant des arbres pour la diffusion des données, et d'autre part les solutions basées sur des structures maillées. Nous présentons enfin des solutions hybrides qui ne peuvent pas être classées de manière stricte dans ces deux catégories.

4.2.1 Protocoles multicast basés sur des arbres

Dans cette partie nous présentons des protocoles utilisant des arbres pour la diffusion de données multicast dans un réseau ad hoc. L'utilisation d'un arbre uni-directionnel permet en particulier d'éviter naturellement la formation de boucles de routage puisque un nœud retransmet uniquement les données envoyées par son père dans l'arbre. Lors de la réémission des données par l'un de ses fils dans l'arbre, le nœud père ne risque donc pas de retransmettre à nouveau les données. Au contraire, avec un arbre bi-directionnel chaque nœud doit vérifier qu'il ne retransmet pas un paquet plus d'une fois.

4.2.1.1 MAODV

MAODV [117] est une extension du protocole AODV qui rajoute le support des communications multicast. Avec MAODV un arbre multicast est partagé et bi-directionnel : tous les membres de l'arbre peuvent donc émettre des données. Le protocole MAODV utilise des mécanismes de découverte des routes similaires à AODV, et il introduit également de nouveaux messages pour la découverte des groupes, la maintenance de l'arbre, et l'activation des routes multicast.

Avec MAODV, le premier membre d'un groupe devient *group leader* jusqu'à ce qu'il décide de quitter le groupe ou en cas de fusion de deux réseaux ad hoc. Le *group leader* est responsable de l'envoi périodique d'un message GROUP_HELLO contenant un numéro de séquence incrémenté à chaque envoi et un compteur indiquant la distance de l'émetteur par rapport au *group leader*. Ce message est envoyé en broadcast de proche-en-proche et chaque nœud intermédiaire est donc chargé d'incrémenter le compteur de distance avant de réémettre le message GROUP_HELLO. De plus, chaque nœud recevant le message GROUP_HELLO maintient un cache lui permettant de faire l'association entre un groupe et son *leader*.

Lorsqu'un nœud désirant adhérer à un groupe n'a pas d'information dans son cache pour joindre le *leader* du groupe, il envoie en broadcast une requête de routage (RREQ) contenant l'adresse du groupe. Si au contraire le nœud connaît le *group leader*, il lui envoie directement en unicast le message RREQ. Cette requête se propage de proche-en-proche et chaque nœud la retransmettant crée dans sa table de routage multicast une entrée contenant l'adresse du groupe et l'adresse de l'émetteur du message RREQ. Cette entrée reste cependant inactive et elle est supprimée si elle n'est pas activée pendant un certain laps de temps. Lorsqu'un nœud recevant la requête est membre de l'arbre et si son numéro de séquence est au moins égal à celui du message RREQ, il renvoie une réponse de routage (RREP) de proche-en-proche à l'initiateur de la requête. Cette réponse contient en particulier un compteur incrémenté par chaque saut intermédiaire qui permet à l'initiateur de la requête de connaître la distance en sauts à laquelle il est situé par rapport au membre de l'arbre lui ayant répondu. On note que ce procédé permet dans le même temps de mettre en place une route unicast entre l'intiateur de la requête et le membre de l'arbre lui ayant répondu.

L'initiateur de la requête peut alors recevoir plusieurs réponses de routage (RREP) : il attend en effet un certain temps avant de poursuivre la procédure d'adhésion. Une fois ce laps de temps écoulé, il sélectionne la requête avec le numéro de séquence le plus élevé (route la plus récente) et avec le compteur de distance ayant la plus faible valeur (membre de l'arbre le plus proche). La route multicast est alors activée par l'envoi de proche-en-proche d'un message d'activation de route (MACT) au membre de l'arbre ayant envoyé le message RREP sélectionné. Les nœuds intermédiaires retransmettant le message MACT activent l'entrée correspondante de leur table de routage multicast ce qui termine la procédure d'adhésion. Les mécanismes d'adhésion de MAODV sont illustrés par la figure 4.3.

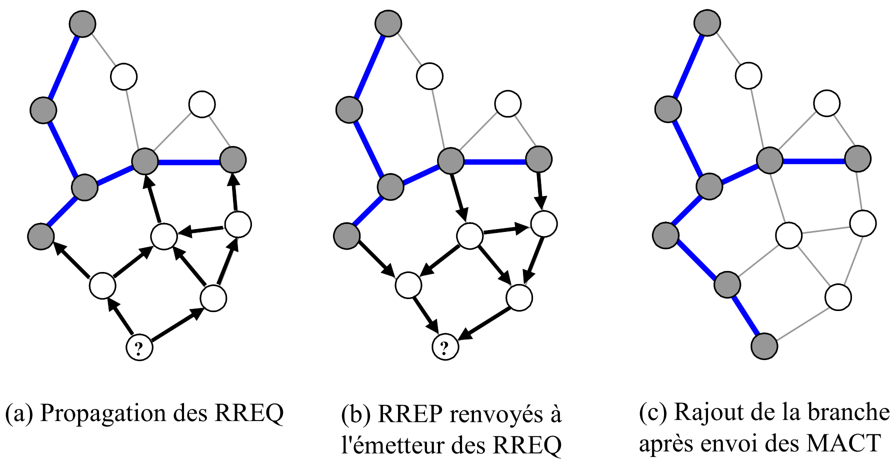

FIG. 4.3 – Mécanismes d'adhésion de MAODV

MAODV permet également de supprimer des branches par l'envoi de messages d'élagage (en fait un message MACT avec un bit particulier positionné à 1). De plus, en cas de rupture de lien, le nœud en aval de la rupture est responsable de la ré-adhésion à l'arbre. Ce nœud relance une procédure d'adhésion en précisant la distance à laquelle il se trouvait avant la rupture par rapport au *group leader* : ce mécanisme évite que les membres du groupe en aval de la rupture répondent à la requête de ré-adhésion. En cas de fusion de deux réseaux ad hoc, une procédure permet également de sélectionner un *group leader* unique pour le nouveau réseau.

L'avantage du protocole MAODV est qu'il utilise un arbre bi-directionnel partagé qui minimise le nombre d'états de retransmission dans les nœuds du réseau. Ce protocole s'intègre également parfaitement dans le protocole AODV avec lequel il partage de nombreux mécanismes. De plus, l'utilisation de numéros de séquence permet d'éviter la formation de boucles de routage. Cependant, l'inconvénient majeur de cette proposition est l'utilisation d'un nœud central, le *group leader*, chargé de maintenir la cohérence au sein de l'arbre de diffusion. Il s'agit en effet d'un point central de panne. De plus, les instabilités liées aux déplacements des mobiles peuvent conduire à de nombreuses ré-élections de *group leader* en cas de partionnement et/ou de fusion du réseau ad hoc. Enfin, le routage entre une source et un récepteur n'est pas forcément optimal car les branches sont construites des récepteurs en direction du *group leader* ce qui peut conduire à un routage triangulaire.

4.2.1.2 MOLSR

Le protocole de routage OLSR a également été étendu pour prendre en compte les communications de groupe. MOLSR [118] utilise en particulier l'aptitude de OLSR à connaître la topologie complète du réseau. Contrairement à MAODV, le protocole MOLSR construit des arbres spécifiques à une source ce qui supprime la complexité liée à l'utilisation d'un nœud central comme le *group leader* dans MAODV. De plus, l'utilisation d'un arbre spécifique à une source assure le routage optimal des données entre la source et les récepteurs. Enfin avec MOLSR tous les routeurs du réseau ad hoc n'ont pas besoin de supporter le routage multicast : il suffit en effet d'un nombre minimal suffisamment grand pour pouvoir *couvrir* tous les récepteurs potentiels.

Avec MOLSR, tout routeur multicast s'annonce comme tel en envoyant un message spécifique MC_CLAIM à destination de tous les nœuds du réseau. On notera que cette diffusion utilise le mécanisme des relais multipoints (MPR) de OLSR afin de minimiser la surcharge induite par cet envoi. L'émission des messages MC_CLAIM permet à tous les nœuds du réseau de connaître la position topologique des routeurs supportant le routage multicast. De plus, chaque source de données multicast diffuse de la même manière un message SOURCE_CLAIM afin que les récepteurs potentiels détectent sa présence. Lorsqu'un nœud recevant un message SOURCE_CLAIM désire adhérer au groupe, il calcule quel est son prochain saut en direction de la source. Contrairement au routage unicast, cette route doit nécessairement passer par des routeurs supportant le routage multicast, c'est à dire par les routeurs ayant envoyé des messages MC_CLAIM. Une fois que le nœud désirant adhérer au groupe a trouvé quel est son prochain saut vers la source, il lui envoie un message d'adhésion CONFIRM_PARENT. Ce message se propage de proche-en-proche vers la source, et crée dans chaque nœud intermédiaire les états nécessaires à la retransmission des données. La propagation du message CONFIRM_PARENT s'arrête lorsqu'il atteint un membre de l'arbre (ou la source). Enfin lorsqu'un membre décide de quitter le groupe, il envoie simplement un message LEAVE à son père dans l'arbre. Ce message se propage jusqu'à atteindre un nœud qui a d'autres fils dans l'arbre ou qui est lui-même membre du groupe.

La maintenance de l'arbre est assurée par l'envoi périodique des messages SOURCE_CLAIM et CONFIRM_PARENT. De plus, lorsqu'un membre d'un arbre détecte un changement topologique, il peut déclencher une nouvelle adhésion si le prochain saut pour atteindre la source a changé. Le nœud envoie alors un message CONFIRM_PARENT au nouveau prochain saut vers la source, et peut (lorsque c'est encore possible) envoyer un message LEAVE à son ancien père dans l'arbre. Des mécanismes supplémentaires de MOLSR permettent également de créer des branches redondantes pour augmenter la fiabilité et la robustesse des arbres.

Un avantage certain du protocole MOLSR est sa simplicité : la complexité liée à l'utilisation d'un arbre partagé est en effet supprimée. De plus, MOLSR construit des arbres spécifiques à une source qui conduisent à un routage optimal entre une source et ses récepteurs. L'inconvénient de ce procédé est cependant la multiplication des états de retransmission dans les routeurs, ce nombre pouvant potentiellement être égal au nombre de sources. Ce point négatif est néanmoins peu pénalisant car limité par la taille relativement petite des réseaux ad hoc. Un inconvénient plus pénalisant de MOLSR est qu'il repose sur l'aptitude des nœuds à détecter les changements topologiques ce qui peut conduire à une réactivité assez lente (de l'ordre de la seconde). Enfin, les incohérences topologiques liées au temps de convergence du protocole OLSR peuvent conduire à la formation temporaire de boucles de routage.

4.2.1.3 AMRIS

Le protocole AMRIS (Ad hoc Multicast Routing protocol utilizing Increasing id-numbersS) [119] utilise des arbres partagés bi-directionnels pour transmettre les données multicast. Afin d'éviter la formation de boucles de routage, AMRIS *assigne* pour chaque groupe un identifiant (noté *msm-id*) à chacun des nœuds du réseau ad hoc. Ce mécanisme repose sur l'élection d'un nœud particulier, le *Sid* (pour Smallest identifier node), responsable de l'émission périodique de messages NEW_SESSION. Un message NEW_SESSION émis par le *Sid* contient en particulier le *msm-id* du *Sid* et l'identifiant de la session multicast. Ce message est propagé de proche-en-proche vers tous les nœuds du réseau. Chaque station recevant ce message construit son propre *msm-id* de telle sorte qu'il soit supérieur à celui du nœud lui ayant envoyé le message NEW_SESSION. De plus, la valeur choisie doit être assez grande pour permettre l'insertion de nœuds entre la station qui vient de recevoir le message NEW_SESSION et l'émetteur de ce message. Lorsqu'un nœud restransmet le message NEW_SESSION, il indique dans ce message la valeur de son *msm-id*. Les identifiants assignés à chaque nœud augmentent donc lorsque l'on s'éloigne du *Sid*.

De plus, chaque nœud maintient une liste des adresses de groupes assignées à chaque *Sid*. Chaque nœud maintient également, grâce à un mécanisme d'échange de messages HELLO, une liste de ses voisins et la valeur de leur *msm-id* pour chaque groupe. Lorsqu'un nœud décide d'adhérer à un groupe, il peut donc trouver lequel de ses voisins a la valeur de *msm-id* la plus faible pour le groupe donné. Etant donné que les valeurs de *msm-id* augmentent en s'éloignant du *Sid*, le nœud désirant adhérer au groupe trouvera donc son voisin P le plus proche du *Sid*. La procédure d'adhésion est alors déclenchée par l'envoi d'une requête d'adhésion (JOIN-REQ) au nœud P. Ce procédé est répété jusqu'à atteindre un membre de l'arbre (ou le *Sid*). Lorsqu'un nœud membre de l'arbre reçoit un message JOIN-REQ, il envoie un message d'acquittement (JOIN-ACK) au nœud lui ayant envoyé la requête JOIN-REQ. Le message JOIN-ACK se propage de proche-en-proche jusqu'à atteindre le nœud ayant initié l'adhésion. Ce message met en place les états de retransmission dans les nœuds intermédiaires. Si la procédure d'adhésion échoue, le nœud désirant adhérer au groupe peut décider d'utiliser une méthode alternative en envoyant la requête d'adhésion à tous ses voisins afin d'augmenter ses chances d'adhérer à l'arbre. De plus un mécanisme supplémentaire permet d'étendre l'envoi de la requête d'adhésion à tous les voisins à R sauts (ou moins) autour de l'émetteur de la requête. La valeur de R est incrémentée à chaque nouvel échec.

L'utilisation d'identifiants est une solution originale permettant de construire des arbres de diffusion en évitant la formation de boucles de routage. De plus, AMRIS est totalement indépendant du protocole de routage unicast utilisé dans le réseau ad hoc. Cependant, ce protocole souffre de l'utilisation d'un nœud central (le *Sid*) qui induit un routage non optimal. De plus, la complexité liée à la sélection du *Sid* pénalise fortement AMRIS [116]. Enfin, les spécifications de ce protocole ne proposent aucun mécanisme concret permettant la sélection dynamique du *Sid* après un partitionnement ou une fusion du réseau. Le dernier point négatif de cette proposition concerne la méthode utilisée par les nœuds pour générer leur *msm-id*. En effet, la technique utilisée pour permettre l'insertion de nœuds dans l'arbre est complexe et peut conduire à des blocages si les déplacements des nœuds sont multiples.

4.2.2 Protocoles multicast basés sur des structures maillées

L'utilisation d'un maillage à la place d'un arbre permet d'augmenter la robustesse de la structure utilisée pour la diffusion des données. Un maillage crée en effet (implicitement) des routes alternatives qui peuvent permettre d'assurer la transmission des données multicast pendant les phases de reconfiguration du protocole de routage multicast (suite à des déplacements de nœuds et/ou à des ruptures de liens). Par contre, tous les protocoles basés sur des structures maillées doivent intégrer un mécanisme pour éviter de former des boucles de routage : chaque nœud retransmet une seule fois un paquet de données et doit donc pouvoir détecter qu'il a déjà reçu un paquet afin d'éviter de le retransmettre.

4.2.2.1 FGMP ET ODMRP

Les deux protocoles FGMP (Forwarding Group Multicast Protocol) [120] et ODMRP (On-Demand Multicast Routing Protocol) [121] sont des variantes d'un même mécanisme qui conduit à la construction d'un maillage partagé par toutes les sources d'un groupe. ODMRP est en fait une variation du protocole FGMP : alors qu'avec FGMP les branches de la structure sont construites de chaque source vers les récepteurs, avec ODMRP elles sont construites des récepteurs vers chacune des sources. Ces deux protocoles reposent également sur l'utilisation d'un protocole de routage unicast proactif qui permet de connaître instantanément le prochain saut pour toutes les destinations du réseau.

Avec FGMP, chaque récepteur envoie en broadcast de proche-en-proche un message de requête d'adhésion. Ce message se propage à tous les nœuds du réseau jusqu'à éventuellement atteindre une source. Chaque source consulte alors sa table de routage afin de trouver son prochain saut vers le récepteur et envoie un message au prochain saut vers le récepteur qui positionne un état de retransmission pour le groupe multicast. Cette procédure se poursuit jusqu'à atteindre un voisin du récepteur. Contrairement au fonctionnement utilisé lors de la construction d'un arbre, les nœuds intermédiaires ne maintiennent pas d'information concernant leur *père* vers la source : ils retransmettent en effet les données quel que soit le nœud qui les leur envoie en évitant uniquement de retransmettre un paquet déjà reçu pour ne pas former de boucles de routage. Avec ODMRP, chaque source annonce son existence en envoyant un message à tous les nœuds du réseau, et les récepteurs initient la construction des branches de la structure maillée vers chacune des sources. Avec les deux protocoles FGMP et ODMRP, les états de retransmission sont maintenus par l'envoi périodique des messages d'annonce et d'adhésion. De plus, aucun de ces protocoles ne proposent de mécanismes permettant l'élagage de branches. Les branches sont en effet supprimées si aucun rafraîchissement des états de retransmission n'a été effectué pendant un laps de temps prédéfini.

L'avantage de ces deux solutions est leur simplicité et leur aptitude à créer des structures potentiellement redondantes. De plus, les maillages ainsi créés sont partagés pour toutes les sources et conduisent à un routage optimal de chaque source vers les récepteurs. En effet, la procédure de construction des branches interdit à un membre de l'arbre d'intercepter les messages d'adhésion car cela pourrait empêcher la création d'une branche vers une source comme illustré par la figure 4.4 : les membres maintiennent en effet uniquement un état de retransmission pour le groupe.

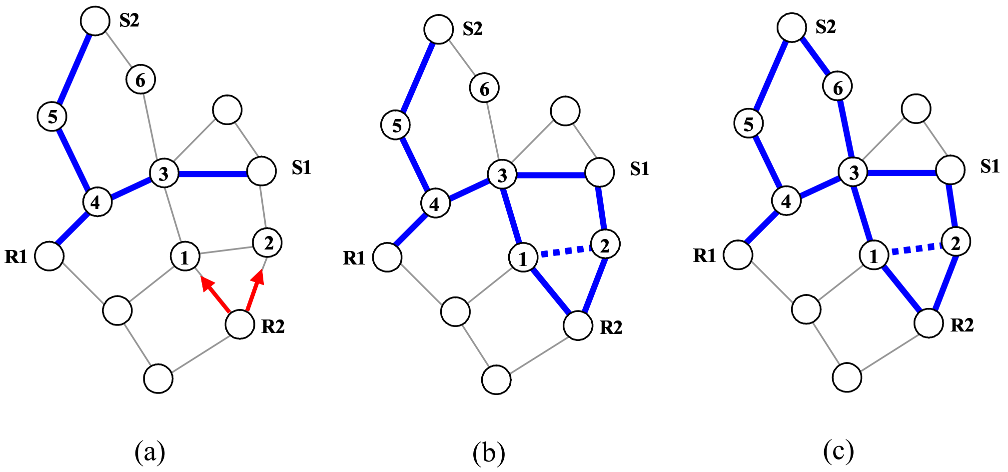

FIG. 4.4 – Mécanismes d'adhésion de ODMRP

Sur la figure 4.4 (a), et si l'on considère le protocole ODMRP, le récepteur R1 a déjà adhéré aux sources S1 et S2. Un deuxième récepteur lance alors l'adhésion vers les deux sources. Dans le cas (b) où l'on autorise un membre de l'arbre à intercepter la requête d'adhésion, la construction de la branche vers S2 s'arrête au nœud 3. Le routage entre S2 et R2 n'est alors pas optimal car le trajet compte 5 sauts. Avec la restriction imposée par ODMRP (c), la requête d'adhésion arrive forcément jusqu'à la source S2 et le routage entre S2 et R2 est optimal et égal à 4 sauts. De plus, la structure en (c) est fortement redondante. On notera également dans les deux cas (b) et (c) la création d'une branche redondante entre les nœuds 1 et 2.

Les avantages de la technique utilisée par FGMP et ODMRP sont nombreux. Premièrement, la structure maillée ainsi créée est généralement fortement redondante et les interruptions de diffusion du trafic multicast (suite aux déplacements des nœuds et à la rupture des liens) sont minimisées [116]. De plus, le routage entre chaque source et les récepteurs est optimal. Enfin, la méthode utilisée pour construire les branches est relativement simple. Cependant, la construction d'une structure fortement maillée induit également certains inconvénients. Le maillage créé par FGMP et ODMRP implique en effet un plus grand nombre de nœuds si l'on compare cette structure avec un arbre. De plus, il n'est pas possible de procéder à un élagage de branche car les nœuds maintiennent uniquement un état de retransmission pour le groupe. Par exemple sur la figure 4.4 (c), si le récepteur R1 décide de quitter le groupe et si l'élagage était autorisé, cette procédure supprimerait sans influence néfaste les nœuds 4 et 5 du maillage. Par contre, avec la suppression de la branche entre R1 et S1, le nœud 3 ne pourrait plus retransmettre les données envoyées de la source S2 vers R2. L'opération d'élagage est donc inexistante dans FGMP et ODMRP.

4.2.2.2 CAMP

Le protocole CAMP (Core-Assisted Mesh Protocol) [122] crée un maillage partagé par toutes les sources d'un groupe multicast. CAMP s'inspire de certains mécanismes du protocole CBT [123]. Il utilise en particulier un nœud central, le *core*, afin de limiter la diffusion des messages d'annonce et d'adhésion (par opposition avec les protocoles FGMP et ODMRP). De plus, CAMP repose sur l'utilisation d'un protocole de routage unicast proactif permettant de connaître le prochain saut vers toutes les destinations du réseau ainsi que la distance (en sauts) entre toutes les paires de nœuds du réseau.

Avec CAMP, chaque membre du maillage informe périodiquement ses voisins de sa participation à la diffusion des données multicast pour un groupe donné. En conséquence, lorsqu'un nœud décide d'adhérer à un groupe, il vérifie si l'un de ses voisins fait déjà partie de la structure maillée. Dans l'affirmative, le nœud informe ce voisin de son adhésion en lui envoyant un message CAMP_UPDATE. Si au contraire il n'existe pas de voisin déjà membre du maillage, le nœud désirant rejoindre le groupe envoie un message d'adhésion JOIN_REQUEST au *core* du groupe. Chaque *core* informe en effet tous les nœuds du réseau de son existence en envoyant périodiquement un message en broadcast. Enfin, si le nœud désirant adhérer au groupe ne connait pas de *core*, il envoie sa requête d'adhésion en broadcast mais en utilisant une méthode basée sur une recherche en anneaux concentriques : la requête est envoyée à tous les voisins à n sauts avec $n = 1$ au départ. À chaque échec, la valeur de n est incrémentée. Contrairement à FGMP et ODMRP, des membres de l'arbre peuvent donc répondre aux requêtes d'adhésion ce qui permet une adhésion plus rapide. Enfin, et afin de maintenir un routage optimal des données entre les sources et les récepteurs, chaque récepteur vérifie régulièrement si son prochain saut vers chacune des sources est bien présent dans la structure de diffusion. Si ce test échoue, le récepteur force l'adhésion de son prochain saut vers la ou les sources correspondantes en lui envoyant un message PUSH_JOIN. Ce message est propagé de proche-en-proche jusqu'à atteindre un nœud de l'arbre dont le prochain saut vers la source fait déjà partie du maillage. Tous les nœuds intermédiaires retransmettant ce message adhèrent bien entendu à la structure maillée.

Un avantage important de CAMP par rapport à FGMP et ODMRP est qu'il autorise l'interception des requêtes d'adhésion par les nœuds faisant partie de la structure maillée. Cette optimisation est en effet possible car chaque membre du maillage vérifie périodiquement, pour chacune des sources, que le prochain saut vers la source fait bien partie de la structure de diffusion. La nature optimale du routage des données est donc régulièrement vérifiée. Le second avantage de CAMP est qu'il évite la diffusion des messages d'annonces : les sources ou les récepteurs n'ont en effet pas besoin d'envoyer un message à tous les nœuds du réseau pour pouvoir se mettre en relation. Cependant, CAMP présente un certain nombre d'inconvénients. Premièrement, l'utilisation d'un nœud central est pénalisante car elle augmente la taille du maillage : chaque récepteur construit en effet une branche vers le *core*, puis optimise le routage en construisant une branche directe vers chaque source. La branche vers le *core* est cependant maintenue car elle permet de recevoir les données de nouvelles sources. L'utilisation de deux branches accroît donc la taille du maillage : la redondance est certes supérieure mais la surcharge de trafic inutile augmente considérablement. De plus, en cas de panne ou d'absence de *core*, l'adhésion par recherche en anneaux concentriques peut s'avérer très coûteuse si le premier membre appartenant au maillage est éloigné du nœud initiant la procédure d'adhésion. Enfin, les auteurs de CAMP ne précisent à aucun moment comment sont sélectionnés les *core* : il est pourtant crucial de pouvoir faire cette sélection de manière dynamique et décentralisée.

4.2.3 Solutions hybrides

Les deux solutions hybrides que nous présentons n'utilisent pas le modèle multicast *traditionnel* inspiré du travail de Deering [17]. Bien que la méthode de diffusion des données soit similaire à celle résultant de la construction d'un arbre, ces deux solutions se démarquent de ce schéma standard. La première proposition utilise en effet des tunnels unicast, et la seconde utilise des listes explicites de destinataires.

4.2.3.1 AMRoute

Le protocole AMRoute (Adhoc Multicast Routing) [124] propose de mettre en place des tunnels unicast entre les membres d'un groupe afin de former un arbre de diffusion. AMRoute crée ainsi des arbres bi-directionnels partagés en utilisant un maillage de contrôle pour maintenir et mettre à jour l'arbre de diffusion. Un nœud particulier, le *core*, initie la création de l'arbre mais contrairement à d'autres solutions utilisant un nœud central, le *core* de AMRoute ne centralise pas la diffusion des données. La sélection du *core* est de plus dynamique et décentralisée.

Avec AMRoute, lorsqu'un nœud décide d'adhérer à un groupe il devient le *core* d'un maillage dont il est le seul membre. Tout *core* émet périodiquement des messages JOIN_REQ en utilisant une méthode basée sur une transmission par anneaux concentriques. Quand un nœud faisant déjà partie d'un maillage reçoit un message JOIN_REQ d'un *core* ne faisant pas partie de sa propre structure maillée, il transmet un message JOIN_ACK à l'émetteur. L'émetteur et le destinataire du message JOIN_ACK deviennent alors des voisins dans le nouveau maillage. Si plusieurs nœuds répondent à l'envoi du message JOIN_REQ, chaque réponse peut conduire à la création d'une nouvelle branche dans le maillage. En pratique, l'émetteur du message JOIN_REQ n'acceptera qu'un nombre limité de réponses JOIN_ACK. Enfin, une branche du maillage se traduit par la mise en place d'un tunnel bi-directionnel entre les deux nouveaux voisins dans le maillage. La nouvelle structure ainsi créée comprend alors deux *core* qui émettent chacun des messages TREE_CREATE déclenchant la création de l'arbre qui va servir à la diffusion des données. Lorsqu'un membre du maillage reçoit des messages TREE_CREATE de plusieurs *core*, il retransmet uniquement les messages du *core* dont l'adresse IP est la plus petite. Ainsi, seuls les messages d'un *core* sont propagés après une courte période de convergence. Les autres *core* arrêtent alors de transmettre des messages TREE_CREATE.

Après cette période de transition, il existe donc un maillage entre les membres du groupe dont l'un est sélectionné comme *core* du groupe. Chaque arrête du maillage est un tunnel bi-directionnel entre ses deux extrémités. Avec AMRoute, la diffusion des données ne se fait pas à travers le maillage car celui-ci est construit de manière chaotique et n'est donc pas optimisé en terme de routage. L'émission des messages TREE_CREATE par le *core* déclenche ainsi la création d'un arbre de diffusion bi-directionnel dont la racine est le *core*. Chaque membre du maillage recevant ce message note en effet de quel voisin (dans le maillage) il a reçu ce message qu'il retransmet alors à ses autres voisins dans la structure maillée. Si un nœud reçoit un message TREE_CREATE qu'il a déjà reçu (ces messages contiennent un numéro de séquence permettant de les identifier), il supprime cette boucle en envoyant un message TREE_CREATE_NAK au voisin lui ayant retransmis le message dupliqué. La détection des messages TREE_CREATE dupliqués déclenche donc l'élagage des branches du maillage conduisant à la formation de boucles. Après convergence, un arbre sans boucle dont la racine est le *core* est créé. L'arbre est maintenu grâce à l'envoi périodique des messages TREE_CREATE et TREE_CREATE_NAK. Les données multicast sont envoyées dans les tunnels unicast : les nœuds topologiques intermédiaires entre deux extrémités d'un tunnel ne maintiennent donc aucun état de retransmission particulier au multicast.

L'avantage principal du protocole AMRoute est l'utilisation de tunnels unicast entre les membres du maillage (et de l'arbre). Les déplacements des nœuds dans le réseau ad hoc n'induisent en effet aucune reconstruction de l'arbre, sauf si il y a partitionnement du réseau ou fusion de plusieurs maillages. De plus, la méthode de sélection dynamique du *core* est simple et ne pénalise pas le fonctionnement de ce protocole. Cependant, l'utilisation de tunnels est également la cause d'un inconvénient majeur. Lors des déplacements des membres du groupe, le maillage (et donc l'arbre) reste identique et la structure peut rapidement devenir très différente de la topologie physique sous-jacente. Les performances en terme de routage peuvent donc se dégrader très rapidement car le protocole AMRoute *subit* les changements topologiques sans réagir. Pour contrer cet inconvénient, il faudrait proposer un mécanisme permettant de reconstruire le maillage lorsque celui-ci devient obsolète par rapport à la topologie physique. Il serait en effet possible de détecter les arêtes non-optimales en mesurant la distance topologique (en sauts) entre les deux extrémités d'un tunnel, soit en utilisant les informations du protocole de routage unicast, soit en utilisant un mécanisme supplémentaire d'écho/réponse.

4.2.3.2 LGT

Le protocole LGT (Location Guided-Trees) [125] est basé sur une méthode inspirée du protocole XCast [66] déjà mentionné dans la section 1.4.6. LGT propose en effet d'ajouter à chaque paquet multicast la liste complète des destinataires du paquet. Chaque nœud intermédiaire recevant un tel paquet regroupe les destinataires contenus dans la liste selon le prochain saut utilisé pour les atteindre. Chaque groupe de destinataires conduit à l'émission d'un nouveau paquet envoyé au prochain saut utilisé pour atteindre ce groupe de destinataires. La liste des destinataires du nouveau paquet est bien entendu mise à jour par chaque nœud intermédiaire. Ce principe de fonctionnement impose l'utilisation d'un protocole de routage proactif qui permet à chaque nœud de connaître la topologie complète du réseau. La procédure de découverte des routes induite par un protocol réactif introduirait en effet un délai inacceptable dans le fonctionnement du protocole LGT.

Comme indiqué par ses auteurs [125], ce protocole est uniquement adapté à de petits groupes. Son avantage est qu'il ne nécessite pas la mise en place d'états de retransmission dans les nœuds du réseau ad hoc. De plus, ce protocole n'est pas influencé par la mobilité des nœuds du réseau puisqu'il n'y a pas à proprement parler de construction d'un arbre de diffusion : les nœuds ne maintiennent en effet aucun état de retransmission. Cependant, ce protocole souffre de plusieurs inconvénients. Premièrement, le fonctionnement de LGT dépend totalement de l'aptitude du protocole de routage unicast à réagir face aux changements topologiques dans le réseau. De plus, LGT peut conduire à la formation temporaire de boucles de routage pendant les phases de convergence du protocole de routage unicast. Enfin, ce protocole n'utilise absolument pas les propriétés de diffusion du support radio qui permettent à un nœud d'envoyer un même paquet à tous ses fils dans l'arbre. On citera également le protocole DDM (Differential Destination Multicast) [126] qui fonctionne sur le même principe que LGT.

4.2.4 Conclusion

Comme pour le routage unicast, la principale difficulté liée au routage multicast dans les réseaux ad hoc est la gestion de la mobilité des nœuds du réseau. Les déplacements des membres d'une structure de diffusion (arbre ou maillage) conduisent en effet à un routage non-optimal et à la formation temporaire de boucles de routage. Ils induisent de plus une surcharge liée aux mécanismes permettant la reconfiguration de la structure de diffusion. En ce qui concerne la diffusion des données, les protocoles de routage multicast pour les réseaux ad hoc conduisent généralement à la création d'un arbre ou d'un maillage. Un maillage a l'avantage de mieux résister aux ruptures de liens grâce à la mise en place d'arêtes redondantes. De plus, le coût de mise en place et de maintenance d'un maillage n'est pas forcément plus important que celui induit par la formation d'un arbre de diffusion. La nature du support radio permet en effet à un nœud d'émettre un même paquet à plusieurs de ses voisins ce qui permet la création implicite de liens redondants [120, 121]. L'inconvénient principal d'un maillage est la création potentielle de boucles de routage : les solutions basées sur ce type de structure utilisent donc obligatoirement un mécanisme permettant d'éviter à un nœud de retransmettre un paquet dupliqué. L'évaluation objective des performances de ces protocoles reste cependant difficile car les conditions de simulation (vitesse de déplacement des nœuds, densité topologique, méthodologie employée) influent grandement sur les performances. On assiste ainsi à la publication de résultats qui semblent parfois contradictoires [116, 122]. En fait, il est clair qu'il n'existe pas de solution optimale adaptée à toutes les variantes de réseaux ad hoc : la topologie de ces réseaux est en effet par nature imprévisible et les simulations restent basées sur des hypothèses simplificatrices qui permettent uniquement d'évaluer les protocoles de routage multicast dans certaines limites. Il semble cependant que les structures de diffusion basées sur un maillage soient mieux adaptées à la transmission de données multicast dans un réseau ad hoc.

Nous avons jusqu'à présent considéré les routages unicast et multicast dans les réseaux ad hoc *autonomes*, c'est à dire pour des réseaux coupés du réseau global Internet. Or, il est généralement admis que les réseaux ad hoc ne resteront pas isolés car leurs utilisateurs voudront avoir accès aux services disponibles à partir des réseaux filaires de l'Internet. L'interconnexion d'un réseau ad hoc avec l'Internet est donc une priorité pour les utilisateurs et les opérateurs de ces réseaux. L'accès à l'Internet est en effet une condition forte au déploiement massif et à la popularité des réseaux ad hoc. Cet accès entraîne cependant une contrainte supplémentaire : il devient indispensable de mettre en place un adressage global dans le réseau ad hoc. Les nœuds du réseau ad hoc doivent en effet être joignables à partir de stations de l'Internet. Leurs adresses doivent donc être uniques et globalement joignables à partir de l'Internet. Nous présentons dans la prochaine section la problématique de l'adressage global dans les réseaux ad hoc, et décrivons un certain nombre de propositions cherchant à résoudre les problèmes liés à l'interconnexion d'un réseau ad hoc avec l'Internet.

4.3 Autoconfiguration d'adresses dans un réseau ad hoc hybride

Un réseau ad hoc *hybride* (illustré par la figure 4.5) est un réseau ad hoc possédant une ou plusieurs connexions vers l'Internet. Cette interconnexion nécessite en particulier la mise en place d'un adressage global au sein du réseau ad hoc. Les stations de l'Internet doivent en effet être capables de joindre les nœuds du réseau ad hoc. L'adresse d'un nœud du réseau ad hoc doit donc être unique et globalement routable dans l'Internet : à partir de toute machine de l'Internet, il doit exister une route permettant de retransmettre des données jusqu'à la passerelle qui connecte le réseau ad hoc avec l'Internet. Le routage de la passerelle jusqu'au nœud du réseau ad hoc est ensuite assuré par le protocole de routage déployé au sein du réseau ad hoc (voir section 4.1). Plus précisément, une passerelle est une station possédant une interface radio connectée au réseau ad hoc et une seconde interface généralement filaire qui offre une connectivité vers l'Internet. On considère de plus qu'une passerelle est fixe : elle est utilisée par les nœuds du réseau ad hoc comme un *point d'accès* vers l'Internet.

Bien que le protocole IPv6 pour l'Internet Nouvelle Génération intègre dans ses spécifications des mécanismes d'autoconfiguration d'adresses, ils ne sont pas directement applicables dans les réseaux ad hoc. Le protocole d'autoconfiguration d'adresses de IPv6 a en effet été conçu pour être utilisé dans un réseau à diffusion, c'est à dire avec un support physique permettant l'émission, sans routage, d'un même message à l'ensemble des nœuds du réseau. Or, les réseaux ad hoc sont par définition multi-sauts. De plus, les mécanismes d'autoconfiguration d'adresses de IPv6 ne sont pas adaptés à la nature dynamique et imprévisible de la topologie des réseaux ad hoc. Enfin, ils ne permettent pas de gérer la présence de passerelles multiples.

Dans cette section, nous présentons dans un premier temps les mécanismes d'autoconfiguration d'adresses du protocole IPv6. Nous décrivons en particulier les limitations de ce protocole dans le cadre des réseaux ad hoc. Nous décrivons également plusieurs solutions permettant l'autoconfiguration des nœuds dans les réseaux ad hoc et étudions leurs avantages et inconvénients, ainsi que leurs limitations.

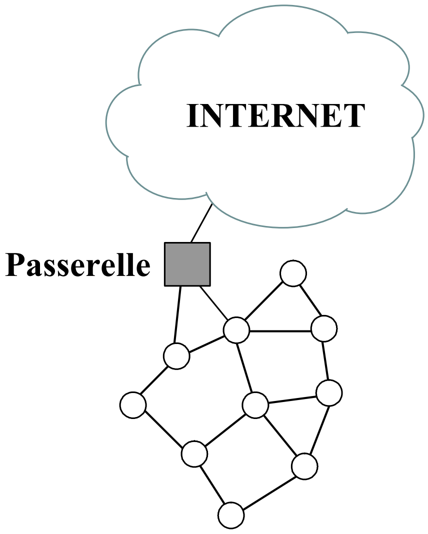

FIG. 4.5 – Réseau ad hoc bybride

4.3.1 L'autoconfiguration d'adresses avec IPv6

Le protocole IPv6 intègre un mécanisme d'autoconfiguration d'adresses sans état permettant de s'affranchir d'un système centralisé comme DHCP [127]. Les protocoles de découverte des voisins [14] (Neighbor Discovery) et d'autoconfiguration d'adresses sans état [128] permettent notamment à toutes les stations d'un réseau IPv6 de configurer automatiquement leurs adresses IPv6. Le protocole de découverte des voisins est également utilisé pour vérifier l'unicité d'une adresse et pour réaliser l'association entre une adresse physique Ethernet et une adresse réseau IPv6.

La configuration automatique sans état des équipements du réseau est l'un des attraits principaux du protocole IPv6. En pratique, elle est mise en œuvre de la façon suivante. Chaque routeur IPv6 émet périodiquement des messages ICMP6 [129] d'annonce de routeur sur les liens physiques auxquels il est connecté. Afin de permettre la configuration automatique des stations du lien, ces messages contiennent une option d'information sur le préfixe qui contient en particulier le préfixe réseau (généralement 64 bits) utilisé sur le lien. La procédure utilisée par chaque station du lien pour générer son adresse globale est alors similaire à la méthode utilisée pour la création de l'adresse lien-local (voir section 1.1.2). Chaque station du réseau utilise en effet l'identifiant d'interface EUI-64 de 64 bits qu'il concatène au préfixe annoncé par le routeur du lien. L'adresse IPv6 de 128 bits ainsi créée contient les 64 bits du préfixe réseau et les 64 bits de l'identifiant d'interface EUI-64. L'unicité de cette adresse est normalement validée car la station a déjà vérifié l'unicité de l'adresse lien-local (et donc de l'identifiant d'interface EUI-64) lors de la création de l'adresse lien-local. L'avantage de cette solution est qu'elle permet de configurer toutes les stations d'un réseau sans avoir à intervenir individuellement sur chacune des machines. Cependant, cette technique nécessite l'utilisation d'un réseau physique à diffusion car le message d'annonce du routeur doit pouvoir atteindre toutes les machines du réseau sans routage. Le protocole de découverte des voisins interdit en effet le routage des messages d'annonce de routeur. De plus, les messages d'annonce de routeur sont également utilisés par chaque station pour configurer la route par défaut de leur table de routage. Le routeur du lien est en effet choisi comme prochain saut par défaut pour toutes les destinations extérieures au réseau local.

La technique d'autoconfiguration d'adresses sans état de IPv6 n'est malheureusement pas applicable en l'état aux réseaux ad hoc. Ces réseaux sont en effet par nature multi-sauts, c'est à dire qu'il n'existe pas un lien physique commun à toutes les stations du réseau. Dans un réseau ad hoc multi-sauts, un message doit être routé pour atteindre tous les nœuds du réseau mais ceci est interdit avec les messages d'annonce de routeur. De plus, le protocole d'autoconfiguration d'adresses ne permet pas de gérer de manière optimale la présence de plusieurs passerelles, ou l'annonce de plusieurs préfixes. Or, un réseau ad hoc hybride peut avoir plusieurs nœuds passerelles vers l'Internet, et ces passerelles peuvent être connectées à des réseaux différents. Enfin, la mise en place de la route par défaut est une procédure délicate dans un réseau ad hoc car elle peut conduire à des incohérences dans le routage. Ces constatations ont conduit à la proposition d'une version modifiée du protocole d'autoconfiguration d'adresses sans état adaptée aux caractéristiques des réseaux ad hoc.

4.3.2 L'autoconfiguration IPv6 adaptée aux réseaux ad hoc

Weniger et al. [130] proposent d'adapter le protocole d'autoconfiguration d'adresses sans état de IPv6 aux réseaux ad hoc. Avec leur solution, un réseau ad hoc est divisé en sous-réseaux utilisant chacun un préfixe différent. Un nœud central, le *leader*, est responsable de la création du préfixe et de son annonce dans le sous-réseau. Cette solution est cependant uniquement appliquée aux réseaux ad hoc autonomes, c'est à dire sans connexion vers l'Internet.

Les auteurs proposent dans un premier temps de modifier les messages de sollicitation de voisin afin de leur permettre d'atteindre des voisins distants au maximum de r_s sauts de l'émetteur du message. En conséquence, un nœud du réseau ad hoc a le droit de propager (donc de router) les messages de sollicitation de voisin à condition qu'ils n'aient pas traversé plus de r_s sauts. Chaque nœud peut donc créer son adresse lien-local, et vérifier périodiquement son unicité dans un rayon de r_s sauts autour de lui. On notera que pour permettre ce test, cette proposition modifie de plus le format des messages de sollicitation de voisin car les nœuds du réseau doivent pouvoir différencier ces messages. En effet, un nœud peut recevoir plusieurs fois le même message mais ne doit le retransmettre qu'une fois pour éviter d'inonder le réseau. Or, deux messages envoyés par des stations ayant créé la même adresse lien-local sont identiques car ils utilisent comme adresse source l'adresse unicast non spécifiée (::). Un identifiant aléatoire est donc rajouté au paquet par l'émetteur d'un message de sollicitation de voisin.

Avec cette proposition, chaque nœud crée une adresse lien-local et vérifie son unicité dans un rayon de r_s sauts autour de lui. De plus, chaque nœud indique dans ses messages de sollicitation de voisin combien il a de voisins directs : chaque station peut en effet détecter un voisin direct lorsqu'elle reçoit un message de sollicitation dont le compteur de nombre de saut est égal à r_s (le paquet n'a pas encore été ré-émis). En conséquence, chaque nœud peut détecter quelle station, dans un rayon de r_s sauts autour de lui, a le plus grand nombre de voisins. Lorsqu'un nœud détecte qu'il est lui-même le nœud ayant le plus grand nombre de voisins, il devient *leader*.

Chaque *leader* crée un préfixe réseau et annonce ce préfixe à tous les nœuds dans un rayon de r_s sauts autour de lui. Un message d'annonce de routeur est en effet envoyé à destination de tous les nœuds du réseau mais avec un compteur de saut initial égal à r_s. Ce compteur est décrémenté avant chaque retransmission : le message se propage jusqu'à maximum r_s sauts du *leader* (un nœud détruit le message si le compteur est égal à zéro). Tous les nœuds dans un rayon de r_s sauts autour du *leader* peuvent donc créer une adresse globale. Comme chaque nœud a vérifié l'unicité de son adresse lien-local dans un rayon de r_s sauts autour de lui-même, il sait que l'adresse globale est unique dans les mêmes conditions. Par contre, l'ensemble des nœuds à r_s sauts du *leader* n'est pas forcément identique au voisinage (à r_s sauts) de l'un des nœuds de cet ensemble. Un nœud ne peut donc pas vérifier l'unicité de son adresse globale dans l'ensemble du *leader*. Chaque nœud doit donc émettre un message de sollicitation pour son adresse globale, et le *leader* doit propager ce message en ré-initialisant le compteur du nombre de sauts à r_s. L'unicité de l'adresse globale peut donc être vérifiée dans un rayon de r_s sauts autour du *leader*. Enfin, l'unicité d'un préfixe créé par un *leader* est vérifiée par celui-ci en émettant un message (modifié) de sollicitation à tout le réseau ad hoc. Seuls les nœuds *leader* ont le droit de répondre à ces messages particuliers dans le cas où un préfixe dupliqué est détecté. Un tel évènement conduit à une ou plusieurs nouvelles créations de préfixes.

L'avantage principal de cette solution est qu'elle utilise des messages et des mécanismes proches du protocole original d'autoconfiguration sans état de IPv6. Cependant, cette proposition souffre de nombreux problèmes. Premièrement, nous avons remarqué qu'il est possible qu'un nœud n'ait pas de *leader* dans son voisinage à r_s sauts. En effet, les ensembles de voisinage à r_s sauts ne sont pas forcément identiques pour chaque nœud ce qui peut conduire à la situation suivante illustrée par la figure 4.6. Pour un nœud n donné, soit $E(n)$ l'ensemble de ses voisins à r_s sauts. Pour un nœud $k \neq n$ de $E(n)$, il peut exister un nœud l *leader* de $E(k)$ tel que $l \notin E(n)$ et $n \notin E(l)$. Si n pense que k est *leader* de $E(n)$, mais que l est *leader* de $E(k)$, le nœud k ne décidera pas de devenir *leader* de $E(k)$ ni donc de $E(n)$. Si ce scénario se répète pour tous les nœuds de $E(n)$, il n'y aura pas de leader dans $E(n)$ et n ne pourra pas construire d'adresse globale.

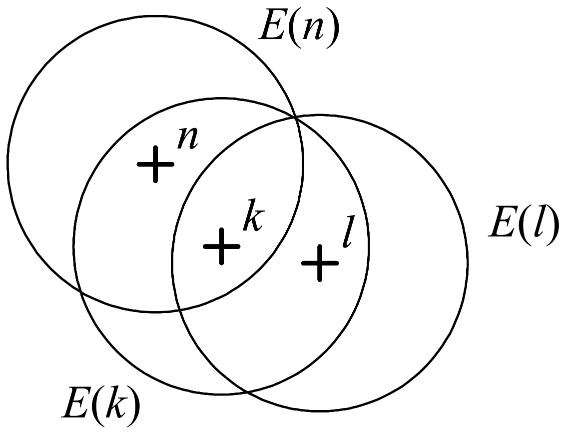

FIG. 4.6 – Problème d'élection de *leader*

Deuxièmement, un des objectifs de cette proposition est de limiter l'envoi de messages [130]. Or, ce protocole nécessite l'émission d'un grand nombre de messages. En effet, dans un réseau ad hoc contenant u nœuds, l'envoi des messages de sollicitation de voisin lors de la phase de création des adresses locales génère au minimum u messages (chaque station transmet son message et $r_s = 1$). Si l'on considère que cette phase conduit à l'élection de v nœuds *leader*, l'envoi, par chaque *leader*, des messages d'annonce de routeur et des messages destinés à vérifier l'unicité des préfixes génère au minimum $v + u$ messages (si $r_s = 1$). Enfin, la phase de vérification de l'unicité des adresses globales génère à nouveau $u + v$ messages (dans le meilleur cas). On aura donc un total minimum de $3u + 2v$ messages, alors qu'en comparaison, une inondation totale du réseau est faite en u messages dans le pire des cas (quand chaque nœud retransmet une fois le message). De plus, si $r_s > 1$, le nombre de messages envoyés sera supérieur à $3u + 2v$ car tous les paquets seront ré-émis par tous les voisins à $r_s - 1$ sauts de l'émetteur d'un message. Enfin, le processus d'élection du *leader* converge en r_s phases du protocole : c'est le temps qu'il faut à chaque nœud pour apprendre le degré (le nombre de voisins) de chaque station de son voisinage à r_s sauts. Le temps de convergence augmente donc avec la taille des sous-réseaux créés, et la convergence est d'autant plus difficile lorsque les déplacements de nœuds sont nombreux. En pratique, le déploiement de cette proposition est donc difficilement envisageable. Des méthodes plus simples basées sur l'utilisation de l'inondation ont donc été proposées.

4.3.3 Autoconfiguration par requêtes et inondations

Deux propositions très proches [131, 132] tendent à adapter les mécanismes d'autoconfiguration de IPv6 aux réseaux ad hoc en utilisant des mécanismes basés sur l'inondation du réseau. Ces deux solutions étant très proches, nous en présentons tout d'abord les points communs puis décrivons les quelques particularités de chacune.

Avec ces deux solutions, chaque nœud du réseau peut créer une adresse locale au réseau ad hoc en utilisant un préfixe spécifique dont la portée est limitée à tous les nœuds du réseau ad hoc (fec0:0:0:ffff::/64). Ces adresses *locales au site*[1] sont uniquement routables dans le réseau ad hoc et ne peuvent donc servir qu'à une communication entre deux nœuds du réseau ad hoc. L'unicité d'une adresse *site* est vérifiée par chaque nœud en effectuant une vérification utilisant une version modifiée des mécanismes du protocole DAD (Duplicate Address Detection) : le message de sollicitation de voisin peut être routé dans tout le réseau ad hoc, ainsi que les messages d'annonce de voisin. Cette procédure se traduit donc en pratique par une inondation du réseau suivie, en cas de duplication d'adresses, de l'inondation du réseau par un second message. Lorsqu'un nœud du réseau a créé son adresse site et qu'il décide de communiquer avec une station de l'Internet (hors du réseau ad hoc), il doit initier la découverte d'une passerelle afin de créer une adresse globale et de mettre en place une route vers l'Internet. Les deux solutions proposent d'utiliser un système proche du mécanisme de découverte des routes des protocoles de routage unicast réactifs. Lorsqu'un nœud a besoin de communiquer avec un destinataire qui est hors du réseau ad hoc, il envoie une requête spécifique en inondant le réseau, et les passerelles repondent en point à point à l'initiateur de la requête. La réponse envoyée par une passerelle contient en particulier le préfixe global à utiliser. Le nœud ayant initié la requête de découverte de passerelle peut alors créer une adresse globale dont l'unicité est déjà validée. De plus, ce nœud met en place les entrées suivantes dans sa table de routage [131] :

Destination	**Saut intermédiaire**
par défaut	par passerelle
passerelle	prochain saut (voisin) vers passerelle

[1] On précisera que l'utilisation d'adresses locales au site avec IPv6 fait actuellement l'object de discussions au sein de l'IETF. Il semble que ce type d'adresses sera finalement supprimé des spécifications de IPv6.

On notera que pour atteindre une destination de l'Internet, un nœud devra effectuer une double recherche dans sa table de routage. De plus, cette technique ne permet pas de gérer plusieurs passerelles en parallèle. Enfin, Xi et Bettstetter [132] proposent une alternative à cette méthode qui consiste en un envoi périodique par chaque passerelle de l'information sur le préfixe. Cette technique permet aux nœuds de construire leur adresse globale et de maintenir la route par défaut même si ils n'ont pas de correspondants extérieurs au réseau ad hoc. Ces informations seront donc immédiatement disponibles si nécessaire. Enfin, ces deux solutions proposent l'utilisation du protocole Mobile IPv6 afin de garantir l'accès à des nœuds du réseau ad hoc dont le réseau mère est dans l'Internet. L'adresse globale construite par un tel nœud dans le réseau ad hoc est enregistrée auprès de l'agent mère de ce nœud comme adresse temporaire Mobile IPv6. Lorsqu'un correspondant de l'Internet désirera communiquer avec l'adresse mère de cette station, les paquets seront envoyés vers le réseau mère puis encapsulés par l'agent mère vers la position courante du nœud dans le réseau ad hoc. Le mécanisme d'autoconfiguration par inondation est illustré par la figure 4.7.

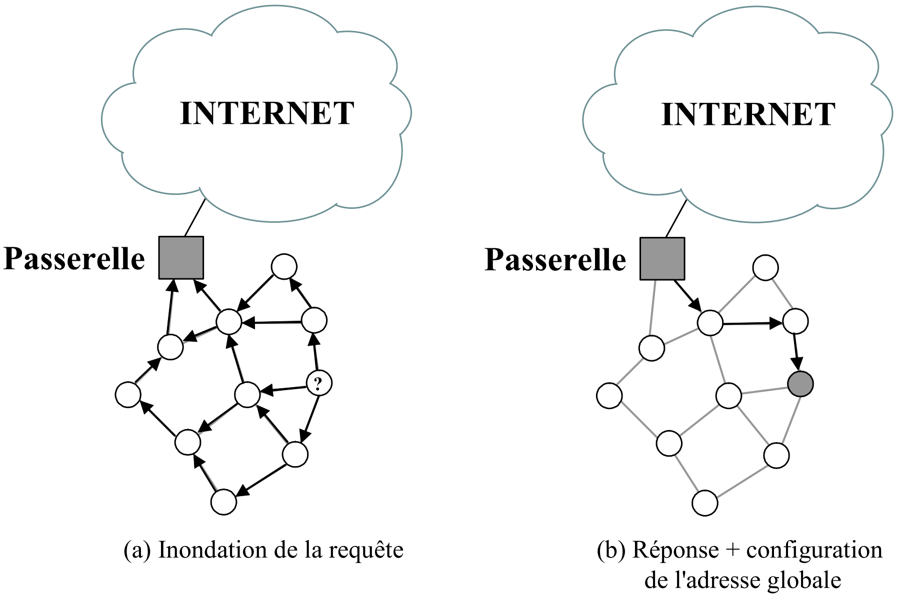

FIG. 4.7 – Autoconfiguration par inondation

Les points forts d'un mécanisme basé sur une inondation du réseau sont sa simplicité et sa robustesse. Cette technique a en effet été éprouvée lors de la mise au point des protocoles de routage unicast réactifs. Cependant, aucune de ces deux propositions ([131, 132]) ne permet de gérer plusieurs passerelles, et aucune ne propose de mécanismes concrets permettant de gérer les partitions du réseau ad hoc. En effet, si une station du réseau se retrouve isolée de sa passerelle, les paquets qui lui sont envoyés depuis l'Internet ne pourront plus l'atteindre. Le nœud doit alors réagir en choisissant une nouvelle passerelle et en créant (si besoin) une nouvelle adresse globale. Or, aucune des deux solutions présentées ne proposent de mécanisme permettant de détecter la perte de connectivité avec la passerelle : elles se contentent de préciser que l'information de passerelle doit être mise à jour périodiquement, et qu'elle a une durée de vie limitée. Aucun détail concret n'est cependant donné en ce qui concerne la procédure à suivre en cas de perte de connectivité avec la passerelle. De plus dans un réseau multi-passerelles, chaque nœud devra utiliser un en-tête de routage IPv6 afin de pouvoir spécifier par quelle passerelle doit passer un paquet destiné à une station de l'Internet. Cet en-tête permet en effet à un nœud de spécifier un point de passage intermédiaire obligatoire lors de l'envoi d'un paquet. L'utilisation d'un tel en-tête est nécessaire car un nœud ne peut pas savoir si les nœuds intermédiaires sur sa route vers la passerelle ont sélectionné la même passerelle. Or, l'ajout d'un en-tête de routage (24 octets) dans chaque paquet de données peut être évité (comme nous le montrons dans le prochain chapitre).

De plus, dans le cas où il existe dans le réseau ad hoc plusieurs passerelles vers l'Internet, aucune solution ne permet la création d'un sous-réseau contigu par passerelle : un graphe connexe dont tous les membres partagent le même préfixe. Avec ces deux solutions, il n'y a en effet aucune sorte d'organisation dans le réseau. Une fois qu'un nœud a choisi une passerelle (et donc généralement un préfixe), il maintient ce préfixe quelle que soit sa position topologique dans le réseau, et tant que la connectivité avec la passerelle est maintenue. Des nœuds utilisant des préfixes différents sont donc *mélangés* les uns avec les autres, d'où la nécessité d'utiliser un en-tête de routage pour les données. Or, si chaque sous-réseau était connexe (dans le sens où tous les nœuds partageant le même préfixe forment un graphe connexe avec la passerelle), il serait possible de faire transiter les paquets vers la passerelle sans utiliser un en-tête de routage dans chaque paquet. De plus, avec une méthode réactive maintenant la nature connexe des sous-réseaux, il est possible de réagir rapidement à une partition du réseau.

4.3.4 Conclusion

L'autoconfiguration d'adresses globales dans les réseaux ad hoc est un problème qui a été peu étudié dans le cadre du protocole IPv6. Pourtant, ce protocole offre des mécanismes permettant, grâce à l'annonce d'un préfixe, de configurer toutes les stations d'un réseau filaire. Il suffit donc a priori d'adapter ces mécanismes à la nature multi-sauts d'un réseau ad hoc. Cette approche, adoptée par [130], est cependant peu attractive car elle rajoute un degré de complexité à un protocole dont la force est la simplicité. La nature même du protocole d'autoconfiguration sans état de IPv6 est donc *déformée*. Au contraire, les solutions basées sur un mécanisme d'inondation sont mieux adaptées à la nature dynamique et imprévisible de la topologie des réseaux ad hoc. En effet, la diffusion (optimisée ou non par un mécanisme de type MPR [101]) est le meilleur moyen de joindre tous les nœuds d'un réseau ad hoc. Par contre, les propositions basées sur cette technique ([131, 132]) sont relativement statiques, dans le sens où elles ne spécifient pas concrètement la manière de sélectionner une passerelle et, plus encore, les méthodes utilisées pour réagir aux changements topologiques. Pourtant, la nature première d'un réseau ad hoc est sa dynamicité topologique : la gestion des déplacements des nœuds d'un réseau ad hoc doit donc être au centre d'une proposition d'autoconfiguration d'adresses. Enfin, il est également important de proposer un système permettant d'organiser un réseau ad hoc multi-passerelles en sous-réseaux connexes, c'est à dire en sous-réseaux dont les nœuds forment un graphe connexe avec leur passerelle.

Nous proposons donc un mécanisme d'autoconfiguration permettant la création de sous-réseaux connexes. De plus, la réactivité face aux changements topologiques est au cœur de notre proposition. Notre solution s'inspire notamment des mécanismes d'autoconfiguration d'adresses sans état de IPv6, dans le sens où la création d'une adresse se fait par l'annonce d'un préfixe et à l'aide des identifiants Ethernet uniques EUI-64. Enfin, nous utilisons une méthode de propagation des messages d'annonce de routeur originale permettant la création immédiate de sous-réseaux connexes. Nous proposons également différents algorithmes de sélection d'une passerelle qui permettent de réagir aux déplacements des nœuds du réseau ad hoc tout en maintenant la connexité des sous-réseaux. Dans le chapitre suivant, nous présentons notre solution d'autoconfiguration d'adresses et présentons les algorithmes qui permettent à un nœud de sélectionner une passerelle. Nous détaillons ces mécanismes et évaluons la complexité et la convergence de notre proposition. Enfin, nous évaluons le comportement de ces différents mécanismes et algorithmes par des techniques de simulation.

Dans un deuxième temps, nous avons appliqué le principe d'autoconfiguration d'adresses et ses conséquences au routage multicast dans les réseaux ad hoc. Pour le routage multicast, nous proposons une approche originale basée sur la nature connexe des sous-réseaux résultant de l'utilisation de notre protocole d'autoconfiguration. Nous évaluons par des méthodes de simulation l'impact de notre proposition sur les structures de diffusion du trafic multicast qui sont créées avec notre approche. Enfin, nous étudions certaines des caractéristiques topologiques de ces structures de diffusion et les comparons avec celles construites par des méthodes classiques basées sur les plus courts chemins.

Chapitre 5

Autoconfiguration d'adresses et continuité de préfixe

5.1 Introduction

Comme nous l'avons vu, l'autoconfiguration d'adresses globales dans les réseaux ad hoc a été peu étudiée dans le cadre du protocole IPv6. Cette fonctionnalité est pourtant cruciale pour un réseau ad hoc hybride dont les nœuds doivent pouvoir communiquer avec des correspondants de l'Internet. Une telle communication impose en effet à tout nœud du réseau ad hoc d'obtenir une adresse topologiquement correcte localisant ce nœud dans l'Internet (dans le réseau ad hoc hybride devenu une feuille de l'Internet). De plus, un protocole d'autoconfiguration d'adresses dans un réseau ad hoc hybride doit être capable de supporter dynamiquement plusieurs passerelles et plusieurs préfixes. Il doit également pouvoir réagir à la fusion de deux ou plusieurs réseaux hybrides (apparition de nouvelles passerelles), et au partionnement d'un réseau hybride (par exemple dans le cas où chaque nouveau réseau créé contient toujours au moins une passerelle). Dans un environnement multi-passerelles et multi-préfixes, il est également primordial de parvenir à organiser dynamiquement le réseau ad hoc en sous-réseaux logiques (au sens IP). Un tel protocole doit de plus être décentralisé, dans le sens où une entité unique ne doit pas avoir la charge de l'attribution des adresses.

De plus, les propositions existantes ne spécifient aucun mécanisme concret permettant de réagir face aux changements topologiques, et en particulier face aux partionnements du réseau ad hoc. Ces solutions nécessitent également l'utilisation d'un en-tête de routage. Cette option permet à un nœud de spécifier par quelle passerelle ses paquets émis à destination de correspondants dans l'Internet doivent être routés. Un tel mécanisme est nécessaire car les sous-réseaux logiques (au sens IP) résultant du processus d'autoconfiguration d'adresses ne sont pas contigus : les nœuds qui composent un sous-réseau ne forment en effet pas un graphe connexe avec leur passerelle. En conséquence, nous proposons un mécanisme d'autoconfiguration d'adresses IPv6 [133] basé sur le concept de *continuité de préfixe*. Notre solution est basée sur une méthode originale de propagation des messages de contrôle qui permet en particulier de créer des sous-réseaux contigus. Ce mécanisme réagit de plus rapidement aux changements topologiques quels qu'ils soient : partitionnements/fusions du réseau, apparitions/disparitions de passerelles, et déplacements des nœuds. Enfin, nous proposons et évaluons plusieurs algorithmes permettant aux nœuds du réseau de choisir leur passerelle et donc leur préfixe réseau.

5.2 Continuité de préfixe dans un réseau ad hoc

Dans les réseaux filaires, la notion de réseau au sens IP du terme induit le partage d'un support de communication par toutes les stations d'un même réseau IP. En pratique, une station peut communiquer avec une autre station du même réseau IP *localement*, c'est à dire sans routage. Topologiquement parlant, toutes les stations d'un réseau IP sont regroupées autour d'un même support physique : un sous-réseau IP est donc formé par le regroupement physique des stations le composant[1]. Ces unités logiques sont ensuite interconnectées à l'aide de routeurs pour former des *amas* de réseaux appelés systèmes autonomes. À l'échelle d'un système autonome, un réseau est représenté par une entité logique unique. Les réseaux IP sont donc organisés, dans le sens où toutes les machines d'un réseau sont regroupées. Cette organisation est illustrée par la figure 5.1. En pratique, ce modèle est à la base du routage dans l'Internet : le routage au sein d'un système autonome est effectué avec les protocoles à vecteurs de distance ou à états de lien ([95, 24]), et le routage entre systèmes autonomes est réalisé avec le protocole BGP (Border Gateway Protocol) [134, 135].

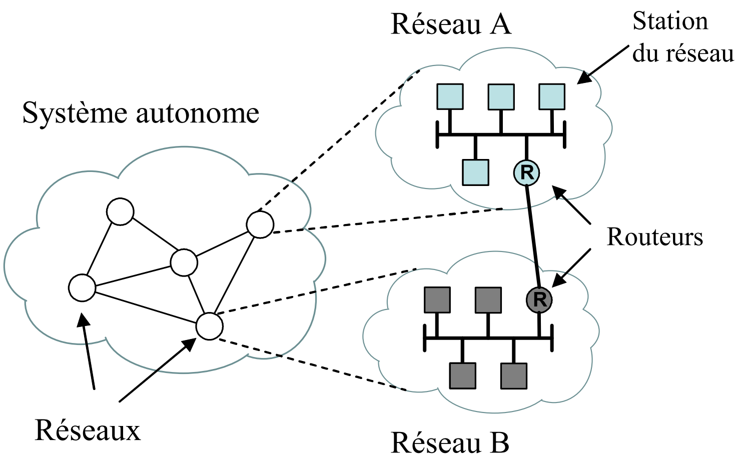

FIG. 5.1 – Organisation des réseaux dans l'Internet

Il est bien entendu nécessaire de pouvoir intégrer les réseaux ad hoc au sein d'un système autonome. Avec un réseau ad hoc bybride mono-préfixe (et dans ce cas généralement mono-passerelle), l'intégration est identique à celle d'un réseau filaire : tous les nœuds du réseau ad hoc partagent en effet le même préfixe. Dans le cas particulier d'un réseau ad hoc hybride multi-passerelles et multi-préfixes, la nature multi-sauts du support physique permet d'envisager cette intégration de deux façons. Ainsi, les solutions d'autoconfiguration d'adresses existantes ([131, 132]) ne cherchent pas à créer des sous-réseaux contigus : les nœuds des différents sous-réseaux sont *mélangés* et ils doivent utiliser un en-tête de routage pour envoyer leurs données. Au contraire, nous proposons une solution permettant de garantir la *continuité de préfixe*,

[1]Plus généralement, le support physique partagé peut être créé en utilisant les fonctionnalités évoluées des commutateurs de niveau 2 qui peuvent être dispersés géographiquement.

c'est à dire la nature connexe d'un sous-réseau. L'intégration de tels réseaux ad hoc hybrides est alors identique à celle des réseaux filaires. Ces deux approches sont illustrées par les figures 5.2 et 5.3 pour lequelles une couleur représente un préfixe réseau.

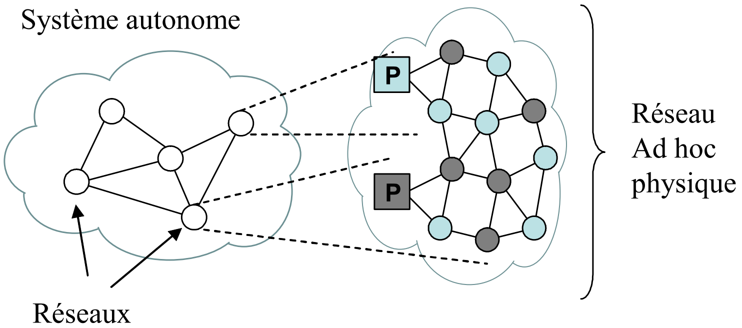

FIG. 5.2 – Intégration d'un réseau hybride SANS continuité de préfixe

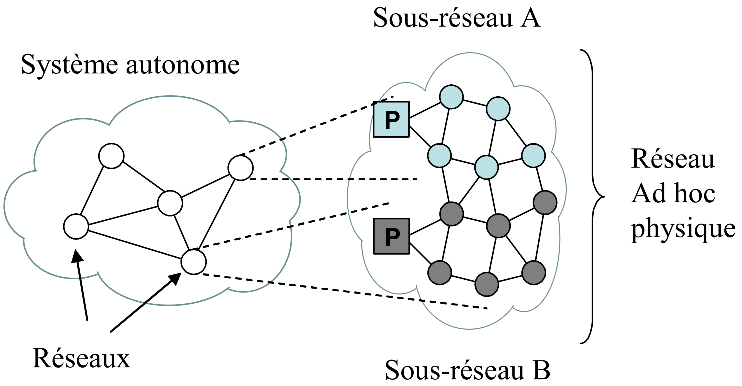

FIG. 5.3 – Intégration d'un réseau hybride AVEC continuité de préfixe

Sur ces deux figures, les passerelles (notées P) sont des nœuds du réseau ad hoc ayant une connexion (non représentée ici) vers l'Internet. On voit sur ces illustrations que le réseau ad hoc physique, c'est à dire l'ensemble multi-sauts des nœuds, est joignable par deux passerelles qui utilisent des préfixes globaux différents. Dans le cas de la figure 5.2, il n'y a pas de continuité de préfixe. Les nœuds utilisent des en-têtes de routage pour imposer à leurs paquets de passer par la passerelle dont ils partagent le préfixe. Au contraire, le réseau ad hoc de la figure 5.3 respecte la continuité de préfixe et chaque sous-réseau IP est un graphe connexe formé par les nœuds partageant un préfixe. Il est alors possible de router les paquets sans utiliser d'en-têtes de routage.

Les avantages de la continuité de préfixe sont multiples. Premièrement, cette méthode impose une organisation logique (au sens IP) du réseau ad hoc en construisant dynamiquement des sous-réseaux contigus. Une telle architecture permet d'envisager le déploiement de solutions de gestion de réseau permettant par exemple à un opérateur de superviser le fonctionnement et l'état du sous-réseau associé à l'une de ses passerelles. Avec la continuité de préfixe, les informations de management envoyées par les stations du sous-réseau ne transitent pas par des nœuds d'autres réseaux et on peut donc également assurer une certaine confidentialité des données. De plus, avec une telle organisation, les données transitent uniquement par les nœuds d'un même sous-réseau, ce qui peut permettre de mettre en place des systèmes de tarification propres à l'opérateur d'une passerelle. On notera cependant que le routage entre les sous-réseaux ad hoc peut être autorisé. Enfin, il devient possible de limiter la diffusion de certains types de traffic (ex : la diffusion d'une radio par Internet) à l'intérieur d'un sous-réseau. Nous avons donc proposé une solution d'autoconfiguration d'adresses basée sur le concept de continuité de préfixe. Afin de réaliser cet objectif, nous avons mis en œuvre une technique de propagation des annonces de préfixes qui conduit naturellement à la création de sous-réseaux contigus.

5.3 Autoconfiguration et continuité de préfixe

5.3.1 Format des messages d'annonce de préfixe

Notre proposition d'autoconfiguration d'adresses [133] repose sur une technique originale de propagation des annonces de préfixe. La continuité de préfixe est en particulier une conséquence inhérente de cette méthode de propagation. Chaque passerelle est responsable de l'envoi périodique de messages d'annonce de préfixe qui vont permettre aux nœuds du réseau ad hoc de générer une adresse IPv6 globale. Ces messages, appelés messages GW_INFO (pour GateWay INFOrmation), se propagent de proche-en-proche dans tout le sous-réseau associé à une passerelle. Leur format est illustré par la figure 5.4.

Un message GW_INFO contient en particulier l'adresse IPv6 globale de la passerelle ayant initié l'envoi de ce message. De plus, un champ *longueur du préfixe* de 8 bits permet d'extraire le préfixe contenu dans l'adresse globale de la passerelle. Ce préfixe est utilisé par tout nœud du réseau ad hoc pour construire son adresse globale en utilisant l'indentifiant Ethernet unique (EUI-64) de l'interface physique le connectant au réseau. Par défaut le champ *longueur du préfixe* doit avoir une valeur égale à 64. Dans le cas particulier où plusieurs passerelles annoncent le même préfixe (pour des questions de redondance), il est possible de différencier ces passerelles grâce à l'adresse globale contenue dans chaque message GW_INFO. Le champ *distance* est initié à zéro par chaque passerelle et est incrémenté par chaque nœud intermédiaire retransmettant le message GW_INFO. Ce champ indique donc la distance à laquelle se trouve l'émetteur du message par rapport à la passerelle. Il nous a paru plus adéquat d'intégrer un tel champ dans les messages GW_INFO plutôt que d'utiliser le champ *nombre de sauts* de l'en-tête IPv6 car il nécessite de connaître sa valeur initiale afin de pouvoir déduire la distance à laquelle on se trouve de l'émetteur initial. Chaque message GW_INFO contient également un numéro de séquence de 16 bits qui est incrémenté à chaque nouvel envoi par une passerelle. Ce champ est utilisé pour différencier les messages d'une passerelle et pour permettre de détecter la perte de messages. Enfin, chaque message GW_INFO contient un champ *voisinage* qui est utilisé par notre mécanisme de découverte des voisins. Ce mécanisme, présenté dans la section 5.3.3, permet à un nœud de valider la nature bi-directionnelle de ses liens avec ses voisins. On notera qu'un message GW_INFO peut également contenir l'adresse globale d'un

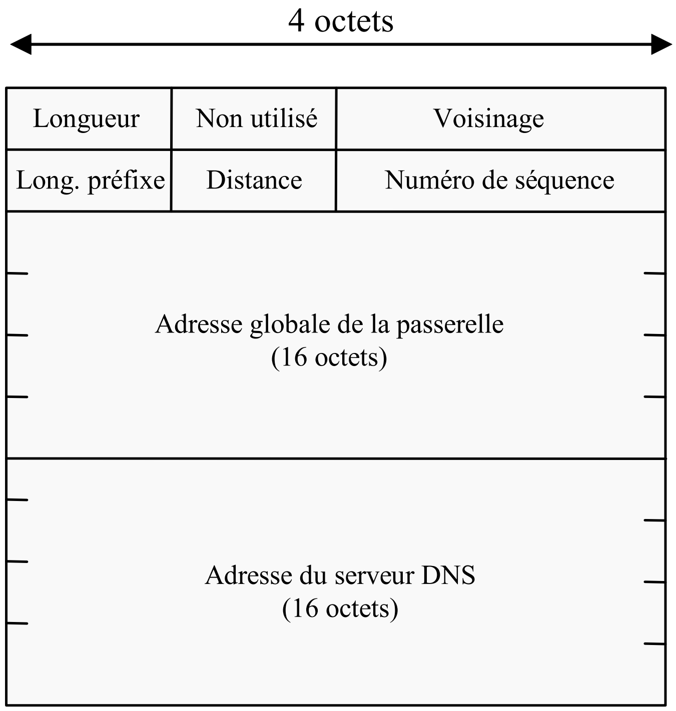

Fig. 5.4 – Format des messages GW_INFO

serveur de nom (DNS). Le champ *longueur* du message permet à un nœud de savoir si un message GW_INFO contient ou non cette information. Il indique en effet la longueur en octets du message GW_INFO.

Tout message GW_INFO est envoyé dans un paquet dont l'adresse IPv6 source est l'adresse lien-local de l'émetteur. Un tel message n'est donc pas routable, c'est à dire qu'il ne peut pas être retransmis. En particulier, le champ *nombre de sauts* de l'en-tête IPv6 du paquet contenant le message GW_INFO doit être positionné à 1 afin que le message ne soit pas routé. De plus, l'adresse destination du paquet contenant un message GW_INFO est l'adresse IPv6 de tous les routeurs du lien (ff02 :: 2). Un message GW_INFO est donc reçu par tous les voisins de l'émetteur du message (dans un réseau ad hoc tous les nœuds sont routeurs). La propagation des informations d'annonce de préfixe s'effectue alors de proche-en-proche avec une méthode visant à maintenir la continuité de préfixe.

5.3.2 Propagation des annonces de préfixe

Chaque passerelle envoie périodiquement (ex : toutes les 2 secondes) un message GW_INFO en incrémentant à chaque envoi la valeur du numéro de séquence. Ce message est reçu par tous les voisins de la passerelle et n'est initialement pas propagé. Selon la topologie du réseau et le nombre de passerelles, un nœud peut alors recevoir plusieurs messages GW_INFO. Chaque nœud sélectionne alors une passerelle (et son préfixe associé) afin de créer son adresse IPv6 globale. Nous introduisons alors la notion de *voisin-père* : un nœud A devient le voisin-père d'un nœud B si B décide d'utiliser les informations d'annonce de préfixe envoyées par A. Cette notion est au cœur de notre mécanisme de propagation. En effet, un nœud retransmet uniquement les informations d'annonce de préfixe qu'il reçoit de son voisin-père. Lorsque le nœud B reçoit un message GW_INFO de son voisin-père A, il retransmet un message GW_INFO contenant les mêmes informations que le message envoyé par A à une seule différence près : le champ *distance* d_B du message envoyé par B est incrémenté ($d_B = d_A + 1$). Enfin, les algorithmes utilisés pour la sélection du voisin-père sont présentés en détails dans la section 5.3.4. Ils permettent notamment à un nœud de sélectionner soit la passerelle la plus proche, soit le voisin-père permettant au nœud de conserver le plus longtemps possible son préfixe courant. La technique de propagation que nous proposons est illustrée par la figure 5.5.

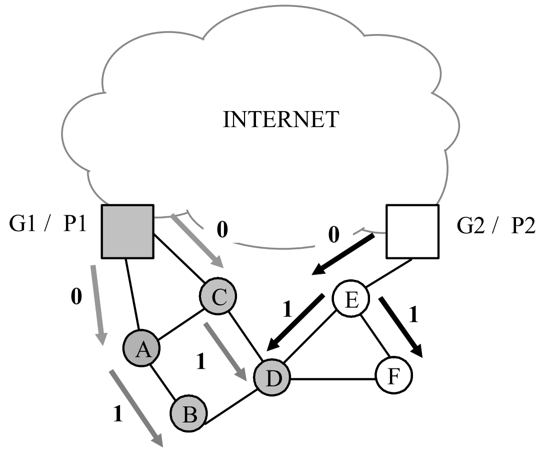

Fig. 5.5 – Propagation de proche-en-proche des messages GW_INFO

Sur la figure 5.5 il y a deux passerelles G1 et G2 annonçant respectivement les préfixes P1 et P2. Les flèches émanant d'un nœud représentent un message GW_INFO : le nombre indique la valeur du champ *distance* et la couleur indique le préfixe contenu dans le message. Pour des raisons de clarté, tous les messages ne sont pas représentés. Chaque passerelle initie l'envoi d'un message GW_INFO comportant une distance nulle. Les nœuds A et C décident alors de choisir la passerelle G1 comme voisin-père. Ils créent une adresse globale avec le préfixe P1, puis retransmettent un nouveau message GW_INFO avec une distance égale à 1. On notera par exemple que le nœud A reçoit le message envoyé par C mais décide de l'ignorer : ce message n'est donc pas retransmis. De même, le nœud D décide d'utiliser l'information contenue dans le message envoyé par C, et ignore donc celui envoyé par E. Le nœud D retransmet uniquement un message contenant les informations envoyées par C. Ce message est alors ignoré par les nœuds B, C, et E : le contenu du message envoyé par D est analysé mais la réception de ce message ne déclenche aucune action. On rappelle en effet que seule la réception d'un message GW_INFO envoyé par son voisin-père déclenche l'envoi d'un message mis à jour. De plus, la diffusion d'une annonce de préfixe envoyée par une passerelle est similaire à la propagation de données dans un arbre dont la racine serait la passerelle, et dont les membres seraient les nœuds ayant décidé d'utiliser ce préfixe.

Cette technique de propagation conduit naturellement à la création de sous-réseaux contigus. En particulier, la continuité de préfixe garantit qu'il existe, entre un nœud X et sa passerelle P, un chemin formé par des nœuds ayant également sélectionné la passerelle P et utilisant donc le même préfixe que le nœud X. En effet, si le nœud X a sélectionné le nœud Y comme voisin-père, alors Y utilise forcément le préfixe annoncé par P car un nœud ne retransmet que l'information sur son préfixe courant. Récursivement, il doit exister un chemin de nœuds utilisant le même préfixe entre le nœud X et sa passerelle P : dans le cas contraire, l'information envoyée par P n'aurait jamais atteint le nœud X. Nous utilisons cette particularité pour mettre en place, sur chaque nœud, une route par défaut dont le prochain saut est le voisin-père du nœud. Cette route est utilisée en particulier pour joindre des correspondants localisés hors du réseau ad hoc. De plus, comme tous les nœuds formant cette route utilisent le même préfixe et la même passerelle, il n'est pas nécessaire pour un nœud d'utiliser des en-têtes de routage pour garantir le transit des données via la passerelle adéquate. Le routage via la passerelle sélectionnée par un nœud est en effet garanti par la méthode utilisée pour mettre en place la route par défaut.

5.3.3 Validation des liens et table des voisins

La continuité de préfixe dans un réseau ad hoc offre donc plusieurs avantages. Elle permet en particulier d'organiser le réseau ad hoc en sous-réseaux contigus, et elle permet également d'éviter l'utilisation d'en-têtes de routage. En pratique, la nature imprévisible de la topologie d'un réseau ad hoc impose bien entendu la mise en place de mécanismes permettant de maintenir la continuité de préfixe. Un nœud doit en effet pouvoir détecter si il devient isolé de ses voisins utilisant son préfixe courant, et en particulier de son voisin-père. Le voisin-père est en effet utilisé comme prochain saut pour la route par défaut et doit donc toujours être joignable afin de permettre les communications avec des correspondants localisés hors du réseau ad hoc. Nous avons donc proposé l'utilisation d'un mécanisme original de découverte périodique du voisinage basé sur l'utilisation conjointe de messages de type HELLO et de messages unicast. Ce mécanisme est utilisé par chaque nœud pour maintenir une table des voisins, qui contient l'adresse lien-local et le préfixe de chaque voisin de ce nœud, ainsi que le numéro de séquence contenu dans les messages GW_INFO envoyés par les voisins.

Les protocoles *classiques* de découverte du voisinage utilisés dans les protocoles de routage unicast pour les réseaux ad hoc ([99, 101, 102]) souffrent en effet du problème des *zones d'ombre* du protocole IEEE 802.11 [136]. Les équipements basés sur ce protocole ayant été très majoritairement déployés dans les réseaux sans-fil à infrastructure, on peut en effet augurer de leur succès dans les réseaux ad hoc. Le problème des zones d'ombre vient du fait que le protocole IEEE 802.11 utilise des débits de transmission différents pour les paquets unicast et les paquets multicast (ou broadcast). Ces derniers sont en effet transmis à de plus faibles débits ce qui augmente en particulier leur portée de transmission. De plus, ils ne sont pas acquittés au niveau de la couche IEEE 802.11. Un paquet de données multicast est donc généralement transmis plus loin qu'un paquet unicast et n'est de plus pas acquitté au niveau de la couche MAC. Or, les protocoles classiques de découverte du voisinage utilisent uniquement des messages envoyés en broadcast pour valider les liens de voisinage. Cependant, ces liens servent généralement à l'envoi de messages unicast : un lien peut donc être validé entre deux nœuds mais être en pratique inutilisable pour l'envoi de messages unicast [136]. En conséquence, nous proposons d'utiliser un échange de messages unicast afin de valider un lien. Ces échanges sont déclenchés par l'envoi et la réception des messages GW_INFO qui sont envoyés en multicast (strictement parlant, il n'y a pas de broadcast avec IPv6). De plus, ce mécanisme utilise le champ *voisinage* des messages GW_INFO afin d'éviter de surcharger le réseau avec les messages de validation des liens.

Nous utilisons un mécanisme de triple poignée de main pour valider un lien comme étant bi-directionnel. Lorsqu'un nœud A reçoit un message GW_INFO d'un de ses voisins B, il vérifie si celui-ci figure déjà dans sa table des voisins. Si le nœud B n'apparaît pas dans la

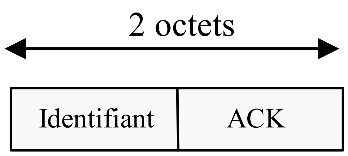

FIG. 5.6 – Format des messages d'identifiant

table des voisins de A, le nœud A crée une entrée pour B qui contient l'adresse lien-local de B ainsi que son préfixe (indiqué dans le message GW_INFO). Cette entrée est dans un premier temps marquée comme invalide. Le nœud A envoie alors un message d'identifiant de voisin en unicast à la station B. Le format de ce message est illustré par la figure 5.6. Le nœud A attribue notamment un identifiant (non encore attribué), noté A_B, à la station B. Cette attribution est uniquement valable entre les nœuds A et B. De plus, A positionne le champ ACK du message à zéro (LINK_REQ). Si la station B reçoit ce message et que le nœud A n'est pas présent dans sa table des voisins, B crée une entrée pour A : cette entrée est marquée comme étant uni-directionnelle. B envoie alors un message d'identifiant à A en lui attribuant également un identifiant (non encore attribué) noté B_A. De plus, B positionne à 1 (LINK_OK) le champ ACK du message. Si le nœud A reçoit ce message, il marque l'entrée concernant B dans sa table des voisins comme étant bi-directionnelle. Le nœud envoie alors un nouveau message d'identifiant à B en lui confirmant son numéro d'indentifiant A_B et en positionnant le champ ACK à 1. A la réception de ce message, la station B peut marquer l'entrée correspondant à A dans sa table des voisins comme étant bi-directionnelle.

Chaque entrée bi-directionnelle de la table des voisins doit de plus être rafraîchie périodiquement (ex : toutes les 6 secondes), sous peine d'être supprimée. Le rafraîchissement des entrées bi-directionnelles nécessite cependant l'envoi périodique de messages unicast. Pour éviter d'avoir à effectuer cet envoi trop souvent, on utilise le champ *voisinage* des messages GW_INFO. Ce mécanisme permet en particulier à un nœud de savoir que son voisin-père retransmet toujours des données unicast envoyées (ou retransmises) par ce nœud à destination de correspondants hors du réseau ad hoc. Ainsi si le nœud A attribue à B l'identifiant A_B, il peut indiquer dans ses messages GW_INFO si il reçoit (et retransmet) des données envoyées (ou retransmises) par B. Pour ce faire, A positionne à 1 le $A_B^{\text{ième}}$ bit du champ *voisinage* de son message GW_INFO. Lorsque le nœud B reçoit le message GW_INFO envoyé par A, il peut apprendre que A reçoit bien les données (unicast) qu'il lui envoie : B est notifié que le lien uni-directionnel de B vers A est opérationnel. Si le flux de données est bi-directionnel, cette procédure pourra être réalisée réciproquement et permettre de rafraîchir l'entrée bi-directionnelle de la table des voisins. Si il n'y a pas de flux de données entre deux voisins, la procédure utilisant les messages unicast doit être redéclenchée avant l'expiration de l'entrée. Enfin, les entrées marquées comme invalides ou uni-directionelles sont supprimées si elles ne sont pas transformées en entrées bi-directionelles avant un certain laps de temps. De plus, si l'entrée correspondant au nœud B est supprimée de la table des voisins du nœud A, ce dernier peut tenter d'informer B en lui envoyant un message d'identifiant avec le champ ACK égal à 2 (LINK_LOST). Si il y a réception de ce message, B supprime de sa table des voisins l'entrée correspondant à A.

5.3.4 Algorithmes de sélection du voisin-père

Dans notre proposition, la sélection du voisin-père est un élément crucial. En effet, cette sélection se doit d'être dynamique car il est indispensable pour tout nœud de pouvoir réagir aux changements topologiques du réseau ad hoc. La mobilité des nœuds d'un réseau ad hoc dans un environnement multi-passerelles et multi-préfixes permet également d'envisager deux approches différentes pour la sélection du voisin-père. On peut en effet privilégier la distance à la passerelle ou la stabilité du préfixe. Avec la première approche, un nœud choisit comme voisin-père la station émettant le message GW_INFO ayant la plus petite valeur du champ *distance* (par rapport aux autres émetteurs de tels messages). Cette technique permet de sélectionner le voisin-père afin d'être le plus proche possible d'une passerelle. La deuxième approche consiste au contraire à privilégier la stabilité du préfixe courant : un nœud choisit comme voisin-père une station lui permettant de continuer à utiliser son préfixe courant. Cette méthode permet à un nœud de minimiser ses changements de préfixe et les conséquences liées à des changements d'adresses.

Nous proposons ainsi trois algorithmes permettant à tout nœud du réseau ad hoc de choisir son voisin-père. Le premier algorithme privilégie la distance à la passerelle, alors que les deuxième et troisième algorithmes privilégient la stabilité du préfixe. On notera qu'un nœud doit obligatoirement choisir comme voisin-père un voisin avec lequel il partage un lien bi-directionnel. De plus, un indicateur particulier permet de préciser dans la table des voisins quelle entrée correspond au voisin-père. Enfin, les informations contenues dans le dernier message GW_INFO du voisin-père sont également conservées. Ces informations sont mises à jour par la réception des messages GW_INFO envoyés par le voisin-père. Si celui-ci arrête d'envoyer des messages pendant un certain temps (ex : par exemple 6 secondes, ce qui correspond à la perte de 3 messages GW_INFO), il n'est plus utilisé comme voisin-père : le chemin entre le nœud, son voisin-père et la passerelle n'est plus valide. L'entrée dans la table des voisins correspondant au voisin père est marquée comme invalide et est supprimée si la nature bi-directionnelle du lien ne peut pas être établie. De plus, le prochain saut de la route par défaut est modifié : le nœud sélectionne le voisin (de même préfixe) annonçant la plus faible distance à la passerelle. Si il n'existe aucun voisin utilisant le même préfixe, le nœud attend de recevoir un nouveau message GW_INFO. Enfin lorsque le voisin-père annonce un nouveau préfixe (après sélection ou non d'un nouveau voisin-père), une nouvelle adresse globale est créée avec l'identifiant Ethernet unique (EUI-64) de l'interface par laquelle le message GW_INFO a été reçu.

5.3.4.1 Sélection de la passerelle la plus proche

Avec ce premier algorithme appelé simplement *algorithme-distance*, un nœud choisit comme voisin-père la station émettant les messages GW_INFO avec la plus petite valeur du champ *distance*. Cet algorithme est illustré par la figure 5.7. Quand un nœud reçoit un message GW_INFO, il vérifie tout d'abord si il a un lien bi-directionnel avec l'émetteur. Si le lien n'est pas bi-directionnel, l'algorithme de sélection du voisin-père ignore le message GW_INFO. De même si l'émetteur du message GW_INFO est le voisin-père du nœud, l'entrée de la table des voisins est simplement mise à jour. Enfin, si le lien avec l'émetteur du message est bi-directionnel et que celui-ci n'est pas le voisin-père, le nœud compare la distance annoncée dans le message GW_INFO avec la distance annoncée par son voisin-père. Si la distance annoncée dans le message GW_INFO est inférieure à celle annoncée par le voisin-père, le nœud compare le numéro de séquence du message avec ceux annoncés par ses voisins utilisant le même préfixe que l'émetteur. Si il n'existe pas de voisins utilisant le même préfixe, l'émetteur du message est immédiatement choisi comme nouveau voisin-père (c'est à dire au bénéfice du doute). Si il existe des voisins utilisant le même préfixe (et la même passerelle), et que le numéro de séquence du message est supérieur ou égal à toutes les entrées correspondant aux voisins de même préfixe, l'émetteur est sélectionné comme nouveau voisin-père. L'utilisation des numéros de séquence permet en effet à une station d'éviter de choisir un nœud comme voisin-père si celui-ci envoie des informations obsolètes (par exemple si une station ne fonctionne pas correctement ou si il y a congestion sur un lien).

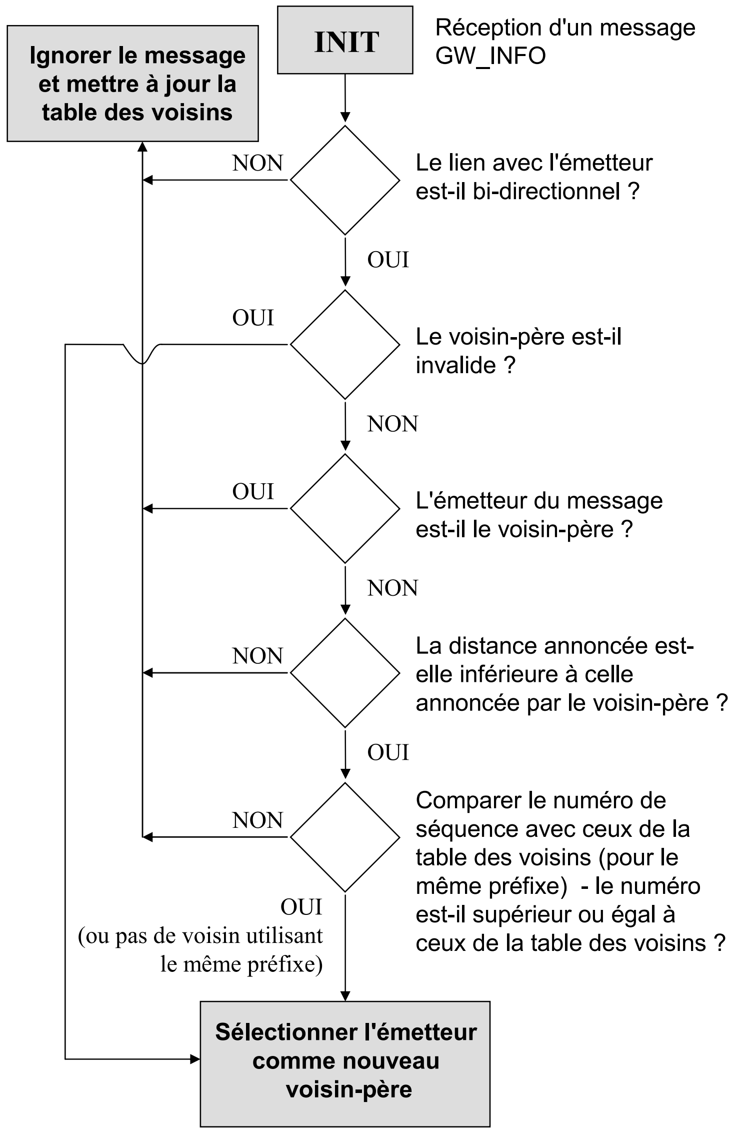

FIG. 5.7 – Algorithme-distance

Cet algorithme est très simple et a l'avantage de permettre à chaque nœud de sélectionner la passerelle la plus proche. Dans un environnement multi-préfixes, cet algorithme ainsi que la méthode de gestion du voisinage permettent aux nœuds de réagir à l'apparition (fusion de réseaux) et à la disparition (partitionnement du réseau) de passerelles. Par exemple si un groupe de nœuds ne reçoit plus de messages GW_INFO d'une passerelle pendant un certain temps, ils vont graduellement décider de changer de voisin-père (à condition qu'une alternative existe). Si il existe un chemin entre ces nœuds et une ou plusieurs autres passerelles, tous les nœuds vont changer de passerelle et de préfixe. Enfin dans le cas où les nœuds d'un réseau ad hoc sont uniformément répartis dans l'aire géographique formée par les passerelles, et si l'on considère que les conditions de transmission radio sont identiques pour tous les nœuds, cet algorithme conduira à une répartition identique (statistiquement parlant) des nœuds dans chaque sous-réseau. En effet, chaque nœud choisit la passerelle la plus proche en terme de sauts, mais l'uniformité des conditions de transmission radio et des localisations géographiques conduit à rendre équivalent distance géographique et distance en terme de sauts ([125]). On considère également que les déplacements des nœuds *ne brisent* pas la distribution géographique uniforme des nœuds dans l'aire formée par les passerelles [137]. En conséquence, les sous-réseaux ainsi formés auront des tailles identiques (statistiquement parlant) ce qui conduit à un partage de charge au niveau des passerelles.

5.3.4.2 Stabilité du préfixe

L'inconvénient de l'algorithme-distance est qu'il peut générer un grand nombre de changements de préfixe. Au cours de ses déplacements, un nœud va en effet toujours chercher à utiliser la passerelle la plus proche quel que soit le préfixe utilisé. Nous proposons donc deux alternatives dont l'objectif est de permettre à un nœud de maintenir son préfixe courant tant qu'il existe un chemin vers sa passerelle courante. Ainsi, un nœud restreint le choix de son voisin-père à ses voisins utilisant son préfixe courant, quelle que soit la distance à laquelle ils se trouvent de la passerelle. Enfin, dans le cas où un nœud n'a plus de voisin-père et si aucun de ses voisins n'utilisent son préfixe courant, il est autorisé à sélectionner une station annonçant un préfixe différent comme nouveau voisin-père. Les deux algorithmes que nous proposons afin de maximiser le temps de maintien du préfixe diffèrent par la procédure utilisée par un nœud pour changer de préfixe. Nous présentons dans un premier temps la partie commune aux deux algorithmes, et décrivons ensuite la procédure de sélection d'une nouvelle passerelle.

Lorsqu'un nœud ayant déjà sélectionné un voisin-père reçoit un message GW_INFO, il vérifie dans un premier temps si il a un lien bi-directionnel avec l'émetteur et si celui-ci annonce le même préfixe qu'il utilise. Si l'un de ces tests échoue, le message GW_INFO est ignoré (l'entrée de la table des voisins est cependant mise à jour). Si le lien est bi-directionnel et si l'émetteur du message GW_INFO utilise le même préfixe, une procédure identique à celle utilisée par l'algorithme-distance permet de sélectionner ou non l'émetteur comme nouveau voisin-père. Le nœud sélectionnera ainsi comme voisin-père le voisin utilisant le même préfixe et étant le plus proche de la passerelle courante. Ainsi, le chemin entre un nœud et sa passerelle sera un plus court chemin au sein du sous-réseau. Au niveau du réseau ad hoc dans sa globalité, ce chemin peut cependant ne pas être un plus court chemin topologique. La partie commune des deux algorithmes maximisant la stabilité du préfixe est présentée par la figure 5.8.

Enfin, lorsqu'un nœud reçoit un message GW_INFO et si il n'a pas de voisin-père (entrée

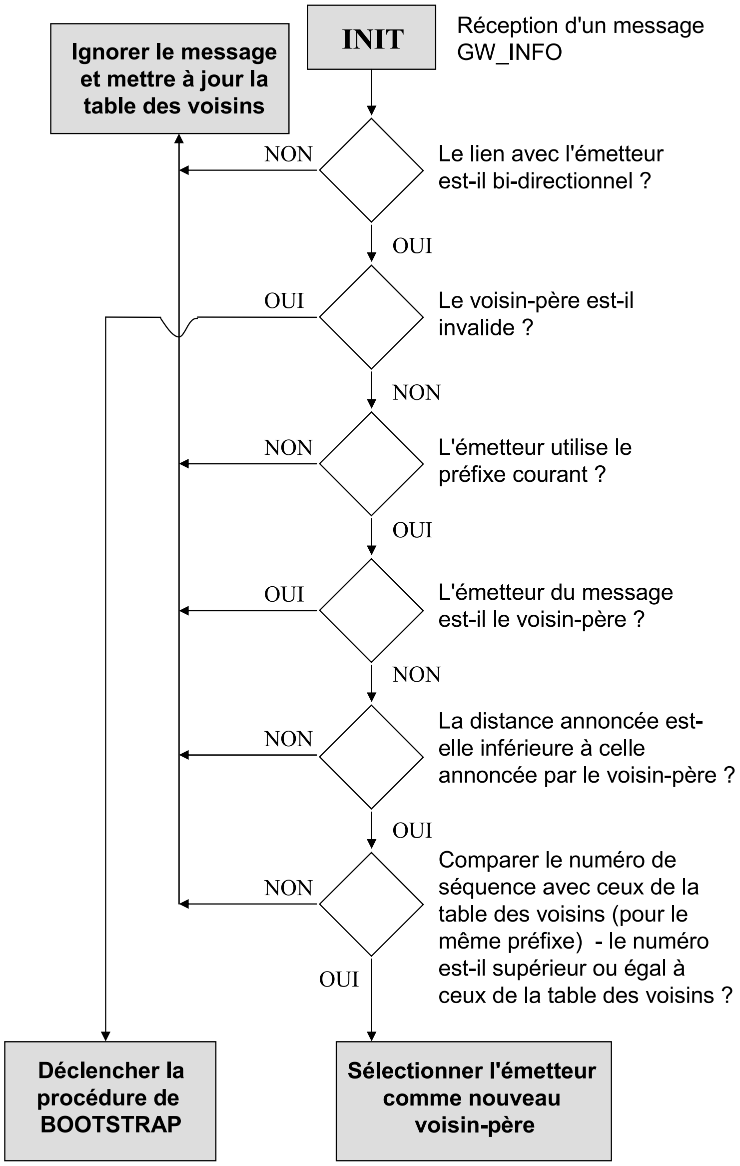

FIG. 5.8 – Partie commune des algorithmes de stabilité

invalide ou expirée), il utilise une fonction de *bootstrap* (démarrage) afin de choisir un nouveau voisin-père. Nous proposons deux variantes de la procédure de bootstrap.

Avec le premier algorithme appelé *algorithme de stabilité sans attente*, la procédure de bootstrap est assez simple. Le nœud vérifie dans un premier temps si l'émetteur du message annonce le préfixe courant. Si c'est le cas, ce voisin est choisi comme voisin-père. Sinon, le nœud vérifie dans sa table des voisins si il a des voisins (avec lesquels le lien est bi-directionnel) utilisant le préfixe courant. Si c'est le cas, le nœud ne choisit pas l'émetteur comme nouveau voisin-père et attend de recevoir un message GW_INFO d'un voisin utilisant le même préfixe que lui. Pendant ce laps de temps, le prochain saut vers la route par défaut est modifié : le nœud sélectionne le voisin (de même préfixe) annonçant la plus faible distance à la passerelle. L'adresse globale peut donc être conservée. Enfin, si le nœud n'a aucun voisin utilisant son préfixe courant, l'émetteur du message GW_INFO est choisi comme voisin-père. L'avantage de cette méthode est que le prochain saut de la route par défaut est immédiatement sélectionné.

Cependant, il peut s'avérer plus intéressant d'utiliser une méthode plus complexe. En effet, avec les algorithmes de stabilité, un nœud peut conserver son préfixe courant aussi longtemps qu'il a des voisins utilisant le même préfixe. On peut donc imaginer une procédure de bootstrap sélectionnant un voisin-père afin de maximiser le nombre de voisins utilisant le même préfixe que le nouveau voisin-père. Etant donné que les déplacements des nœuds sont imprévisibles, on peut espérer qu'un tel choix permettra d'augmenter les chances d'un nœud de conserver des voisins de même préfixe. Le second algorithme de stabilité, appelé *algorithme de stabilité lent*, utilise donc une version plus évoluée de la procédure de démarrage. Cette procédure est identique à la précédente, sauf dans le cas où le nœud n'a aucun voisin utilisant son préfixe courant lorsque son voisin-père est invalide. Au lieu de choisir immédiatement l'émetteur du message GW_INFO comme nouveau voisin-père, le nœud va sélectionner un préfixe parmi les préfixes utilisés par ses voisins, et va attendre que l'un de ses voisins utilisant le préfixe sélectionné lui envoie un message GW_INFO (la méthode utilisée pour sélectionner le nouveau préfixe est présentée dans le prochain paragraphe). Le nœud attend un tel message pendant un certain lap de temps, et passe à la procédure sans attente si aucun message n'est reçu pendant ce laps de temps. L'algorithme de stabilité lent réagit donc moins vite à la perte de connectivité avec le sous-réseau courant car le nœud peut être ammené à devoir attendre l'envoi de nouveaux messages GW_INFO.

La sélection du préfixe lors de la procédure de bootstrap de l'algorithme de stabilité lent est effectuée de la manière suivante. Le nœud choisit dans un premier temps le préfixe pour lequel il a le plus de voisins. Dans le cas où plusieurs préfixes sont utilisés par un même nombre de voisins, le nœud choisit, parmi les préfixes utilisés par le plus grand nombre de voisins, le préfixe (ou groupe de nœuds) pour lequel un voisin annonce la distance à la passerelle la plus faible. Enfin, si un choix n'est toujours pas possible, la moyenne des distances à la passerelle d'un groupe est utilisée pour permettre la sélection du préfixe. Avec cette procédure de sélection, nous espérons pouvoir permettre à un nœud d'augmenter le temps pendant lequel il pourra maintenir son nouveau préfixe.

5.3.5 Implémentation et convergence

Nous avons implémenté notre protocole d'autoconfiguration sur les plate-formes Linux et FreeBSD sous la forme d'un démon autonome utilisé en parallèle du protocole de routage unicast. De plus, dans le cadre du projet RNRT SAFARI, nous avons intégré notre proposition d'autoconfiguration au sein du protocole OLSR. En pratique, nous avons utilisé des valeurs pour la fréquence d'envoi des messages similaires à celles utilisées dans les protocoles de routage pour les réseaux ad hoc. De même, les valeurs des temporisateurs (utilisés par exemple pour considérer qu'un lien est rompu) sont du même ordre de grandeur que celles utilisées pour le routage. La convergence du protocole est en effet dépendante des valeurs utilisées pour la fréquence d'envoi des messages et pour les temporisateurs.

Comme pour OLSR et TBRPF, nous avons dans un premier temps proposé d'utiliser une fréquence d'envoi des messages GW_INFO égale à 2 secondes. Cependant, étant donné la faible taille de ces messages (122 octets avec les en-têtes IPv6, Ethernet et 802.11), nous avons décidé d'augmenter cette fréquence à un paquet toutes les secondes. En pratique, chaque nœud retransmet en effet uniquement un message GW_INFO (à partir des informations envoyées par le voisin-père) toutes les secondes. La surcharge liée à cet envoi (976 b/s par lien radio) est donc faible si on la compare aux débits des technologies sans-fil actuelles (11 Mb/s pour IEEE 802.11b et 54 Mb/s pour IEEE 802.11a). Nous notons cette période P_{GW}. En ce qui concerne la valeur du temporisateur utilisé pour les entrées de la table des voisins, nous proposons de supprimer une entrée si elle n'a pas été re-validée au bout de 3,5 secondes. Nous admettons donc, dans le pire des cas, la perte de 3 messages GW_INFO. Nous notons ce temporisateur t_l (et $t_l = 3,5 * P_{GW}$). De plus, le temps nécessaire à la validation d'un lien (procédure de triple poignée de main) est noté ϵ : ce temps dépend uniquement des délais nécessaires à l'envoi des messages sur un lien et du temps de traitement des messages. Nous admettons que ϵ est identique pour toutes les paires de nœuds du réseau ad hoc. Enfin nous notons α le temps nécessaire à un nœud pour traiter puis envoyer un nouveau message GW_INFO. Nous admettons également que α est identique pour tous les nœuds.

Au démarrage du protocole d'autoconfiguration, un nœud distant de p sauts d'une passerelle recevra donc un message GW_INFO en $t_s = p(\epsilon + \alpha)$ secondes. Si l'on considère ϵ et α de l'ordre de quelques millisecondes, la convergence initiale est extrêmement rapide et dépend de la distance entre un nœud et la passerelle la plus proche de ce nœud. En ce qui concerne la réactivité du protocole, on considère le pire cas de figure suivant. Un nœud N reçoit un message GW_INFO de son voisin-père, et celui-ci se déplace immédiatement après hors de portée radio du nœud N. Il faudra donc t_l secondes au nœud N pour détecter la perte du lien. Cependant, juste avant de considérer le lien comme rompu, N reçoit un message GW_INFO d'un voisin V (quel que soit l'algorithme utilisé) dont l'annonce est moins intéressante que celle envoyée par le voisin-père. Comme le lien vers le voisin-père est encore valide, ce message est ignoré. Lorsque le prochain message GW_INFO envoyé par V atteint N, un temps de $t_l + P_{GW} + \alpha$ s'est écoulé depuis le départ du voisin-père. La connectivité vers l'Internet est alors rétablie au bout d'environ 5 secondes dans le pire des cas. On notera que si par exemple l'information envoyée par V avait été plus intéressante que celle annoncée par le voisin-père, la réactivité aurait dans le meilleur des cas été immédiate (le message GW_INFO de V arrive juste après celui envoyé par le voisin-père). De plus, lors du déplacement par exemple d'une *grappe* de nœuds, la convergence dépend généralement du nœud situé le plus en amont, c'est à dire celui le plus proche d'une passerelle après le déplacement. En effet, lorsque ce nœud décide de choisir un nouveau voisin-père, cela déclenche l'envoi de messages GW_INFO en aval de la grappe de nœuds. Dans le pire des cas, le nœud en amont attendra un peu moins de $t_l + P_{GW} + \alpha$ secondes pour retransmettre son premier message GW_INFO. Or pendant ce même laps de temps, tous les nœuds de la grappe auront également supprimé l'entrée de la table des voisins correspondant à leur voisin-père respectif : ils pourront donc immédiatement accepter le nouveau message GW_INFO. Le temps de convergence pour un nœud à la profondeur p dans la grappe est donc de $t_l + P_{GW} + \alpha + p(\epsilon + \alpha)$, soit environ 5 secondes (si l'on considère α et ϵ de l'ordre de quelques millisecondes. Enfin, on notera qu'il est possible de réduire le temps de convergence en augmentant la fréquence d'envoi des messages GW_INFO. En pratique, il faut trouver une valeur de compromis entre vitesse de convergence et surcharge liée à l'envoi des messages GW_INFO.

5.3.6 Détection d'adresses dupliquées

Comme nous l'avons vu précédemment, les spécifications d'autoconfiguration sans état du protocole IPv6 permettent de tester l'unicité d'une adresse IPv6 sur un lien. Dans un réseau ad hoc, cette procédure ne peut pas être utilisée en l'état et nécessite certaines modifications

([130, 131]). Or, l'utilisation d'identifiants Ethernet uniques (EUI-64) pour la création des adresses garantit, dans des conditions d'utilisation normale, l'unicité des adresses ainsi créées. Le protocole LIN6 ([46]) assume ainsi l'unicité des identificateurs LIN6 (les LIN6-ID) car la probabilité d'une collision d'adresses est quasi nulle lorsque les adresses sont créées automatiquement à partir des EUI-64. Nous proposons donc de ne pas utiliser la procédure de détection d'adresses dupliquées dans un réseau ad hoc car elle induit une surcharge significative pour un gain quasi nul. On notera qu'il est cependant possible de rajouter cette procédure à notre proposition en utilisant par exemple une méthode basée sur la diffusion dans tout le réseau des messages de sollicitation de voisin ([131]).

5.4 Intégration avec Mobile IPv6

Nous proposons de plus l'utilisation du protocole Mobile IPv6 afin de permettre aux nœuds du réseau ad hoc d'être joignables quel que soit leur préfixe courant. En effet, si l'adresse créée dans le réseau ad hoc est utilisée comme adresse temporaire du protocole Mobile IPv6, un nœud pourra être joignable via son agent mère. Après avoir créé son adresse temporaire dans le réseau ad hoc, un nœud enregistre cette adresse auprès de son agent mère. Les paquets de données envoyés par des correspondants à destination de l'adresse mère du nœud mobile seront donc encapsulés par l'agent mère à destination de l'adresse temporaire du mobile dans le réseau ad hoc. Les algorithmes de stabilité du préfixe sont donc particulièrement adaptés à cette utilisation car ils permettent de minimiser les changements de préfixe et donc les enregistrements d'adresses temporaires auprès des agents mère. Un changement d'adresse dans le réseau ad hoc induit en effet l'envoi d'un message d'enregistrement de l'association entre un mobile et son agent mère. On cherchera donc à minimiser cette procédure car elle peut induire une perte des données lors du changement d'adresse. Les algorithmes de stabilité du préfixe sont donc tout naturellement adaptés à l'utilisation de Mobile IPv6.

5.5 Conclusion

Nous avons présenté dans ce chapitre notre proposition visant à permettre l'autoconfiguration des adresses globales des nœuds d'un réseau ad hoc. Nous avons proposé une méthode originale de propagation des messages d'annonce de préfixe qui conduit naturellement à la création de sous-réseaux IPv6 contigus. Cette continuité de préfixe est de plus un concept original dans le cadre des réseaux ad hoc. Nous avons également décrit comment un nœud du réseau sélectionne son voisin-père, et donc son préfixe et sa passerelle. Nous proposons ainsi de privilégier soit la distance à la passerelle, soit la stabilité du préfixe. Nous avons de plus proposé l'utilisation d'une méthode de découverte des voisins basée sur l'utilisation conjointe de messages unicast et multicast. Cette technique permet d'éviter le problème des zones d'ombre ([136]) introduit par le protocole de communication IEEE 802.11. Elle permet également d'utiliser les messages GW_INFO afin d'effectuer la validation d'un lien entre deux nœuds voisins grâce à l'utilisation d'un mécanisme d'attribution d'identifiants.

Notre proposition s'intègre de plus parfaitement avec le protocole Mobile IPv6. Une adresse créée automatiquement dans le réseau ad hoc peut en effet être utilisée comme adresse temporaire enregistrée auprès d'un agent mère. Cette interaction permet à un nœud mobile du réseau ad hoc d'être joignable quelle que soit sa position dans le réseau ad hoc, et quel que soit le préfixe global utilisé pour la création de son adresse.

Dans le chapitre suivant, nous évaluons les algorithmes de sélection du voisin-père en utilisant divers indicateurs. Nous étudions également l'impact de ces algorithmes sur la création des sous-réseaux IPv6. En particulier, nous analysons les caractéristiques topologiques des sous-réseaux ainsi créés. Enfin, nous évaluons la vitesse de convergence des différents algorithmes en introduisant la notion de nœud orphelin.

Chapitre 6

Evaluation du protocole d'autoconfiguration d'adresses

6.1 Introduction

Le protocole d'autoconfiguration d'adresses que nous avons proposé repose principalement sur les algorithmes permettant la sélection du voisin-père. En effet, la continuité de préfixe est assurée par la méthode de propagation des annonces de préfixe. Or la sélection du voisin-père est au cœur de cette méthode, puisque seules les informations envoyées par le voisin-père sont propagées par un nœud. De plus, les algorithmes de sélection du voisin-père ont une forte influence sur les nœuds et sur l'organisation du réseau ad hoc. Ils sont en effet à la base de la création des sous-réseaux contigus et conditionnent donc l'organisation du réseau ad hoc en sous-réseaux IP.

Afin de valider le fonctionnement des algorithmes et afin de pouvoir les évaluer et les comparer, nous avons procédé à un grand nombre de simulations à l'aide du simulateur NS. Nous avons utilisé différents indicateurs afin de comparer les algorithmes de sélection du voisin-père. De plus, nous avons étudié certaines des caractéristiques topologiques des sous-réseaux créés par le processus d'autoconfiguration. Il est bien entendu difficile de simuler des réseaux ad hoc car il n'existe pas de modèle type permettant de simuler la topologie du réseau et les déplacements des nœuds. Nous utilisons donc les modèles de mobilité et les topologies utilisés dans d'autres études ([110, 111, 112, 113, 116]) visant à évaluer les performances de protocoles pour les réseaux ad hoc.

6.2 Environnement de simulation

Nous utilisons l'implémentation du protocole IEEE 802.11 du simulateur NS pour simuler les communications radio entre les nœuds du réseau ad hoc. La portée radio des nœuds est fixée à 250m, et la méthode de propagation des ondes est basée sur un modèle d'atténuation du signal proportionnelle à la distance r par rapport à la source des données. Cette atténuation est proportionnelle à $1/r^2$ pour les distances proches, et à $1/r^4$ pour les plus grandes distances (pour une portée de 250m, la distance de transition entre ces deux méthodes est calculée comme étant égale à 86m).

Nous simulons les déplacements des nœuds avec une version modifiée du modèle de déplacement aléatoire classique (random waypoint ou RWP) utilisé dans la plupart des simulations de réseaux ad hoc. Plusieurs études [137, 138] ont en effet montré que le modèle RWP standard introduit une fluctuation des conditions de simulation au cours du temps. En particulier, la vitesse moyenne des déplacements des nœuds diminue au cours du temps, ce qui ne permet pas de tirer des conclusions valides sur l'influence de la vitesse de déplacement sur les résultats. De plus, la répartition géographique des nœuds dans l'espace converge vers une distribution non-uniforme des nœuds car l'aire géographique simulée est nécessairement bornée [137]. Or avec le modèle RWP, la distribution initiale des nœuds dans l'espace est uniforme. Nous utilisons donc une version modifiée du modèle RWP [139]. Ce modèle stabilise la vitesse moyenne des déplacements des nœuds au cours du temps et positionne initialement les nœuds dans l'aire de simulation selon la distribution géographique vers laquelle le modèle RWP converge.

Nous avons considéré une aire de simulation de 2000x2000 m^2 avec 100 nœuds mobiles (voir figure 6.1). Nous avons placé une passerelle aux quatre coins de cette aire, à 250 mètres de chacun des côtés. Ces quatre passerelles annoncent chacune un préfixe réseau différent. De plus, nous utilisons des nœuds particuliers que nous nommons nœuds relais, afin de garantir une couverture radio minimale des passerelles. En effet, nous avons constaté que les déplacements aléatoires des nœuds peuvent conduire à l'isolation d'une passerelle (du point de la couverture radio). La distribution géographique vers laquelle converge le modèle RWP ne couvre en effet pas les coins d'une surface carrée ou rectangulaire [137]. Or lorsqu'une passerelle est isolée, un grand nombre de nœuds peuvent être isolés de toute passerelle ce qui conduit à l'impossibilité d'évaluer notre proposition d'autoconfiguration d'adresses. Pour éviter l'isolement radio des passerelles, nous avons donc placé des nœuds relais fixes dans le voisinage des passerelles. Ils permettent de propager les annonces de préfixe dans un rayon assez large autour d'une passerelle afin de ne pas avoir de nœuds isolés. Nous utilisons 3 nœuds relais pour chacune des quatre passerelles : ils forment avec chaque passerelle un carré de 200 mètres de côté "orienté" vers le centre de l'aire de simulation. Bien entendu les nœuds relais sont ignorés lors de l'évaluation des performances des algorithmes de sélection du voisin-père.

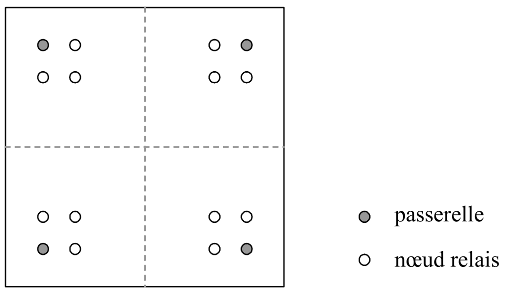

FIG. 6.1 – Schéma de l'architecture simulée

Enfin, le modèle de déplacement RWP permet de configurer la vitesse moyenne de déplacement des nœuds ainsi que le temps de pause moyen pendant lequel une station reste fixe. Nous avons utilisé diverses valeurs moyennes pour la vitesse de déplacement et le temps de pause afin d'évaluer l'impact de ces facteurs sur notre proposition. Pour une simulation donnée, tous les nœuds utilisent les mêmes valeurs statistiques. Au cours des diverses simulations, nous avons fait varier la vitesse moyenne V_m de déplacement des nœuds entre 1 et 5 m/s (en utilisant uniquement les valeurs entières). La vitesse instantanée V_i d'un nœud est choisie aléatoirement et uniformément telle que $V_i \in [V_m + 0,5; V_m - 0,5]$. De plus, le temps de pause P est fixe au cours d'une simulation et égal à $P = i * 30$ avec $i \in [1, 5]$. Nous avons donc obtenu 25 combinaisons différentes en associant vitesse de déplacement et temps de pause. Au cours de ce chapitre, nous utilisons le terme *combinaison* pour indiquer une paire (V_m ;P). Nous avons ainsi généré 10 simulations (ou scénarios de déplacements) pour chacune des 25 combinaisons. En fait, les résultats présentés dans ce chapitre ont convergé très rapidement mais nous avons néanmoins continué à utiliser 10 simulations pour chaque combinaison. Les résultats convergent soit vers une valeur particulière, soit vers une distribution mathématique. Enfin, chaque simulation dure 900 secondes (15 minutes) de temps simulé. Le volume de déplacements générés par le modèle RWP ne permet en effet pas d'obtenir des résultats différents si l'on augmente la durée d'une simulation au delà de 15 minutes. Enfin, les résultats présentés sont des statistiques obtenues à partir des 10 simulations effectuées pour chaque combinaison. Chaque simulation a de plus été utilisée pour évaluer chacun des trois algorithmes. On précisera également que pour chaque simulation, tous les nœuds du réseau ad hoc utilisent le même algorithme.

6.3 Mesures et indicateurs

Afin d'évaluer notre proposition et afin de comparer les différents algorithmes de sélection du voisin-père, nous avons utilisé plusieurs indicateurs. Nous avons ainsi étudié le temps de maintien d'un préfixe, c'est à dire le temps pendant lequel un nœud conserve son préfixe courant. Cet indicateur nous permet en particulier d'évaluer le gain obtenu avec les algorithmes de stabilité. Conjointement, nous avons étudié le nombre de fois où un nœud change de voisin-père. Nous avons de plus distingué les changements de voisin-père qui n'occasionnent pas de changement de préfixe. Nous avons ensuite analysé certaines caractéristiques topologiques des sous-réseaux IPv6 générés par la procédure d'autoconfiguration d'adresses. Nous avons enfin introduit la notion de nœuds orphelins afin d'étudier les cas où un nœud est isolé par la faute d'un algorithme de sélection du voisin-père.

6.3.1 Temps de maintien d'un préfixe

Le temps moyen T_m pendant lequel un nœud conserve son préfixe courant est le premier indicateur que nous avons décidé de mesurer. La figure 6.2 présente les résultats obtenus au cours des simulations pour les 25 combinaisons (V_m ;P). L'algorithme-distance est noté *A.Distance*, l'algorithme de stabilité sans attente est noté *A.Stab-S.Attente*, et l'algorithme de stabilité lent est noté *A.Stab-Lent*. Au niveau général, on voit sur cette figure que la vitesse moyenne V_m de déplacement a une influence assez forte sur le temps T_m de maintien d'un préfixe, et ce quel que soit l'algorithme utilisé. Au contraire, l'influence du temps de pause est plus faible. Ces conclusions sont cohérentes avec une étude dont l'objectif était d'évaluer l'influence des paramètres des simulations de protocoles dans les réseaux ad hoc [112].

En ce qui concerne la comparaison des algorithmes, on voit clairement que les algorithmes de stabilité permettent aux nœuds de conserver en moyenne leur préfixe courant entre 6 et

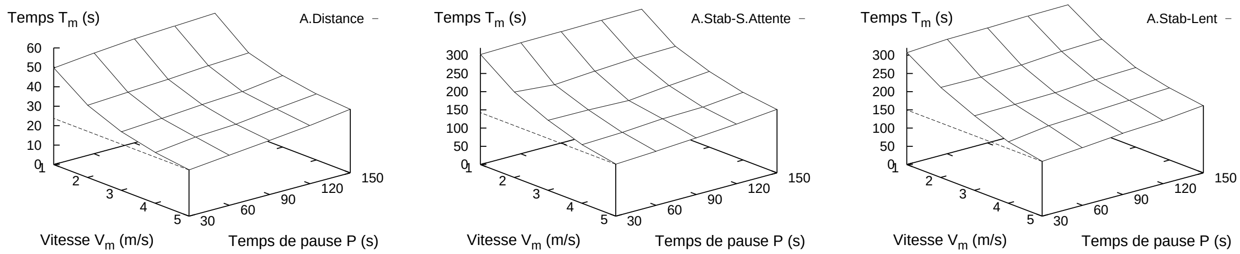

FIG. 6.2 – Temps moyen de maintien d'un préfixe

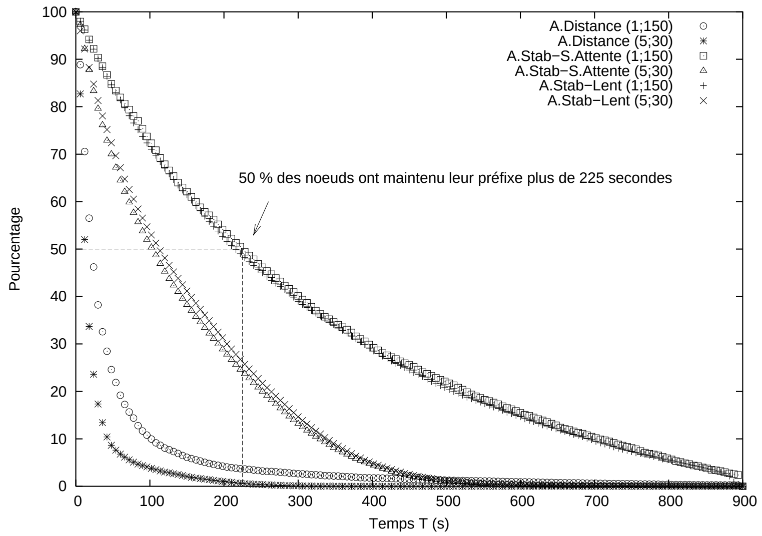

FIG. 6.3 – Temps de maintien d'un préfixe

8 fois plus longtemps qu'avec l'agorithme-distance. Ces algorithmes sont donc très efficaces et leur fonctionnement est validé. On remarque également que ces deux algorithmes ont des performances très proches. Afin d'avoir une vision plus précise du temps de maintien d'un préfixe, nous présentons également ces résultats sous la forme d'une fonction cumulative inverse. La figure 6.3 montre en effet le pourcentage de nœuds ayant été capables, au cours des simulations, de maintenir leur préfixe courant pendant une durée supérieure au temps T. Nous présentons ces résultats pour les trois algorithmes et pour les deux combinaisons (V_m ;P) *extrêmes* suivantes : (1 ;150) et (5,30). La combinaison (1 ;150) correspond en effet à une vitesse de déplacement lente et à un temps de pause élevé : la mobilité des nœuds est donc faible. Au contraire, la combinaison (5,30) correspond à une vitesse de déplacement plus rapide et à un temps de pause plus faible : la mobilité des nœuds est donc plus élevée que dans le cas précédent. On remarque en particulier que les deux algorithmes de stabilité ont des performances quasi identiques même lorsque l'on ne considère pas uniquement le temps moyen comme dans la figure précédente (6.2). On note également que lorsque la mobilité est faible et avec les algorithmes de stabilité, un pourcentage non nul (environ 2%) de nœuds parvient à conserver son préfixe pendant toute la durée de la simulation (900 secondes). De plus dans les mêmes conditions, 10% des nœuds parviennent à conserver leur préfixe pendant plus de 700 secondes. Avec une mobilité plus élevée, les temps de maintien du préfixe sont environ divisés par deux.

Enfin avec l'algorithme-distance, le temps de maintien d'un préfixe est fortement réduit.

6.3.2 Changements de voisin-père

Le temps de maintien d'un préfixe présenté précédemment est bien entendu lié aux changements de voisin-père. En effet, un changement de préfixe résulte forcément d'un changement de voisin-père. Cependant, un changement de voisin-père n'occasionne pas obligatoirement un changement de préfixe. En effet, un nœud peut décider de sélectionner un nouveau voisin-père si celui-ci annonce une distance plus intéressante (que le précédent voisin-père) vers sa passerelle courante. Il y a alors simplement une optimisation du routage entre un nœud et sa passerelle. De plus avec les algorithmes de stabilité, un nœud peut sélectionner un nouveau voisin-père même si celui-ci indique une distance plus élevée que les autres voisins du nœud. L'objectif principal de ces algorithmes est en effet de conserver un préfixe aussi longtemps que possible. Nous avons donc étudié le nombre de changements de voisin-père occasionnés par chaque algorithme, en distinguant ceux occasionnant un changement de préfixe de ceux qui n'en occasionnent pas. La figure 6.4 présente le nombre moyen de changements de préfixe (changements de voisin-père occasionnant un changement de préfixe), et la figure 6.5 présente le nombre moyen de mises à jour du préfixe (changements de voisin-père sans changement de préfixe).

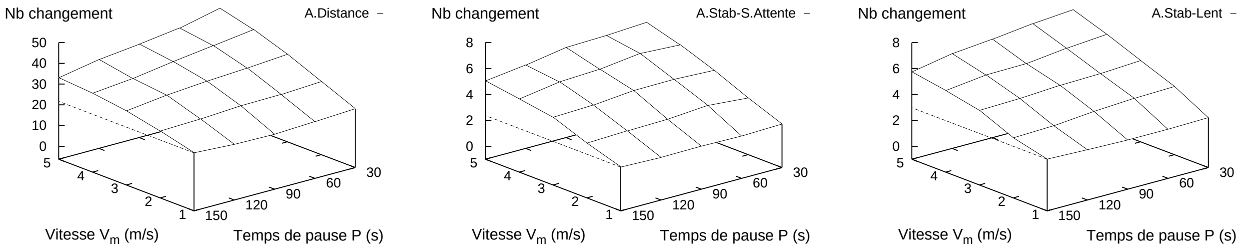

Fig. 6.4 – Nombre moyen de changements de préfixe

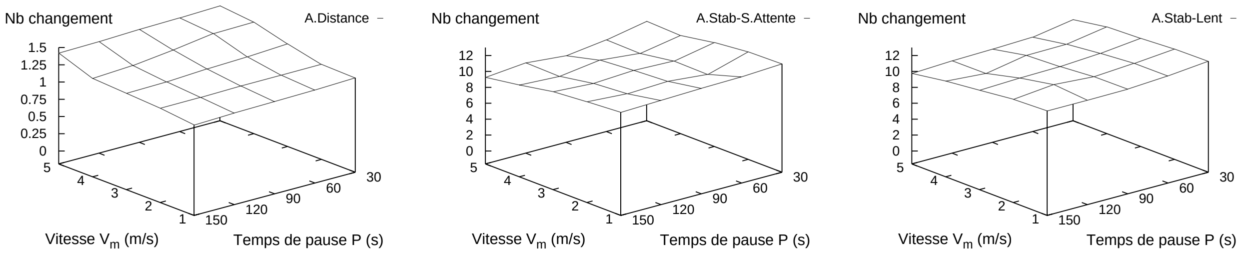

Fig. 6.5 – Nombre moyen de mises à jour du préfixe

On constate dans un premier temps que l'algorithme-distance induit un grand nombre de changements de préfixe mais très peu de mises à jour. Le nombre de changements de préfixe est en effet compris entre 20 et 50, alors que le nombre moyen de mises à jour est très faible et reste inférieur à 1,5. Cela indique qu'avec cet algorithme, les nœuds du réseau ad hoc sont régulièrement capables de trouver une meilleure route vers une nouvelle passerelle. Nous pensons que ce comportement est en partie une conséquence de la distribution géographique des nœuds lorsque le modèle de mobilité RWP est utilisé. En effet, lorsque ce modèle est utilisé avec une aire de simulation carrée ou rectangulaire, les nœuds ont tendance à se concentrer dans un disque (ou un ovale) centré dans cette aire [137]. Si l'on divise l'aire de simulation que nous utilisons en quatre carrés de même taille contenant chacun une passerelle, un nœud va généralement *osciller* entre ces quatre zones et va en conséquence passer moins de temps à se déplacer à l'intérieur d'une zone. Etant donné que tous les nœuds ont une portée radio identique et que l'aire de repos est symétrique d'un point de vue du placement des passerelles, chaque zone géographique correspond à peu près à un sous-réseau IPv6 ([125]). Les déplacements des nœuds induisent donc plus de changements de préfixe que de mises à jour du préfixe.

Au contraire, les algorithmes de stabilité induisent plus de mises à jour de préfixe que de changements de passerelle. En comparaison, ces algorithmes génèrent entre 6 et 10 fois plus de mises à jour de préfixe que l'algorithme-distance. Ce resultat est consistant avec la nature même des algorithmes de stabilité dont l'objectif est de maintenir le préfixe courant aussi longtemps que possible. On remarque également que l'algorithme de stabilité lent induit très légèrement moins de changements de préfixe que l'algorithme de stabilité sans attente. Cependant, on montrera plus loin que cette différence provient de la réactivité plus lente de cet algorithme. Enfin, les algorithmes de stabilité génèrent un total de changements de voisin-père inférieur au nombre de changements total induit par l'algorithme-distance. Cet algorithme souffre en effet d'un effet *ping-pong* dû au modèle de mobilité utilisé. Les algorithme de stabilité permettent donc de diminuer de manière significative le nombre de changements de voisin-père en évitant cette oscillation.

6.3.3 Taille des sous-réseaux

Nous avons également étudié la taille des sous-réseaux générés par chaque algorithme. En effet, dans un environnement multi-préfixes il est important que le processus d'autoconfiguration conduise à la création de sous-réseaux équilibrés afin de répartir sur les différentes passerelles disponibles la charge générée par la transmission des données. Nous avons ainsi mesuré la taille de chaque sous-réseau toutes les deux secondes pendant chacune des simulations. Les déplacements des nœuds induisent en effet des variations fréquentes de la taille des sous-réseaux. Il est cependant difficile de représenter l'évolution de la taille d'un sous-réseau sans avoir recours à un très grand nombre de graphiques. Nous avons donc décidé de calculer la taille moyenne de chaque sous-réseau (indépendamment de la taille des autres sous-réseaux) au cours du temps, et ce pour les 25 combinaisons $(V_m\,;P)$ possibles. Nous obtenons donc 100 valeurs moyennes pour chaque algorithme. Ces données sont présentées par la figure 6.6.

On constate qu'avec l'algorithme-distance, la taille moyenne d'un sous-réseau est très proche de la valeur *idéale* de 25 nœuds par sous-réseau qui conduit à la répartition équilibrée des 100 stations mobiles dans les sous-réseaux. Comme décrit précédemment, la distribution géographique des nœuds et la symétrie de l'aire de simulation par rapport aux positions des passerelles induisent une répartition uniforme des stations dans chacune des quatre zones associées aux passerelles. Comme chaque nœud choisit la passerelle la plus proche, un nombre sta-

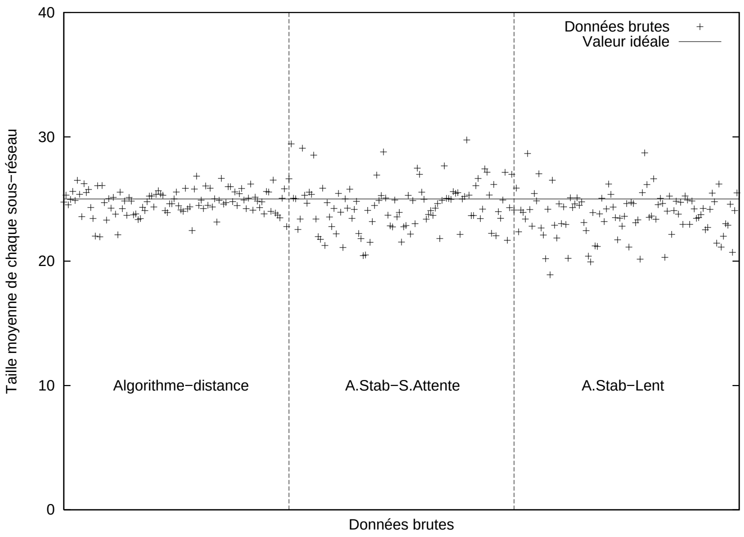

FIG. 6.6 – Taille moyenne de chaque sous-réseau

tistiquement égal de nœuds s'associe à chacune des quatre passerelles. Lors des déplacements, un même nombre de nœuds (statistiquement parlant) rejoint et quitte un sous-réseau donné : l'équilibre est donc maintenu. Au contraire, avec les algorithmes de stabilité on remarque que la taille moyenne des sous-réseaux est plus dispersée autour de la valeur idéale. L'équilibre est plus difficile car les nœuds maintiennent leur préfixe courant aussi longtemps que possible. Le préfixe utilisé par un nœud est donc moins influencé par sa position géographique. On constate également que pour les trois algorithmes la taille moyenne est généralement inférieure à la taille idéale. La somme des nœuds des quatre sous-réseaux est en effet presque toujours légèrement inférieure au nombre total de nœuds dans le réseau. Quelques nœuds n'appartiennent donc à aucun sous-réseau : nous les nommons nœuds *orphelins* et les étudions plus en détails à la fin de cette section.

Afin d'affiner cette analyse, nous avons également calculé le nombre de fois où une certaine taille a été mesurée. La seule moyenne ne donne en effet qu'une indication globale. Ces statistiques sont illustrées par la figure 6.7 qui présentent ces résultats pour les deux combinaisons extrêmes $(V_m\,;P)$ déjà considérées dans une section précédente. En fait, une courbe de la figure 6.7 correspond aux données ayant conduit au calcul de quatre points de la figure 6.6. La figure 6.7 permet en particulier de constater qu'avec l'algorithme-distance, la distribution de la taille instantanée des sous-réseaux suit une courbe Gaussienne centrée à une taille d'environ 25 nœuds. Cette constatation renforce l'affirmation selon laquelle l'algorithme-distance permet d'équilibrer la taille des sous-réseaux autour de la valeur idéale. De plus, on voit que l'influence de la mobilité est quasi nulle. Au contraire, les distributions obtenues à partir des algorithmes de stabilité ne permettent pas de tirer des conclusions précises en ce qui concerne la taille instantanée des sous-réseaux lorsqu'on utilise ces deux algorithmes.

Enfin, la figure 6.8 présente le nombre moyen de nœuds orphelins pour chacune des 25 combinaisons $(V_m\,;P)$ et pour chacun des trois algorithmes. Pour un algorithme donné, les points sont représentés de telle sorte que la vitesse de mobilité augmente lorsque l'on considère les points de gauche à droite. Chaque groupe de cinq points représente une valeur de V_m, et les valeurs successives de P. Comme précédemment, nous avons mesuré le nombre de nœuds

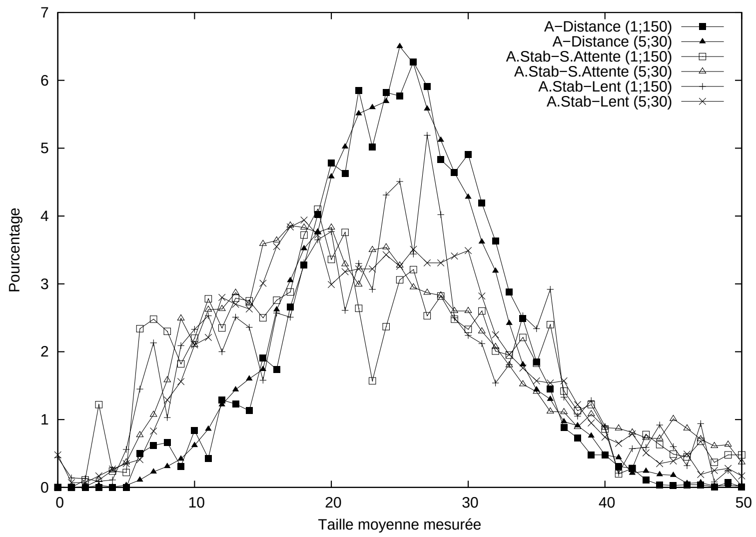

FIG. 6.7 – Distribution de la taille instantanée des sous-réseaux

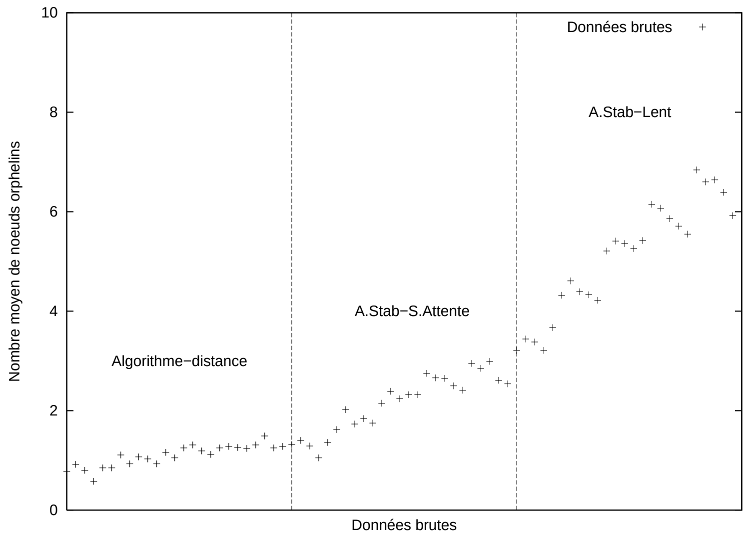

FIG. 6.8 – Nombre moyen de nœuds orphelins

orphelins toutes les 2 secondes.

Une première constatation est que le nombre de nœuds orphelins résultant de l'utilisation de l'algorithme-distance est le minimum possible. En effet avec cet algorithme, un nœud orphelin est forcément un nœud isolé au niveau de la couverture radio. Dans un tel cas, un nœud ne peut pas choisir de voisin-père et il ne pourra faire partie d'aucun sous-réseau. La seule possibilité d'éviter la présence de nœuds orphelins est d'augmenter la portée de la transmission radio. Cet isolement n'est donc pas une conséquence de notre proposition. Par contre, le nombre moyen de nœuds orphelins augmente lorsque l'on utilise les algorithmes de stabilité. Pour l'algorithme de stabilité sans attente, nous avons constaté que cette augmentation provient du cas de figure suivant. Lorsqu'un nœud A commence à ne plus recevoir de messages GW_INFO de son voisin-père, il refusera de choisir un nouveau nœud B comme voisin-père si celui-ci annonce un préfixe différent de son préfixe courant, et ce même si la distance annoncée est plus intéressante. Si le nœud A continue à se déplacer, il peut s'éloigner du nœud B jusqu'à être complètement isolé. Pendant le laps de temps au cours duquel les nœuds A et B pouvaient communiquer, le nœud A n'a pas sélectionné le nœud B comme nouveau voisin-père. A contrario, avec l'algorithme-distance, le nœud A aurait choisi le nœud B comme nouveau voisin-père dès que possible. Les mesures du nombre de nœuds orphelins étant effectuées à intervalle de temps régulier, il est possible à un instant donné de considérer A comme un nœud orphelin si un algorithme de stabiltité est utilisé alors que ce même nœud ne sera pas isolé avec l'algorithme-distance. Enfin, on constate également que le nombre de nœuds orphelins augmente avec l'algorithme de stabilité lent. Cette augmentation est due au fait que lors de la procédure de bootstrap, un nœud attend de recevoir un message GW_INFO des voisins dont il a décidé de sélectionner le préfixe. La réactivité n'est donc pas immédiate. De plus, si le nœud s'éloigne des voisins dont il espère recevoir un message GW_INFO, il devra attendre l'expiration d'un temporisateur avant de décider de sélectionner comme nouveau voisin-père l'émetteur du prochain message GW_INFO reçu. Pendant ces phases transitoires, un nœud peut donc être orphelin même si il n'est pas isolé au niveau des transmissions radio.

6.3.4 Caractéristiques topologiques des sous-réseaux

Afin de pouvoir évaluer l'influence des algorithmes de sélection du voisin-père sur les topologies des sous-réseaux, nous avons également étudié deux caractéristiques topologiques de ces sous-réseaux. Nous avons dans un premier temps mesuré la longueur (en nombre de sauts) des routes entre chaque nœud et sa passerelle. Cet indicateur nous permet en effet de mesurer l'influence des algorithmes de stabilité qui ne garantissent pas l'utilisation de la route topologique la plus courte entre un nœud et sa passerelle. Nous avons également mesuré le degré moyen des nœuds d'un sous-réseau afin d'évaluer l'ampleur des interférences causées par l'envoi de données. Une transmission est en effet exclusive : aucun autre nœud dans le voisinage de l'émetteur ne peut envoyer ou recevoir des données en même temps.

Nous avons ainsi mesuré la longueur r de la route entre tous les nœuds d'un sous-réseau et leur passerelle. La figure 6.9 montre le pourcentage de routes de longueur r mesurées toutes les 2 secondes au cours des simulations. De plus, nous présentons ici les mesures pour un seul sous-réseau car elles sont relativement identiques pour tous les sous-réseaux. Enfin, nous présentons comme précédemment les résultats pour les deux combinaisons $(V_m\,;P)$ extrêmes.

On remarque que globalement, les trois algorithmes induisent des routes de longueur équivalente. De plus la mobilité des nœuds n'a quasiment pas d'influence sur ces résultats. Nous avons également calculé la longueur moyenne des routes pour chaque algorithme et pour toutes les combinaisons : cette valeur est égale à 3,11 pour l'algorithme-distance, 3,22 pour l'algorithme de stabilité sans attente, et 3,23 pour l'algorithme de stabilité lent. Les routes sont donc en moyenne 3,5% plus longues avec les algorithmes de stabilité qu'avec l'algorithme-distance. Cette très faible différence provient du fait qu'avec les algorithmes de stabilité un nœud recherche le chemin le plus court au sein du sous-réseau : il utilise donc toujours la route la plus courte vers la passerelle si l'on considère uniquement les nœuds de son sous-réseau. De plus, dans les simulations de réseaux ad hoc, la densité des nœuds est généralement assez élevée pour éviter de trop fréquents partitionnements du réseau. La principale conséquence de cette forte densité est la multiplication des routes alternatives entre un nœud et sa passerelle ce

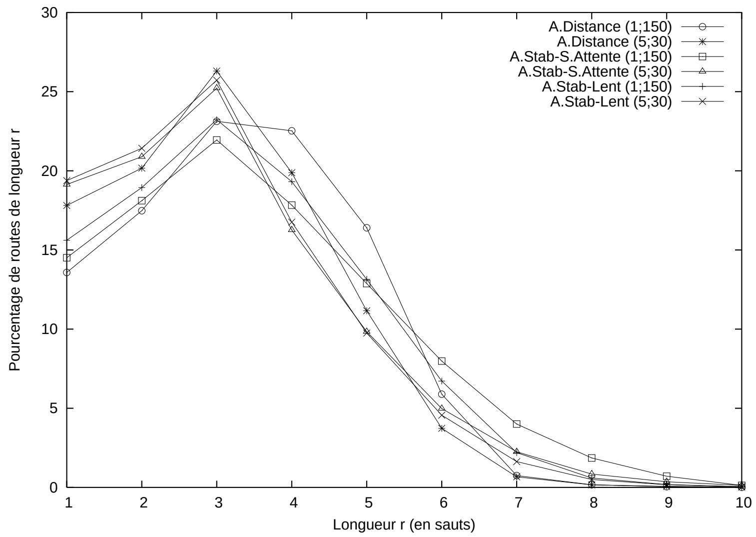

FIG. 6.9 – Distribution de la longueur r des routes des nœuds à leur passerelle

qui permet généralement à un nœud de trouver une route topologique de plus court chemin vers sa passerelle. La principale conclusion de ces mesures est que les algorithmes de stabilité n'induisent pas nécessairement le rallongement des routes entre les nœuds et les passerelles.

Enfin la figure 6.10 montre le pourcentage de nœuds ayant un degré d avec ses voisins du même sous-réseau. Comme avec la longueur des routes, nous présentons les mesures pour un sous-réseau car elles sont significatives pour tous les sous-réseaux. Comme précédemment, nous avons mesuré le degré des nœuds au sein d'un sous-réseau toutes les 2 secondes au cours des simulations. Enfin, nous présentons de même les résultats pour les deux combinaisons $(V_m;P)$ extrêmes.

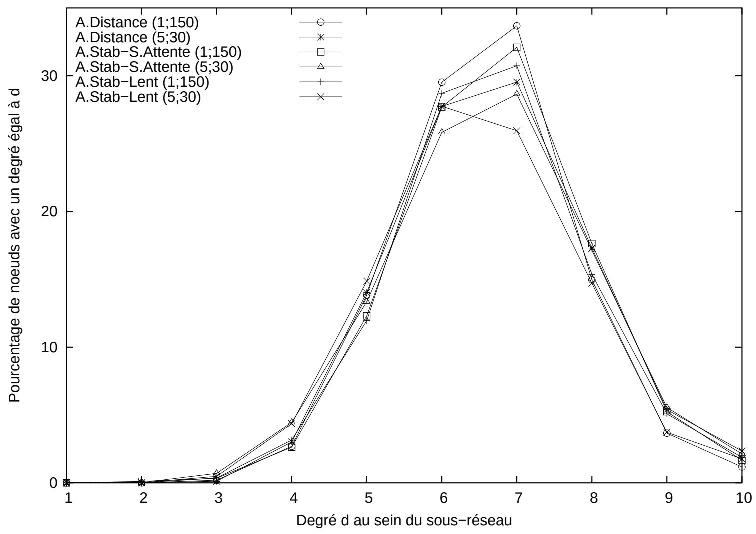

FIG. 6.10 – Distribution du degré d des nœuds d'un sous-réseau

On remarque que le degré des nœuds au sein d'un sous-réseau est identique quel que soit l'algorithme utilisé. De plus, la vitesse de mobilité n'a pas d'influence sur ces résultats. On note que les nœuds ont un degré relativement élevé avec une majorité de nœuds ayant un degré égal à 6 ou 7. Le degré d'un nœud est en effet principalement lié à la densité des nœuds présents dans l'aire de simulation et à la portée des transmissions radio [140]. De plus, la distribution du degré au sein d'un réseau ad hoc (on peut en effet considérer ici un sous-réseau comme un réseau ad hoc autonome) suit généralement une loi Gaussienne [140] comme sur la figure 6.10. L'influence de la topologie est donc prépondérante et le degré des nœuds au sein d'un sous-réseau est indépendant de la taille des sous-réseaux : en effet, seul le voisinage direct d'un nœud influe sur la valeur du degré. Dans la majorité des cas, ce voisinage fait partie du même sous-réseau : seuls les nœuds étant situés à la bordure d'un sous-réseau peuvent présenter des valeurs plus faibles. Sur la figure 6.10 on constate en effet qu'un plus faible pourcentage de nœuds présentent un degré égal à 5. Enfin, les nœuds situés au centre (géographique) d'un sous-réseau vont présenter un degré plus élevé. La principale conclusion de cette analyse est que les algorithmes n'ont pas d'influence sur le degré des nœuds au sein de leur sous-réseau.

6.3.5 Impact des algorithmes

Enfin nous avons voulu évaluer l'impact des changements de préfixe sur les nœuds situés en aval d'un changement. En effet, un nœud est directement affecté par les changements de préfixe de son voisin-père ou des nœuds situés en amont vers la passerelle. Nous avons évalué cet impact en introduisant la notion de changement initial. Nous définissons un changement initial comme un changement de préfixe n'ayant pas été causé par un autre changement de préfixe. Un changement initial de préfixe peut en particulier se produire lorsque

- le lien entre un nœud et son voisin-père est rompu : ce nœud décide alors de choisir un nouveau voisin-père utilisant un préfixe différent. Ce cas de figure est possible avec tous les algorithmes.
- un nœud sélectionne un nouveau voisin-père annonçant une distance plus intéressante vers une passerelle utilisant un préfixe différent. Ce cas de figure est uniquement possible avec l'algorithme-distance.

Nous avons mesuré cet impact avec les trois algorithmes de sélection du voisin-père et avec les deux combinaisons (V_m ;P) déjà considérées précédemment. Les résultats de ces mesures sont présentés par la figure 6.11.

Non représentés sur cette figure, nous avons mesuré qu'entre 70 et 80 % des changements initiaux n'ont aucun impact, et ce quel que soit l'algorithme utilisé. On remarque de plus que l'algorithme utilisé ainsi que la mobilité des nœuds ont une très faible influence sur cet indicateur. Enfin, nous avons calculé que l'impact moyen d'un changement initial est égal à 3,01 pour l'algorithme-distance, 3,69 pour l'algorithme de stabilité sans attente, et 3,63 pour l'algorithme de stabilité lent. Il peut sembler surprenant que les algorithmes de stabilité aient un impact plus important que l'algorithme-distance. En effet, avec ces algorithmes un nœud essaie de ne pas changer de préfixe si son voisin-père change de préfixe ou si il n'est plus joignable. En fait, nous avons observé que ces algorithmes conduisent parfois à la création de longues *branches* de nœuds. Lorsque le nœud le plus en amont d'une telle branche change de préfixe, tous les nœuds en aval sont forcés d'utiliser un nouveau préfixe. L'impact d'un changement initial est donc parfois très élevé, et la moyenne est donc supérieure à celle calculée

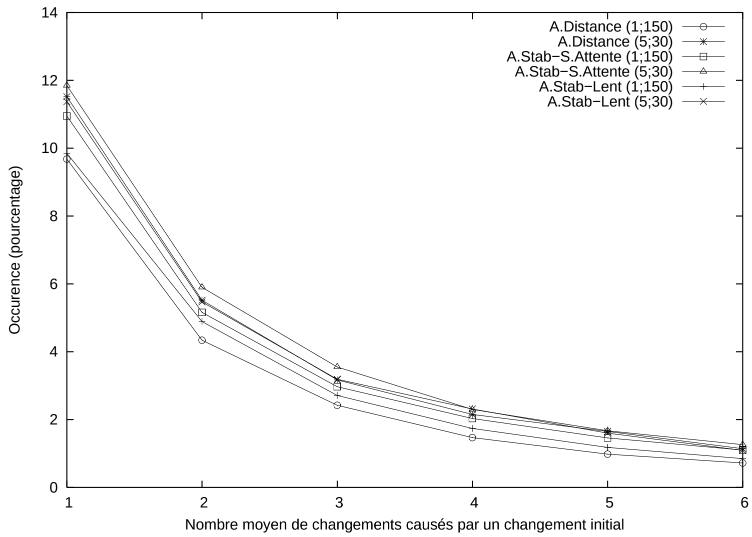

FIG. 6.11 – Distribution du nombre moyen de changements de préfixe occasionnés par un changement initial

avec l'algorithme-distance. On note cependant que l'impact causé par les trois algorithmes est assez faible : les changements de préfixe sont donc très localisés et on n'assiste que très rarement à des changements majeurs de la configuration logique (au sens IP) du réseau.

6.4 Isolement des nœuds

Dans la section précédente nous avons mesuré divers indicateurs afin d'évaluer les performances des différents algorithmes. Nous utilisons à présent certaines de ces mesures afin d'estimer le temps T_i (temps d'isolement) pendant lequel un nœud ne peut plus communiquer avec des correspondants de l'Internet. Nous calculons une unique valeur de T_i pour l'ensemble des différentes combinaisons de temps de pause et de vitesse de déplacement.

Dans un premier temps, nous avons calculé le temps moyen pendant lequel un nœud n'est plus capable de communiquer avec sa passerelle. Ce temps comprend les périodes pendant lesquelles un nœud est orphelin ainsi que celles pendant lesquelles la route entre un nœud et sa passerelle est inutilisable (c'est à dire lorsqu'un lien sur cette route est rompu). Chaque algorithme de sélection du voisin-père doit normalement réagir à ce type d'événement. Lorsqu'il y a rupture d'un lien, le voisin-père respectif des nœuds situés en aval de la rupture n'envoie plus de messages GW_INFO et après un certain délai il n'est donc plus utilisé comme voisin-père. Cependant, l'algorithme-distance est moins restrictif que les algorithmes de stabilité et il peut permettre à un nœud de trouver une alternative plus rapidement si par exemple une nouvelle route vers une passerelle plus proche apparaît. De surcroît l'algorithme de stabilité lent introduit de par sa nature même un délai de réactivité supplémentaire. Afin de mesurer le temps moyen T_l pendant lequel un nœud est isolé de sa passerelle et donc de l'Internet, nous avons utilisé un mécanisme d'écho/réponse : chaque nœud envoie périodiquement un message d'écho vers sa passerelle qui y répond lorsque celle-ci est joignable. Si un lien de la route entre un nœud et sa passerelle est rompu, le nœud ne recevra pas de réponse à sa requête. La valeur de T_l est donc le temps pendant lequel un nœud n'est pas joignable depuis l'Internet car soit il n'y a pas de route entre ce nœud et sa passerelle, soit l'algorithme utilisé ne réagit pas assez rapidement à un changement topologique (il existe une route mais le nœud ne l'utilise pas). On précise que T_l est le temps moyen total (en secondes) pendant lequel un nœud n'est pas joignable au cours des 900 secondes d'une simulation. La table 6.1 présente les résultats de ces mesures. Comme nous l'avons déjà mentionné (voir section 6.3.3), on remarque que l'algorithme-distance minimise le temps moyen pendant lequel un nœud n'a plus de lien vers sa passerelle car cet algorithme est le plus réactif. On note également que l'agorithme de stabilité lent est très pénalisant alors que dans les conditions de simulation que nous considérons il ne présente pas de gain significatif par rapport à l'algorithme de stabilité sans attente.

	A.Distance	**A.Stab-S.Attente**	**A.Stab-Lent**
T_l (s)	9,83	19,34	45,86

TAB. 6.1 – Temps moyen sans lien vers une passerelle

D'autre part, un nœud n'est également pas joignable lorsqu'il change de préfixe. En effet, lorsqu'un nœud change d'adresse globale, les paquets envoyés depuis l'Internet continuent à être routés vers l'ancienne adresse du nœud et ils n'arrivent donc pas à destination. Le nœud doit en effet informer ses correspondants et/ou son agent mère de sa nouvelle adresse en leur envoyant un message Mobile IPv6 de mise à jour de l'association. Le délai introduit par cette procédure est proportionnel à la distance séparant l'émetteur de ce message de son destinataire. Il dépend également de la rapidité de traitement avec laquelle le destinataire prend en compte le message. Nous notons ce délai T_{BU} (pour Binding Update). Or comme nous l'avons montré dans la section précédente, chaque algorithme génère un nombre différent de changements de préfixe. Le nombre moyen de changements de préfixe N_{BU} induit par chaque algorithme au cours d'une simulation est présenté dans la table 6.2. Au cours d'une simulation, le temps d'isolement résultant des changements de préfixe est donc égal à $N_{BU} * T_{BU}$. On note également que T_{BU} est indépendant de l'algorithme utilisé dans le réseau. En effet, nous avons montré que la longueur de la route entre un nœud et sa passerelle n'est que 3,5% plus longue avec les algorithmes de stabilité par rapport à l'algorithme-distance : nous pouvons donc ignorer ce paramètre.

	A.Distance	**A.Stab-S.Attente**	**A.Stab-Lent**
N_{BU}	31,49	4,23	4,84

TAB. 6.2 – Nombre moyen de changements de préfixe

Au cours d'une simulation, nous pouvons donc calculer le temps total pendant lequel un nœud n'est pas joignable depuis l'Internet. Ce temps est égal à $T_i = T_l + N_{BU} * T_{BU}$. Pour chacun des trois algorithmes, nous avons calculé ce temps au cours des simulations en considérant différentes valeurs pour T_{BU} (le délai nécessaire afin d'effectuer l'enregistrement auprès des correspondants et de l'agent mère) que nous avons fait varier de 10 ms à 1 seconde.

La figure 6.12 présente le pourcentage du temps total d'une simulation pendant lequel un nœud n'est pas joignable depuis l'Internet. On remarque que par rapport au temps pendant lequel un nœud est isolé de l'Internet, l'algorithme-distance est plus intéressant que les algorithmes de stabilité lorsque la valeur de T_{BU} est faible. Le nombre de changements de préfixe mesuré avec chaque algorithme a en effet une influence proportionnelle à la valeur de T_{BU}. Avec l'algorithme-distance, le temps d'isolement d'un nœud s'accroît ainsi très rapidement et devient supérieur au temps mesuré avec l'algorithme de stabilité sans attente lorsque $T_{BU} > 350$ ms. La croissance du temps d'isolement calculé avec les algorithmes de stabilité est en effet faible car ils génèrent peu de changements de préfixe et l'influence de T_{BU} est en conséquence moindre. L'algorithme de stabilité lent est cependant fortement pénalisé par la forte valeur de T_l associée à cet algorithme.

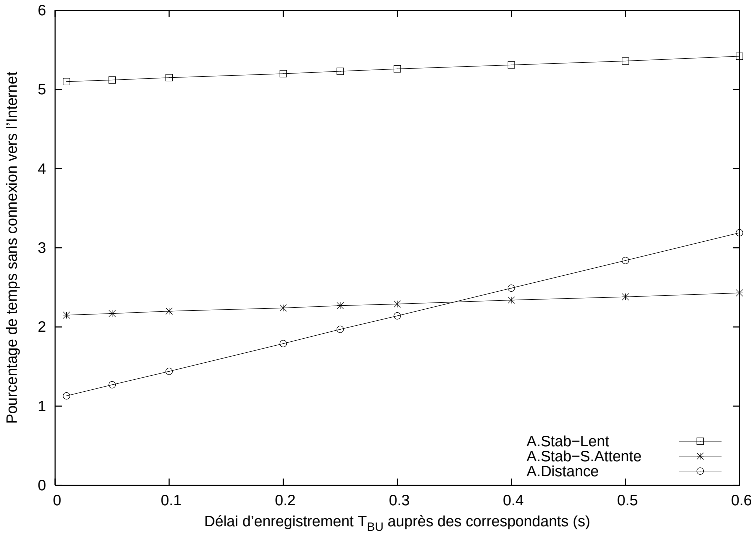

FIG. 6.12 – Isolement d'un nœud

Lorsque T_{BU} est inférieur à 350 ms, il est donc plus intéressant pour les nœuds d'utiliser l'algorithme-distance car le temps moyen d'isolement est minimal. Cependant, les changements de préfixe sont nombreux et l'organisation des nœuds en sous-réseaux est donc relativement instable et fluctuante. L'algorithme de stabilité sans attente apparaît donc comme une alternative très intéressante car il est bien moins sensible à la valeur de T_{BU} et il génére très peu de changements de préfixe. On note qu'avec cet algorithme lorsque $T_{BU} = 600$ ms, un nœud est en moyenne isolé au maximum pendant environ 2,4% du temps total de la simulation ce qui correspond à environ 22 secondes pour une simulation de 900 secondes (15 mn). De plus on rappelle que si un nœud n'est pas orphelin, le temps de convergence maximal des algorithmes est de 5 secondes (voir section 5.3.5). Si un nœud n'est jamais orphelin et si l'on considère qu'un temps d'isolement long est très pénalisant pour les applications, un nœud sera dans le pire des cas isolé quatre fois pendant 5,6 secondes (5 secondes pour la convergence et 600 ms pour la mise à jour de l'association). Sur un total de 15 mn, ces interruptions semblent toutefois acceptables. Nous rappelons qu'il est cependant possible d'augmenter la réactivité des algorithmes en augmentant la fréquence d'envoi des messages GW_INFO, cette valeur étant un compromis entre réactivité des nœuds et surcharge du réseau.

6.5 Conclusion

Nous avons évalué et comparé dans ce chapitre les performances des algorithmes de sélection du voisin-père de notre proposition d'autoconfiguration d'adresses. Nous avons en particulier validé le fonctionnement des algorithmes de stabilité qui permettent d'augmenter de manière substantielle le temps pendant lequel un nœud conserve son préfixe courant. De plus, les différentes simulations ont mis en évidence la capacité de notre protocole à réagir avec succès aux changements topologiques d'un réseau ad hoc et à créer des sous-réseaux contigus respectant la continuité de préfixe. Nous avons également montré que les algorithmes de sélection du voisin-père ont une faible influence sur les caractéristiques topologiques des sous-réseaux créés. Enfin, nous avons calculé le temps moyen pendant lequel un nœud du réseau ad hoc n'est plus joignable depuis l'Internet. Pour conclure, les avantages et les inconvénients des algorithmes de sélection du voisin-père sont résumés par le tableau 6.3.

	Algorithme-distance	**Algorithmes de stabilité**
Avantages	- Chemin topologique le plus court (vers passerelles) - Taille relativement stable des sous-réseaux - Répartition équilibrée des nœuds dans les sous-réseaux	- Temps de maintien d'un préfixe maximisé - Chemin vers la passerelle le plus court au sein du sous-réseau - Temps d'isolement très peu influencé par T_{BU}
Inconvénients	- Changements plus fréquents de préfixe - Peut souffrir d'un effet d'oscillation	- Pas nécessairement de plus court chemin topologique (vers passerelles) - Augmentation du nombre de nœuds orphelins

TAB. 6.3 – Avantages et inconvénients des algorithmes de sélection du voisin-père

Dans le chapitre suivant, nous appliquons le concept de continuité de préfixe aux communications de groupe. Nous évaluons en particulier l'impact de la continuité de préfixe sur les arbres multicast si ceux-ci sont construits avec des nœuds utilisant tous le même préfixe.

Chapitre 7

Arbres multicast et continuité de préfixe

7.1 Introduction

Après avoir proposé un protocole d'autoconfiguration d'adresses basé sur le concept de continuité de préfixe, nous appliquons ce concept aux communications de groupe. Les protocoles de routage multicast pour les réseaux ad hoc ne permettent en effet pas de garantir que tous les membres d'un arbre multicast partagent le même préfixe. Dans un réseau ad hoc respectant la continuité de préfixe, cette particularité peut néanmoins être très intéressante. Premièrement, la continuité de préfixe interdit le transit de données multicast entre sous-réseaux par l'intermédiaire du réseau ad hoc. Seuls les nœuds d'un même réseau collaborent pour transporter des données multicast à partir ou à destination d'une station de leur propre sous-réseau. Le transit de données entre sous-réseaux (ad hoc) est en effet effectué par l'intermédiaire des passerelles et du réseau filaire. La restriction au niveau du transit de données multicast permet en particulier d'économiser des ressources (bande-passante, énergie) dans le réseau ad hoc en forçant l'utilisation du réseau filaire. Il est également possible de réaliser des opérations de management et de facturation puisque les passerelles sont un point de transit obligatoire pour le trafic multicast entre sous-réseaux. Nous présentons ces motivations dans la prochaine section.

Deuxièmement et ce contrairement à la majorité des protocoles existants, nous proposons une technique d'adhésion qui évite l'inondation périodique du réseau ad hoc avec des messages d'annonce. Nous proposons en effet d'utiliser les passerelles comme *points de rendez-vous* entre les récepteurs et les sources. Il devient ainsi possible de réaliser des opérations de contrôle et de filtrage lors de l'adhésion à des groupes multicast. Chaque passerelle peut de plus proposer un système de facturation collaboratif : chaque membre d'un groupe multicast est mis à contribution afin de *rémunérer* les nœuds du réseau participant à la transmission des données multicast. Chaque nœud d'un arbre de diffusion pourrait ainsi obtenir un *crédit de service* auprès des membres du groupe multicast pour lequel il retransmet les données. De cette façon, il est possible de promouvoir la collaboration des nœuds : si un nœud *rend un service* à une autre station il sait qu'il pourra bénéficier en retour d'une collaboration lorsqu'il en aura besoin. Ce système pourrait enfin intégrer la diffusion multicast mais aussi la réservation de bande-passante et la qualité de service. Le *crédit de service* de chacun des nœuds d'un sous-réseau serait centralisé au niveau de la passerelle.

De plus, on peut envisager l'utilisation de techniques de transcodage de données au niveau des passerelles. Il est ainsi possible d'effectuer une conversion du format des données multicast afin par exemple de sécuriser l'information transmise [141] ou afin d'utiliser un codage adapté à la puissance de calcul des terminaux mobiles [142]. La passerelle peut également réduire le débit de transmission des données afin de réduire la consommation de la bande-passante dans le réseau ad hoc. Enfin, il est possible d'adapter la procédure de transmission des données (taille des paquets, séquencement des transmissions) afin de permettre aux membres de l'arbre de diffusion de réduire le coût énergétique de la transmission des données [143].

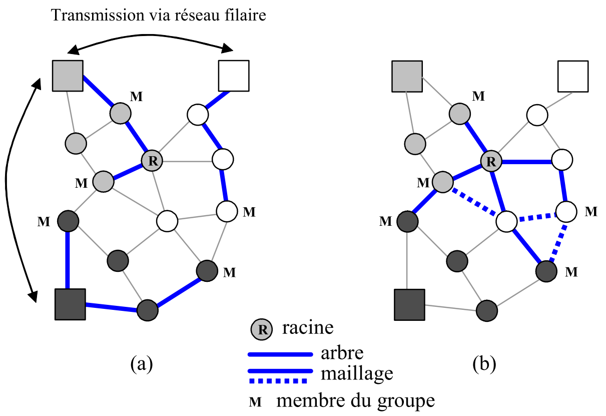

FIG. 7.1 – Structure basée (a) ou non (b) sur la continuité de préfixe

La figure 7.1 illustre l'application du concept de continuité de préfixe aux communications multicast. Cette figure représente un réseau ad hoc avec trois passerelles. Dans le premier cas (a), la structure de diffusion des données multicast respecte la continuité de préfixe : chaque *sous-arbre* est en effet constitué de nœuds utilisant tous le même préfixe. La transmission des données entre sous-réseaux (ad hoc) est réalisée par l'intermédiaire de l'infrastructure de routage multicast utilisée dans le réseau filaire (par exemple grâce au protocole PIM-SM). Au contraire dans le deuxième cas (b), la structure de diffusion multicast est formée par des nœuds n'utilisant pas forcément le même préfixe (on notera que cette structure peut être un arbre ou un maillage). On remarque que dans les deux cas le réseau ad hoc est organisé en sous-réseaux contigus respectant la continuité de préfixe. La continuité de préfixe n'induit donc pas forcément la création de structures de diffusion multicast formées par des nœuds utilisant tous le même préfixe.

Dans ce chapitre, notre objectif principal est d'évaluer l'impact de la continuité de préfixe sur les structures de diffusion multicast. Nous voulons en particulier étudier certaines des caractéristiques topologiques de ces structures afin d'étudier l'efficacité du routage multicast respectant la continuité de préfixe. Notre objectif n'est donc pas d'étudier les performances d'un protocole de routage multicast (convergence, réactivité, efficacité) permettant de respecter la continuité de préfixe : nous nous intéressons uniquement à la topologie des structures de diffusion ainsi créées.

7.2 Routage entre sous-réseaux

Jusqu'à présent, nous n'avons jamais parlé du routage unicast ou multicast entre les sous-réseaux d'un réseau ad hoc. Sans continuité de préfixe et donc sans sous-réseaux IP, cette notion n'a pas lieu d'être car les nœuds utilisant différents préfixes sont *mélangés* dans la topologie physique. D'un point de vue topologique il n'y pas de sous-réseaux et tous les nœuds routent les paquets unicast et multicast quel que soit le préfixe utilisé par l'émetteur. D'un autre côté, si l'on considère un réseau ad hoc organisé en sous-réseaux connexes, deux approches sont envisageables. Avec la première alternative, le routage *direct* entre les sous-réseaux du réseau ad hoc est autorisé et les protocoles de routage unicast et multicast permettent à deux nœuds utilisant des préfixes différents de communiquer directement via le réseau ad hoc. Au contraire avec la seconde approche un tel routage est interdit et deux nœuds de sous-réseaux différents doivent communiquer via leurs passerelles et via le réseau d'infrastructure (généralement filaire) les reliant. Les spécifications actuelles de l'architecture des réseaux ad hoc [97] sont interprétables car la notion de continuité de préfixe n'est pas considérée. Ce document précise en effet qu'un réseau ad hoc hybride est généralement une feuille de l'Internet qui n'autorise pas le transit de paquets : les données envoyées d'un sous-réseau A vers un sous-réseau B ne peuvent pas transiter via un sous-réseau C. Cependant ces spécifications considèrent le réseau ad hoc comme une entité unique et ne mentionnent en particulier pas si le routage entre sous-réseaux d'un même réseau ad hoc est autorisé ou non.

Chacune de ces deux approches a ses avantages et ses inconvénients. Le routage direct utilise le plus court chemin entre deux nœuds du réseau ad hoc et il est a priori plus performant qu'un routage triangulaire[1] via les passerelles des correspondants. En fait on peut trouver des arguments en faveur du routage indirect. Si par exemple le réseau ad hoc est très étiré et est composé de plusieurs sous-réseaux *alignés* en une longue file, il peut être préférable d'utiliser le réseau filaire afin de faire communiquer deux machines situées à chacune des extrémités du réseau. La route directe peut être très longue (en nombre de sauts) comparée à la route via les passerelles : un paquet peut transiter via plusieurs nœuds du réseau ad hoc pour passer d'un sous-réseau à un autre alors qu'il peut n'y avoir qu'un saut entre deux passerelles si l'on considère le réseau filaire. De plus, le lien filaire offrira généralement une latence plus faible et un débit plus élevé si on le compare à la route directe composée de plusieurs nœuds du réseau ad hoc. Lorsque la portion de la route indirecte qui utilise le réseau ad hoc est plus petite que la route directe (en nombre de sauts), le routage via les passerelles est donc une alternative intéressante au routage direct.

De plus le routage indirect via les passerelles permet d'envisager le déploiement de solutions de management au niveau des passerelles. Tout le trafic de données entrant et sortant d'un sous-réseau transite via la passerelle et il est donc possible d'y réaliser par exemple des opérations de facturation. De plus si toutes les passerelles d'un réseau ad hoc appartiennent à une entité administrative unique (ex : un opérateur de télécommunications), il est possible de centraliser ces informations et d'offrir des services cohérents à un nœud qui se déplace d'un sous-réseau à un autre. Enfin comme pour les réseaux cellulaires, on peut envisager la mise en place d'accords entre opérateurs de réseaux ad hoc afin de permettre à un nœud de se connecter à un réseau ad hoc pour lequel il n'a par exemple pas d'abonnement. Toutes ces constatations nous amènent à étudier les particularités du routage multicast respectant la continuité de préfixe dans les réseaux ad hoc.

[1]Strictement parlant un tel routage ressemble plutôt à un quadrilatère

7.3 Principes du routage multicast

Dans cette section nous proposons un ensemble de règles permettant de créer des arbres (ou des maillages) multicast respectant la continuité de préfixe. Comme nous l'avons expliqué précédemment, notre objectif n'est pas d'évaluer un tel protocole de routage mais d'étudier les structures de diffusion résultant de son utilisation. Nous présentons donc uniquement les principes que doit respecter un protocole de routage multicast basé sur la continuité de préfixe. Nous proposons également l'utilisation des passerelles comme points de rencontre entre les sources et les récepteurs d'un groupe multicast.

7.3.1 Retransmission des données

Afin d'éviter le transit de données multicast entre les structures de diffusion créées dans chaque sous-réseau, un nœud N ne doit jamais retransmettre des données multicast si elles ont été retransmises (ou envoyées) par l'un de ses voisins si celui-ci ne partage pas le même préfixe réseau que N. Ainsi, il ne peut pas y avoir de *fuites* de données multicast entre sous-réseaux. Cette vérification est possible grâce à la table des voisins que maintient chaque nœud dans le cadre du protocole d'autoconfiguration d'adresses. Cependant, l'identification d'un voisin est uniquement possible grâce à l'adresse MAC source de la trame envoyée. En effet, l'adresse IP source d'un paquet est celle de l'émetteur initial et seule l'adresse MAC permet d'identifier un nœud retransmettant des données. Il faut donc utiliser l'adresse MAC d'un nœud afin de savoir si celui-ci utilise ou non le même préfixe réseau. En pratique, on peut par exemple reconstruire l'identifiant Ethernet unique (EUI-64) de l'émetteur à partir de la trame reçue et le comparer avec celui utilisé dans l'adresse IPv6 de chacun des voisins. Des techniques basées sur les adresses MAC sont utilisées dans la quasi totalité des protocoles de routage multicast créant des arbres de diffusion [117, 118, 119].

7.3.2 Procédure d'adhésion au sein du réseau ad hoc

Dans presque tous les protocoles de routage multicast pour les réseaux ad hoc, les sources ou les récepteurs annoncent leur existence en inondant la totalité du réseau ad hoc avec des messages particuliers. Ces annonces sont de plus envoyées périodiquement afin de gérer la dynamicité des groupes : il en résulte donc une surcharge directement proportionnelle au nombre de membres émettant ces messages. Dans un réseau ad hoc hybride, on peut cependant utiliser les passerelles afin de mettre en relation les sources et les récepteurs d'un sous-réseau ad hoc. Ainsi, chaque source s'enregistre périodiquement auprès de sa passerelle. En cas de changement de passerelle, un message d'enregistrement est immédiatement envoyé à la nouvelle passerelle. Lorsqu'un récepteur désire adhérer à un groupe, il envoie une requête d'adhésion à sa passerelle. Si il existe des sources dans le sous-réseau du récepteur, la passerelle peut envoyer à ce dernier un message contenant la liste de ces sources. Le récepteur peut alors construire une branche de la structure de diffusion (arbre ou maillage) vers chacune des sources. Cette technique permet en particulier d'éviter l'inondation périodique du réseau ad hoc avec des messages d'annonce. On peut de plus envisager la mise en place d'un système de management permettant de connaître les participants d'une communication de groupe. Enfin cette technique est nécessaire pour permettre la mise en relation de sources et de récepteurs lorsque ceux-ci ne sont pas dans les mêmes sous-réseaux : dans certains cas, la passerelle du sous-réseau ad hoc doit en effet impérativement connaître la liste des membres présents dans son sous-réseau.

7.3.3 Mise en relation de membres de réseaux différents

Lorsqu'un nœud d'un sous-réseau ad hoc adhère à un groupe, il peut exister des sources extérieures au sous-réseau de ce récepteur : la procédure d'adhésion est alors particulière. Il est en effet nécessaire d'utiliser le protocole de routage du réseau filaire afin de permettre la diffusion des données entre les récepteurs et les sources de réseaux différents. En pratique seuls les protocoles PIM-SM et PIM-SSM sont déployés dans l'Internet Nouvelle Génération. Avec PIM-SSM, un récepteur adhère à un canal (S,G) et connait donc l'adresse de la source S. Si la source S est extérieure au sous-réseau du récepteur, celui-ci adhère simplement au canal via sa passerelle. Les messages de routage multicast utilisés entre le récepteur et la passerelle sont donc propres au réseau ad hoc, et ceux envoyés à partir de la passerelle seront des messages propres au protocole PIM-SSM. Dans le cas de PIM-SM, la passerelle ne connaît pas les sources hors du réseau ad hoc car avec ce protocole seuls les points de rendez-vous du domaine PIM connaissent toutes les sources. La passerelle envoie donc simplement un message d'adhésion propre au protocole PIM-SM au point de rendez-vous approprié du domaine PIM auquel elle appartient. Les mécanismes de routage du protocole PIM-SM (voir section 1.2.2) permettront à terme de créer une branche de l'arbre de diffusion PIM entre la passerelle et chacune des sources.

Pour la réciproque, c'est à dire dans le cas où une source est présente dans le réseau ad hoc et qu'il existe des récepteurs extérieurs au sous-réseau de la source, le fonctionnement est similaire. Avec PIM-SSM, un récepteur adhèrera au canal (S,G) et le message d'adhésion du protocole PIM-SSM arrivera jusqu'à la passerelle de la source S. La passerelle utilisera donc le protocole de routage multicast du réseau ad hoc afin de créer une branche de diffusion entre elle-même et la source S. Dans le cas du protocole PIM-SM, on utilise le fait que chaque source du réseau ad hoc est enregistrée auprès de sa passerelle. La passerelle peut donc regulièrement envoyer des messages PIM_REGISTER au point de rendez-vous du domaine PIM afin de l'informer de l'existence de la source. Lors de l'adhésion de récepteurs qui ne sont pas présents dans le sous-réseau (ad hoc) de la source, un message d'adhésion PIM-SM sera envoyé vers la passerelle de la source. La passerelle pourra alors construire une branche de diffusion entre elle-même et la source et les données pourront être diffusées à l'extérieur du sous-réseau ad hoc.

7.4 Environnement de simulation

Comme expliqué précédemment, notre objectif est d'étudier les caractéristiques topologiques des structures de diffusion respectant la continuité de préfixe. En particulier, nous ne cherchons pas à évaluer la méthode utilisée pour construire ces structures. En effet, on peut utiliser n'importe quel protocole de routage multicast pour les réseaux ad hoc à condition de suivre les quelques règles présentées dans la section précédente.

Comme dans la section 6.2, nous utilisons le simulateur NS et une version modifiée du

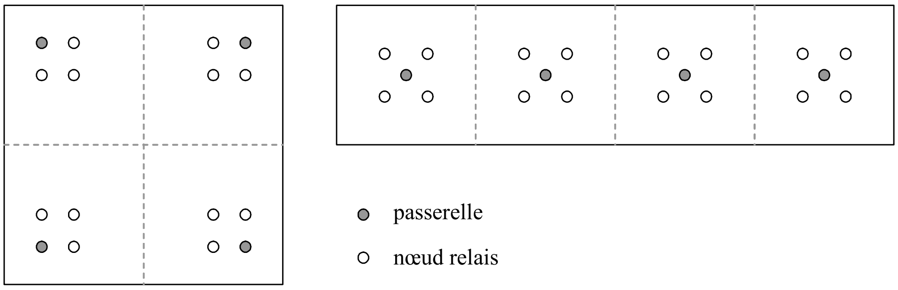

Fig. 7.2 – Architectures simulées : CARRE et TUNNEL

modèle RWP pour simuler la mobilité des nœuds. Nous utilisons également l'implémentation de NS du protocole IEEE 802.11 pour les communications radio. Enfin, nous considérons le même nombre de nœuds mobiles, ainsi que les mêmes conditions topologiques : taille de l'aire de simulation, placement des passerelles, et utilisation de nœuds relais. Dans la suite de ce chapitre, nous utiliserons le terme CARRE afin de nous référer à cette architecture (dont nous rappelons que la topologie est un carré de 2000 mètres de côté). Nous introduisons de plus une architecture supplémentaire afin de mettre en évidence l'influence de la topologie physique. Cette architecture nommée TUNNEL est un rectangle de 1000x4000 m^2, avec également 100 nœuds mobiles comme précédemment. La surface des deux architectures étant égale, on obtient une densité de nœuds mobiles identique. L'architecture TUNNEL contient également quatre passerelles, situées aux centres des quatre carrés de 1000 mètres de côté formant cette architecture. Chaque passerelle est également entourée par quatre nœuds relais qui forment un carré de 300 mètres de côté dont la passerelle est le centre. Comme précédemment (section 6.2), les nœuds relais permettent d'éviter l'isolement géographique d'une passerelle lors des déplacements des nœuds du réseau ad hoc. Ces deux architectures sont illustrées par la figure 7.2 (l'échelle des distances n'est pas respectée).

Afin d'évaluer l'impact de la continuité de préfixe appliquée à la diffusion multicast, nous avons simulé la construction de groupes multicast. Dans la suite de ce chapitre, nous utiliserons le terme arbre de diffusion pour indiquer la structure utilisée pour réaliser la diffusion multicast. Cependant et comme nous l'avons déjà expliqué dans la section 4.2, dans un réseau ad hoc un arbre ou un maillage implique généralement les mêmes nœuds car la structure de diffusion est construite dans les deux cas de la même façon. La seule différence est qu'avec un maillage un nœud peut retransmettre les données quelle que soit la station ayant émis ou réémis ces données. Afin d'étudier la topologie de ces arbres, nous avons fait varier le nombre de membres d'un groupe de 5 à 50, soit entre 5 et 50 % du nombre total des nœuds mobiles du réseau ad hoc.

Chaque arbre est construit en sélectionnant une racine parmi les membres du groupe. Nous avons dans un premier temps construit l'arbre des plus courts chemins connectant tous les membres à la racine de l'arbre. Dans la suite de ce chapitre, nous appelerons cet arbre *arbre des plus courts chemins*. Dans un deuxième temps, nous avons construit la structure de diffusion multicast en respectant la continuité de préfixe. Pour chaque membre du groupe, nous avons construit une branche de la façon suivante. Si le membre du groupe et la racine ont le même préfixe, nous rajoutons dans la structure la branche formée par le plus court chemin entre le membre et la racine. Dans le cas contraire, une première branche construite entre le membre et sa passerelle est mise en place. Une deuxième branche, celle construite entre la racine et sa passerelle est alors rajoutée. Cette structure de diffusion, qui dans notre cas et au niveau du réseau ad hoc est en fait formée par quatre arbres indépendants, est appelée *arbre avec continuité de préfixe* (même si en fait ce n'est pas strictement un arbre). Nous n'avons pas simulé l'interconnexion des quatre *sous-arbres* de cette structure car nous nous intéressons uniquement au réseau ad hoc. On notera également que les arbres que nous simulons peuvent être des arbres par source (la racine est la source des données), ou des arbres partagés (la racine est un nœud central du protocole de routage multicast).

7.5 Résultats des mesures

Nous avons utilisé un certain nombre d'indicateurs afin d'évaluer l'impact de la continuité de préfixe sur la diffusion multicast. En particulier, nous avons dans un premier temps étudié la taille de la structure de diffusion, c'est à dire le nombre de nœuds en faisant partie. Nous avons également étudié la profondeur de ces structures, c'est à dire la distance entre la racine et chacun des nœuds de l'arbre de diffusion. De plus, nous avons mesuré les interférences causées par chaque structure de diffusion, c'est à dire la gêne occasionnée par l'envoi des données. Enfin, nous avons analysé les effets de la superposition de plusieurs structures de diffusion afin de mesurer le nombre d'états de retransmission présents dans chaque nœud du réseau. On notera que les résultats présentés dans cette section sont significatifs, c'est à dire que nous avons réalisé autant de simulations nécéssaires à la convergence des données.

7.5.1 Taille des arbres/maillages

Comme nous l'avons déjà présenté dans la section 2.3, la loi de Chuang-Sirbu permet de connaître le nombre de nœuds nécessaires à la construction d'un arbre connectant N membres d'un groupe. Nous rappelons que cette relation est donnée par l'équation

$$L_m = \overline{L_u} N^k \tag{7.1}$$

dans laquelle L_m est le nombre de liens de l'arbre multicast (ou le nombre de nœuds moins 1), $\overline{L_u}$ est la distance moyenne dans la topologie, et N est le nombre de membres du groupe multicast. Le facteur k est appelé facteur de mise à l'échelle (scaling factor) et $k \in [0; 1]$. Nous rappelons que la valeur de k oscille généralement entre environ 0,7 et 0,8 pour des réseaux filaires de quelques milliers de routeurs.

Nous avons mesuré la taille T des arbres de diffusion multicast pour différentes valeurs de N et pour les deux architectures TUNNEL et CARRE. Nous avons uniquement considéré les nœuds du réseau ad hoc, et avons en particulier ignoré les branches d'interconnexion reliant les sous-arbres lorsque la continuité de préfixe est respectée. En effet, nous estimons que les routeurs interconnectant des réseaux dans l'Internet filaire ont en général des ressources suffisantes (énergie, bande-passante, mémoire) afin de supporter le routage et la retransmission multicast. Nous nous intéressons en particulier uniquement à l'impact de la continuité de préfixe sur les nœuds du réseau ad hoc. Ces résultats sont présentés par la figure 7.3.

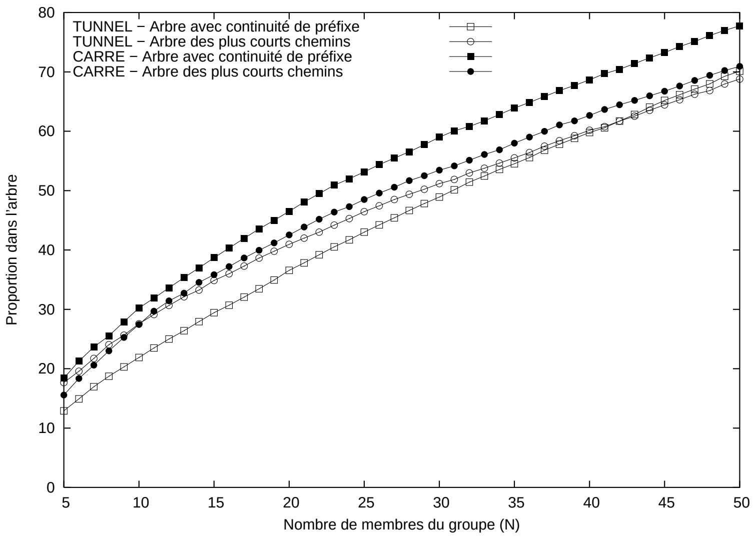

FIG. 7.3 – Taille moyenne des arbres de diffusion

On remarque dans un premier temps que la différence de taille entre les arbres avec ou sans continuité de préfixe est au maximum égale à 8 nœuds. De plus, on remarque que la topologie a une influence importante car avec l'architecture CARRE les arbres de plus courts chemins sont plus petits alors qu'avec l'archtecture TUNNEL on observe le contraire lorsque $N < 40$. Il est clair que le placement des passerelles dans ces architectures a également une influence importante. Nous insistons sur ce point afin de mettre en évidence l'importance de l'influence de la toplogie physique et des conditions de simulation. Néanmoins, on remarque que la taille T des arbres ne varie pas énormément et reste dans le même ordre de grandeur quelles que soient les conditions. Nous avons également observé que T suit une loi de puissance telle que

$$T = \alpha N^k \tag{7.2}$$

qui a une forme identique à la loi de Chuang-Sirbu. Nous avons calculé les valeurs de α et k permettant d'obtenir la meilleure approximation des données, et avons également calculé la valeur de k lorsque $\alpha = \overline{L_u}$ (c'est à dire avec la forme exacte de la loi de Chuang-Sirbu). Ces résultats sont présentés dans la table 7.1.

	Valeurs exactes		Avec $\alpha = \overline{L_u}$	
	α	k	$\overline{L_u}$	k
TUNNEL - arbres avec continuité de préfixe	4.2	0.72	6.26	0.61
TUNNEL - arbres des plus courts chemins	7.4	0.57	6.26	0.62
CARRE - arbres avec continuité de préfixe	7.9	0.59	5.41	0.69
CARRE - arbres des plus courts chemins	7.2	0.59	5.41	0.67

TAB. 7.1 – Paramètres de la loi de puissance

Lorsque $\alpha = \overline{L_u}$, on remarque que les valeurs de k sont très proches pour une architecture donnée (CARRE ou TUNNEL) et ce, que la continuité de préfixe soit respectée ou non. Bien que les équations obtenues avec $\alpha = \overline{L_u}$ ne soient pas celles permettant de représenter les données mesurées avec la meilleure précision, on observe que k est compris entre 0,61 et 0,69 c'est à dire dans un intervalle très proche de celui observé ([0,7 ;0,8])dans les réseaux filaires. Nous pensons que la valeur de k est inférieure car nous simulons un réseau de taille relativement faible (100 nœuds) comparé aux réseaux utilisés dans les études existantes qui considèrent des topologies de quelques milliers de routeurs. Or il a été montré que la taille de la topologie a une influence sur k [82], en particulier car la distance moyenne dans le graphe augmente ce qui diminue la probabilité de superposition des branches d'un arbre multicast. En fait on remarque qu'il semble équivalent de construire un arbre des plus courts chemins reliant N nœuds ou quatre arbres (un par sous-réseau) reliant (statistiquement parlant) $N/4$ nœuds. Pour les deux architectures considérées, la continuité de préfixe n'a donc pas une influence significative sur la taille des arbres de diffusion multicast.

7.5.2 Composition des arbres

Comme dans la section 2.3, nous avons également étudié la composition des arbres, c'est à dire la proportion de membres d'un groupe dans l'arbre de diffusion. Un arbre est en effet formé par les membres ayant adhéré au groupe mais aussi par les nœuds ayant été rajoutés dans l'arbre lors de la construction des branches. Nous avons donc calculé la proportion des nœuds membres dans les arbres multicast pour différentes tailles N de groupe. Ces résultats sont présentés par les figures 7.4 et 7.5. Comme dans les réseaux filaires, les nœuds membres sont minoritaires lorsque le nombre de membres est faible. Dans le cas de l'architecture CARRE, le point d'équilibre entre les proportions de nœuds membres et de nœuds non membres est atteint lorsque N est égal à 23 (arbres des plus courts chemins) et 29 (arbres avec continuité de préfixe). Avec l'architecture TUNNEL, ces valeurs sont égales à 14 (arbres avec continuité de préfixe) et 21 (arbres des plus courts chemins). La topologie a donc à nouveau une influence majeure puisque dans un cas (TUNNEL) l'équilibre est atteint plus rapidement avec la continuité de préfixe et dans l'autre cas (CARRE) les arbres de plus courts chemins permettent d'atteindre le point d'équilibre le plus rapidement. On note cependant que dans les deux cas la proportion s'inverse comme nous l'avions constaté avec des réseaux filaires. Cet effet est notamment directement lié à la loi de Chuang-Sirbu : le nombre total de membres d'un arbre a une croissance plus faible lorsque le nombre de membres du groupe augmente linéairement. La continuité de préfixe n'affecte donc pas ce phénomène avec les deux architectures considérées.

7.5.3 Profondeur des arbres

Le deuxième indicateur que nous utilisons pour étudier l'influence de la continuité de préfixe est la profondeur moyenne des arbres. Nous mesurons la distance (en sauts) entre chaque membre de l'arbre et la racine de celui-ci. Cet indicateur est important car il permet de connaître le nombre moyen de retransmissions nécessaires pour qu'un paquet atteigne chaque membre du groupe à partir de la racine. Dans un réseau ad hoc, chaque retransmission utilise des ressources (énergie, bande passante) et il est donc important de minimiser le nombre total de retransmissions. La figure 7.6 présente la profondeur moyenne des membres d'un arbre par rapport à la taille N des groupes multicast. On notera que l'on ne mesure pas la profondeur moyenne des membres du groupe multicast mais la profondeur des membres de l'arbre. En

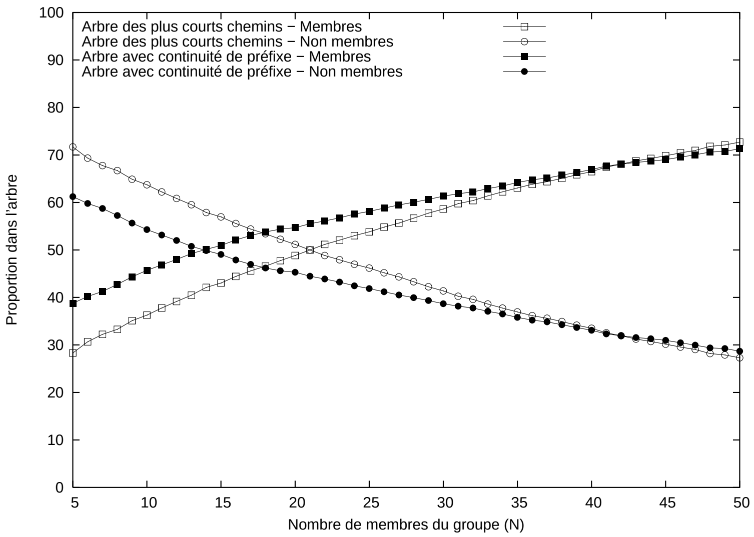

FIG. 7.4 – Proportions de nœuds membres et non membres pour l'architecture TUNNEL

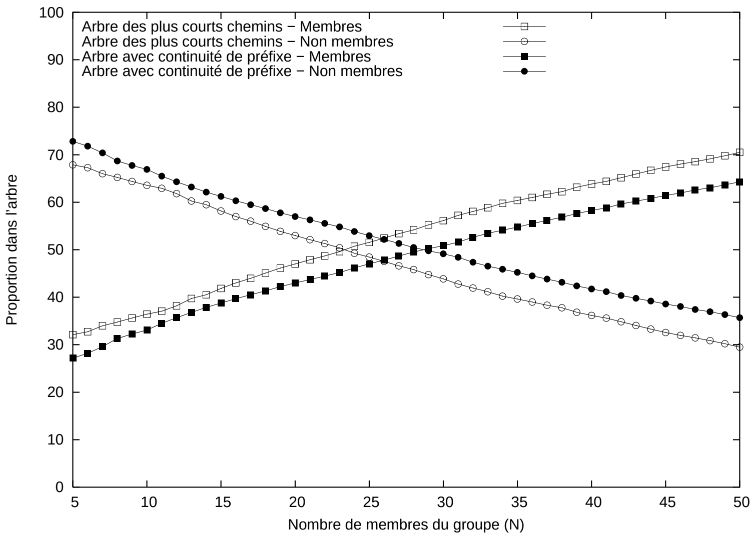

FIG. 7.5 – Proportions de nœuds membres et non membres pour l'architecture CARRE

effet, la seule mesure de la profondeur des membres n'est pas assez fine car elle ne permet pas par exemple de différencier le coût de transmission lorsque deux branches se chevauchent.

On remarque qu'avec l'architecture CARRE, la profondeur moyenne est supérieure lorsque la continuité de préfixe est respectée alors qu'elle est inférieure avec l'architecture TUNNEL. Cette constatation est cohérente avec la nature des architectures. La longueur moyenne $\overline{L_u}$ est en effet plus importante avec l'architecture TUNNEL qu'avec l'architecture CARRE (voir le tableau 7.1). En conséquence, les arbres de plus courts chemins sont plus profonds avec l'architecture TUNNEL. Au contraire, les arbres respectant la continuité de préfixe sont plus profonds avec l'architecture CARRE car les passerelles sont excentrées et placées dans les coins de cette architecture. Les branches entre les membres et les passerelles sont donc plus longues qu'avec l'architecture TUNNEL dans laquelle les passerelles sont au centre de chaque zone géographique (voir figure 7.2). L'influence de la topologie sous-jacente est donc primordiale.

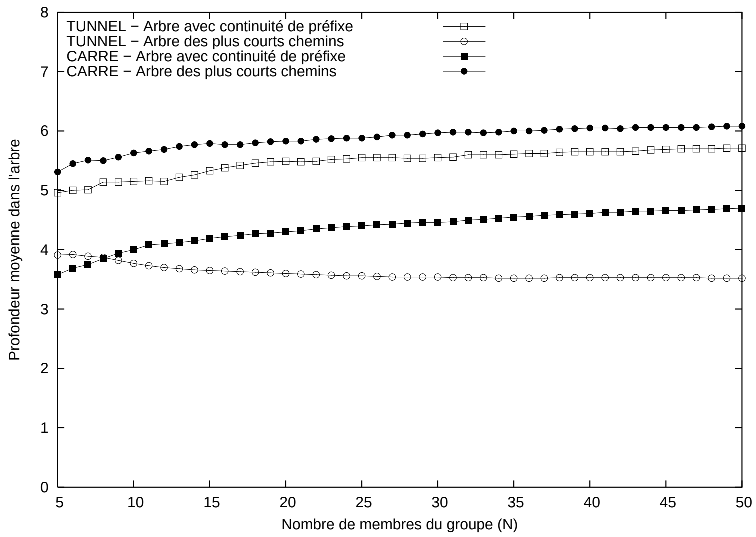

FIG. 7.6 – Profondeur moyenne des arbres de diffusion

Plus précisément, on constate que le placement des passerelles à une influence majeure sur la profondeur moyenne des arbres. Il est en effet toujours possible de trouver une position optimale des passerelles permettant de minimiser la profondeur des arbres. On remarque de plus que la profondeur moyenne n'est pas fortement influencée par la taille des groupes multicast. En pratique, si l'on veut construire des arbres multicast respectant la continuité de préfixe, il est donc primordial de placer les passerelles de façon optimale.

7.5.4 Bruit des arbres

Nous introduisons dans cette partie la notion de *bruit* d'un arbre. Nous définissons le bruit d'un membre d'un arbre multicast comme le nombre de voisins de ce nœud ne faisant pas partie de cet arbre. Lorsqu'un nœud membre d'un arbre multicast retransmet les données envoyées dans l'arbre, il empêche en particulier ses voisins n'appartenant pas à l'arbre d'émettre et de recevoir d'autres données. Or ces nœuds ne sont pas intéressés par les données envoyées à destination de ce groupe multicast : ils considèrent donc ce trafic comme du bruit. Cet indicateur nous permet donc d'étudier dans quelles mesures un arbre multicast est bruyant pour les nœuds n'en faisant pas partie. Cet indicateur est bien plus intéressant que par exemple le degré dans l'arbre. En effet, dans un réseau filaire on étudie généralement le degré des membres d'un arbre car il permet de savoir combien de retransmissions sont effectuées par chaque membre de l'arbre (un paquet par voisin dans l'arbre). Or dans un réseau ad hoc, un membre d'un arbre émet un unique paquet à destination de tous ses voisins dans l'arbre car le support radio est à diffusion. Pour les réseaux ad hoc il est donc plus intéressant de s'intéresser au bruit moyen d'un arbre plutôt qu'à son degré moyen. La mesure du bruit moyen des arbres multicast est ainsi présentée par la figure 7.7.

On voit sur cette figure que le bruit moyen d'un arbre est quasiment indépendant de la continuité de préfixe. En fait, le bruit des arbres de plus courts chemins est même plus important lorsque la taille des groupes est inférieur à environ 30 membres. Lorsque la continuité de préfixe est respectée, on a en effet constaté que chaque sous-arbre créé dans chacun des

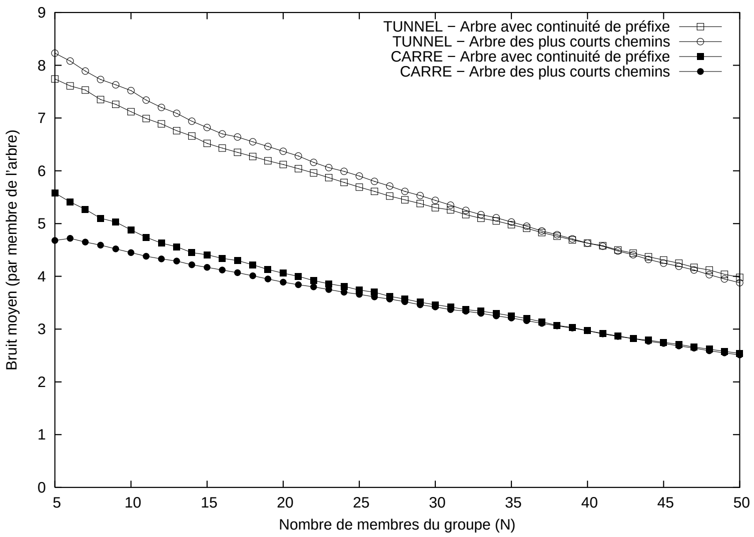

FIG. 7.7 – Bruit moyen des arbres de diffusion

sous-réseaux implique rapidement la majorité des nœuds qui sont proches de la passerelle du sous-réseau. Dans ces conditions, le bruit des nœuds proches d'une passerelle est quasi nul. Au contraire, les nœuds plus éloignés d'une passerelle ont un bruit assez élevé. Avec un arbre des plus courts chemins, le même phénomène se produit autour de la racine de l'arbre. Cependant, l'arbre est plus grand (comparé à chacun des quatre sous-arbres du cas précédent) et il y plus de nœuds distants de la racine : le bruit moyen est donc plus important. On note cependant que le bruit moyen d'un arbre multicast est faiblement influencé par la continuité de préfixe.

7.5.5 Concentration des états multicast

Dans cette dernière partie, nous étudions les effets de la superposition d'arbres multicast. En particulier, nous nous intéressons au nombre d'états multicast dans chaque nœud du réseau ad hoc. Ce nombre représente le nombre d'arbres auxquels appartient un nœud. Il permet notamment de mesurer la surcharge de chaque nœud : un nœud faisant partie d'un grand nombre d'arbres devra potentiellement propager un grand volume de données, causera en conséquence beaucoup de bruit, et utilisera d'autant plus rapidement sa réserve d'énergie. Nous avons donc mesuré le nombre d'états multicast présents dans chaque nœud du réseau ad hoc lorsque nous construisons entre 1 et 50 arbres multicast. Les membres de chaque groupe sont choisis aléatoirement et uniformément parmi les 100 nœuds mobiles du réseau. On construit alors les arbres et on mesure le nombre d'états de chaque nœud du réseau. Les résultats de ces mesures sont présentés par les figures 7.8 et 7.9. Les données sont représentées sous la forme d'une fonction cumulative inverse. On présente ainsi le pourcentage de nœuds ayant plus de S états de propagation lorsqu'on simule la construction de G arbres multicast.

On remarque principalement que la continuité de préfixe a une faible influence sur la concentration des états multicast dans les nœuds du réseau ad hoc. Lorsqu'il y a beaucoup de groupe ($G = 50$), on constate néanmoins que le nombre moyen d'états est supérieur lorsque la continuité de préfixe est respectée. En pratique, les nœuds proches des passerelles sont en effet grandement sollicités car chaque sous-arbre de chacun des sous-réseaux est enraciné à la passerelle du sous-réseau. Les nœuds proches des passerelles font donc partie de la majorité des arbres. Au contraire, avec les arbres de plus courts chemins, les racines des différents groupes ne sont pas forcément identiques et les membres des groupes sont également différents. Les membres des arbres sont donc *éparpillés* dans le réseau et la superposition des arbres a une influence plus faible. Cependant, la distribution géographique obtenue avec le modèle de mobilité RWP ([137]) conduit à ce qu'une majorité des branches traversent le centre de la topologie. Les nœuds situés autour de cet endroit ont donc une probabilité très élevée de faire partie d'un grand nombre d'arbres multicast. Ces deux phénomènes, d'un côté la concentration autour des passerelles et de l'autre la concentration autour du centre de la topologie, ont donc une influence quasi équivalente et les mesures restent donc très proches.

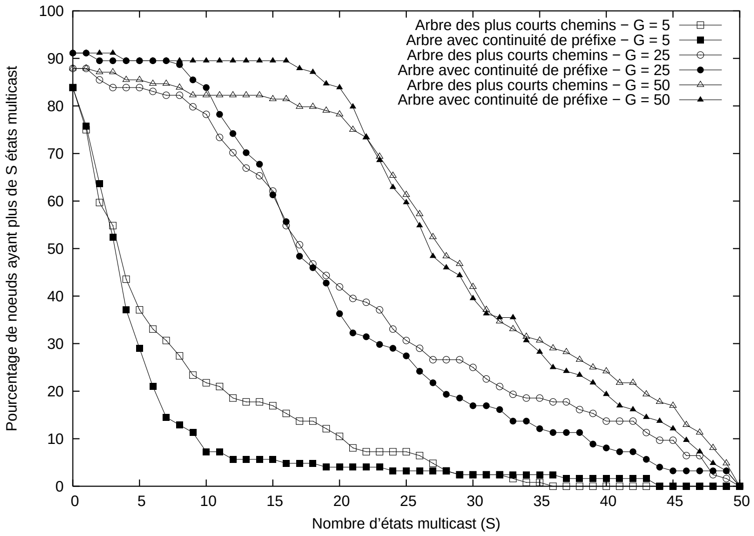

Fig. 7.8 – Concentration d'états avec l'architecture TUNNEL

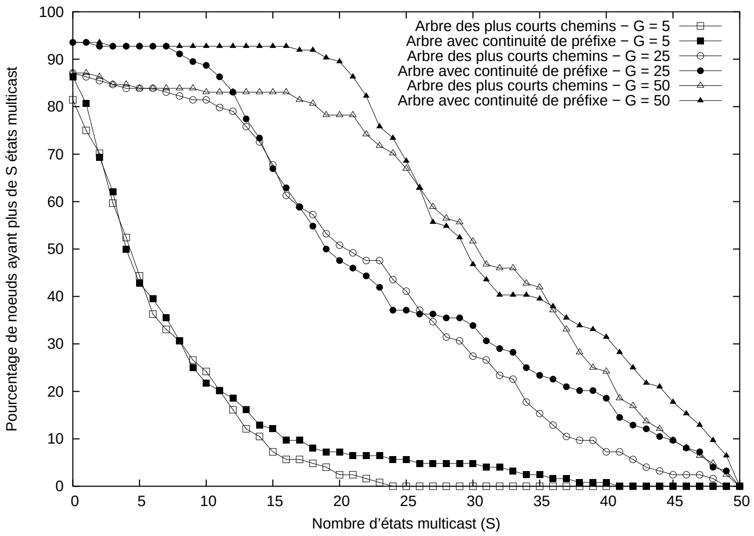

Fig. 7.9 – Concentration d'états avec l'architecture CARRE

De plus, la loi de Chuang-Sirbu nous permet d'estimer le nombre de membres d'un arbre multicast. Or chaque membre de l'arbre va avoir un état de propagation : la loi de Chuang-Sirbu nous indique donc également combien d'états multicast vont être répartis dans le réseau. Etant donné qu'on construit G groupes de taille N, on aura donc à *distribuer* $G * \overline{L_u} N^k$ états multicast dans le réseau. On a de plus montré dans la section 7.5.1 que la loi de Chuang-Sirbu est applicable que les arbres respectent ou non la continuité de préfixe. On a donc généralement un nombre quasi équivalent d'états de propagation à distribuer dans le réseau ad hoc. Etant donné les phénomènes expliqués dans le paragraphe précédent (concentration autour des passerelles ou autour du centre de la topologie), on assiste à une concentration équivalente des états multicast dans le réseau mais à des endroits différents.

7.6 Conclusion

Dans ce chapitre, nous avons appliqué le concept de continuité de préfixe aux communications multicast. Nous avons en particulier évalué l'impact de la continuité de préfixe sur la forme des structures de diffusion multicast. Nous avons ainsi montré que ces structures (arbres ou maillages) ont des caractéristiques similaires avec celles obtenues lorsque la continuité de préfixe n'est pas respectée. Plus précisémement, la loi de Chuang-Sirbu sur la taille des arbres peut être appliquée aux réseaux ad hoc que la continuité de préfixe soit ou non respectée. De plus, le bruit généré par ces structures est similaire dans les deux cas de figure. Cependant, nous avons constaté que les topologies de réseau utilisées ont une influence majeure, et que le placement des passerelles dans le réseau ad hoc a un impact prédominant sur la forme et les caractéristiques des structures de diffusion. Nous avons en particulier mis en évidence ce phénomène en étudiant la profondeur des arbres, qui dépend fortement de la topologie physique du réseau. Le routage multicast respectant la continuité de préfixe est donc une alternative intéressante au routage utilisant les plus courts chemins.

Il est en effet possible de réaliser des opérations de contrôle, de gestion, et de filtrage au niveau des passerelles de chaque sous-réseau. De plus chaque passerelle peut effectuer un transcodage des données afin de réduire la consommation des ressources (bande-passante, énergie) des nœuds de son sous-réseau et afin d'adapter le format du flux multicast à la puissance de calcul des stations mobiles. Enfin le transit de données multicast entre sous-réseaux est interdit. On n'évite ainsi d'utiliser les ressources des nœuds d'un sous-réseau pour effectuer un transit de données entre sous-réseaux. Dans ce cas l'infrastructure de routage multicast du réseau filaire est mise à contribution afin de faire transiter les données entre des sous-réseaux ad hoc.

Conclusion

Dans ce mémoire, nous avons présenté la problématique de la mobilité des participants d'une communication de groupe. En ce qui concerne les réseaux sans-fil basés sur une infrastructure, l'étude des propositions existantes pour le support de la mobilité au sein de communications de groupe nous a permis de mettre en évidence les difficultés et les particularités liées à cette intégration.

Une première conclusion est qu'il est difficile de concilier vitesse de réaction et routage optimal : chaque solution propose en effet un compromis entre ces deux contraintes. Nous avons cependant montré qu'il est possible de proposer des solutions visant à permettre de réagir rapidement face aux déplacements des participants mobiles tout en proposant un routage optimal après un temps de convergence. Un récepteur mobile peut ainsi adhérer à un groupe depuis son réseau visité tout en profitant de la réactivité offerte par l'utilisation d'un tunnel via son agent mère. Afin de garantir cette dernière, nous avons proposé d'utiliser une extension du protocole MLD permettant à un nœud de demander à son agent mère de maintenir l'adhésion à un groupe multicast sans pour autant retransmettre les données. Après un déplacement du récepteur mobile, la retransmission des données via l'agent mère sera donc très rapide car on supprime le délai supplémentaire qui serait induit par une ré-adhésion de l'agent mère au groupe multicast considéré.

Nous avons étendu ce concept à la mobilité d'une source lorsque celle-ci est la racine d'un arbre multicast. Nous avons en effet proposé un protocole qui permet de minimiser le temps d'interruption de la diffusion des données tout en permettant leur routage optimal après un temps de convergence. Nous avons montré qu'il est ainsi possible de diminuer de manière significative la longueur des routes empruntées par les données envoyées par une source mobile. Notre proposition est de plus distribuée et ne nécessite aucune synchonisation puisque chaque récepteur adhère de manière indépendante au nouveau canal de diffusion enraciné à la position courante de la source. Notre solution supporte également les déplacements multiples d'une source mobile.

Nous avons ensuite étendu notre étude aux réseaux ad hoc, en considérant tout particulièrement les réseaux ad hoc hybrides qui ont une interconnexion vers l'Internet. Contrairement à ce que l'on observe dans les réseaux sans-fil basés sur une infrastructure, avec les réseaux ad hoc la mobilité des nœuds est au cœur des protocoles de routage multicast : c'est en effet l'une des contraintes principales considérées lors du développement de ces protocoles. Au cours de notre étude, nous avons dans un premier temps introduit le concept de continuité de préfixe dans un réseau ad hoc. Ce concept conduit à la création de sous-réseaux IP connexes : chaque sous-réseau forme un graphe connexe dont tous les nœuds ainsi que leur passerelle partagent un même préfixe réseau. La continuité de préfixe permet d'éviter l'utilisation d'en-têtes de routage car elle impose une organisation logique dans le réseau ad hoc. Nous avons évalué notre proposition en comparant notamment les différents algorithmes que nous proposons pour sélectionner un voisin-père. Nous avons ainsi montré qu'il est possible de proposer des algorithmes permettant à un nœud de conserver son préfixe courant aussi longtemps que possible.

Enfin, nous avons appliqué le concept de continuité de préfixe aux communications de groupe. Nous avons pour cela proposé des règles qui, appliquées à un protocole de routage multicast, permettent de créer des structures de diffusion multicast dont tous les membres partagent un même préfixe réseau. Il est ainsi possible d'interdire par exemple le transit des données entre deux sous-réseaux via un troisième, et ce afin de minimiser l'usage de ressources (bande-passante, énergie) dans le réseau ad hoc. La continuité de préfixe permet également d'envisager l'utilisation d'outils de management, par exemple pour réaliser des opérations de facturation. Nous avons étudié l'impact de la continuité de préfixe sur les structures de diffusion multicast en comparant les arbres respectant la continuité de préfixe avec des arbres de plus courts chemins. Nous avons ainsi montré que les deux types d'arbres ont des caractéristiques topologiques similaires, et qu'ils respectent notamment la loi de Chuang-Sirbu qui modélise la taille des arbres multicast. Cette étude permet en particulier d'envisager l'application de la continuité de préfixe aux communications de groupe dans les réseaux ad hoc.

Perspectives

Les travaux que nous avons réalisés nous ont permis de mieux comprendre les implications de la mobilité dans les communications multicast. A partir de ces constatations, les perspectives de travaux futurs sont nombreuses.

Dans un premier temps, on peut s'interroger sur la possibilité d'intégrer le concept de mobilité dans les protocoles de routage multicast et dans les routeurs de l'Internet. La gestion de la mobilité des participants d'une communication de groupe deviendrait ainsi une fonctionnalité inhérente au protocole de routage multicast. L'architecture LAR (voir section 1.4.5) est un exemple de protocole intégrant la mobilité en utilisant une couche supplémentaire d'adressage logique. Il serait intéressant d'étudier comment l'association entre l'adresse courante d'un nœud mobile et son identificateur unique pourrait être intégrée au cœur des protocoles de routage de l'Internet, ou même au sein du protocole IP. La mobilité pourrait ainsi être gérée par le réseau et non pas par les entités communicantes. Les applications futures sont nombreuses puisque les réseaux de téléphonie mobile de troisième et de quatrième générations fourniront à n'en pas douter de nombreux services multimédia comme la télévision et la radio par Internet.

L'autoconfiguration d'adresses dans les réseaux ad hoc hybrides ouvre également de nouveaux horizons. Dans un premier temps, il serait intéressant d'étudier l'applicabilité du concept de continuité de préfixe aux réseaux ad hoc utilisant la version 4 du protocole IP. Avec cette version il n'est en effet pas possible d'utiliser les identifiants Ethernet uniques (EUI-64) afin de créer les adresses à partir d'un préfixe réseau. La technique de propagation des informations envoyées par les passerelles que nous proposons permet cependant d'envisager l'utilisation de méthodes de création d'adresses basées sur un système d'attribution d'identifiants. Les nœuds participant à la propagation de l'information envoyée par une passerelle données forment en effet un arbre : il est donc possible d'utiliser des techniques de numérotation des nœuds de l'arbre afin de leur assigner de manière dynamique un identifiant unique qui pourra être utilisé pour créer une adresse IPv4 non dupliquée.

De plus, il serait intéressant d'étudier l'impact de la mobilité des passerelles. Nous avons en effet uniquement considéré des passerelles fixes au milieu d'un réseau ad hoc dont tous les nœuds sont mobiles. La mobilité des passerelles aura à n'en pas douter un impact très important sur l'organisation dynamique en sous-réseaux IP. D'autre part, la mobilité des passerelles aura de plus un impact sur les structures de diffusion multicast respectant la continuité de préfixe. Il serait intéressant d'étudier les variations de la topologie d'une telle structure et d'analyser en particulier l'appartenance dynamique des nœuds aux arbres de diffusion. Nous pourrions également considérer l'autoconfiguration d'adresses lorsque les nœuds du réseau ad hoc ont plusieurs interfaces utilisant ou non des technologies de communication différentes : un nœud pourraît ainsi appartenir à plusieurs sous-réseaux afin d'augmenter sa connectivité vers l'Internet.

Nous aimerions de plus étudier les interactions possibles entre la continuité de préfixe, la diffusion multicast, et les protocoles de gestion de réseaux. Nous pensons que l'organisation logique apportée par la continuité de préfixe favorise le déploiement de protocoles de supervision et de management de réseaux. En effet, puisqu'il existe une certaine organisation du réseau ad hoc en sous-réseaux contigus, il devient possible de réaliser des opérations de gestion de réseaux au sein d'un sous-réseau sans pour autant utiliser les ressources des machines d'un autre sous-réseau. Un opérateur peut également, via sa passerelle, proposer des services à ses utilisateurs. On peut par exemple imaginer un système de réservation de bande-passante au sein d'un sous-réseau : les nœuds mis à contribution au cours du processus de réservation pourraient obtenir une *rémunération* sous la forme de temps de connexion vers l'Internet ou d'accès à des services tels que la diffusion multicast. La collaboration est au cœur des réseaux ad hoc et nous pensons qu'il serait possible de l'encourager en mettant en place un système de rémunération : chaque service rendu permet d'obtenir des points qui permettent à leur tour d'obtenir des services. Nous pensons que la continuité de préfixe et l'association à une passerelle permettrait de renforcer la mise en place et l'efficacité d'un tel système.

A plus long terme, il serait également pertinent de pouvoir étudier l'influence de la mobilité sur les communications de groupe en utilisant des modèles de mobilité et des contextes d'utilisation plus réalistes. Ces informations nous permettraient en effet d'évaluer nos propositions dans le cadre d'un projet de déploiement concret. Il est cependant diffile d'obtenir des opérateurs de telles informations. Il est donc préférable de s'orienter vers des projets de partenariat dans le déploiement de réseaux sans-fil. L'Université Louis Pasteur déploie ainsi actuellement un réseau sans-fil qui sera composé à termes de plusieurs centaines de points d'accès. Il serait alors possible d'étudier de manière systématique et anonyme la mobilité des utilisateurs afin d'obtenir des modèles de déplacements. Il serait également possible de proposer un service de diffusion multicast attrayant qui permettraient d'étudier à moyenne échelle l'influence de la mobilité sur la structure des arbres de diffusion multicast. On pourrait par exemple proposer un service de diffusion de programmes musicaux afin d'attirer un maximum d'utilisateurs.

Bibliographie

[1] Data Processing - Open Systems Interconnection - Basic Reference Model, 1984. ISO IS 7498.

[2] J. Postel. RFC-791 - Internet Protocol, September 1981.

[3] J. Postel. RFC-793 - Transmission Control Protocol, September 1981.

[4] J. Postel. RFC-768 - User Datagram Protocol, August 198.

[5] C. Zaccone, Y. T'Joens, and B. Sales. Address reuse in the Internet, adjourning or suspending the adoption of IP next generation ? In *Proceedings of Icon'00*, September 2000. Singapore.

[6] P. Grossetete, J. Bound, and T. Hain. Rapport de la North American IPv6 Task Force, September 2003.

[7] T. Bu, L. Gao, and D. Townsley. On characterizing BGP routing table growth. In *Proceedings of IEEE Globecom'02*, volume 3, pages 2185–2189, November 2002. Taipei, Taiwan.

[8] G. Cizault. *IPv6 - 3ème édition*. Editions O'Reilly, 2002.

[9] S. Deering and R. Hinden. RFC-1883 - Internet Protocol, Version 6 (IPv6) Specification (Obsoleted by RFC-2460), December 1995.

[10] S. Deering and R. Hinden. RFC-2460 - Internet Protocol, Version 6 (IPv6) Specification, December 1998.

[11] P. Gross and P. Almquist. RFC-1380 - IESG Deliberations on Routing and Addressing, November 1992.

[12] R. Hinden. RFC-2450 - Proposed TLA and NLA Assignment Rules, December 1998.

[13] IEEE Standard - Guidelines for 64-bit Global Identifier (EUI-64) Registration Authority, March 1997.

[14] T. Narten, E. Nordmark, and W. Simpson. RFC-2461 - Neighbor Discovery for IP Version 6 (IPv6), December 1998.

[15] M. Mackay, C. Edwards, C. Dunmore, T. Chown, and G. Carvalho. A scenario-based review of IPv6 transition tools. *IEEE Internet Computing*, 7(3) :27–35, May-June 2003.

[16] D. Waddington and F. Chang. Realizing the transition to IPv6. *IEEE Communications Magazine*, 40(6) :138–147, June 2002.

[17] S. Deering. RFC-1112 - Host Extensions for IP Multicasting, August 1989.

[18] R. Stevens. *TCP/IP illustré - Les Protocoles (vol. 1)*. International Thomson Publishing, 1994.

[19] S. Deering, W. Fenner, and B. Haberman. RFC-2710 - Multicast Listener Discovery (MLD) for IPv6, October 1999.

[20] R. Vida and L. Costa (editors). RFC-3810 - Multicast Listener Discovery Version 2 (MLDv2) for IPv6, June 2004.

[21] D. Waitzman, C. Partridge, and S. Deering. RFC-1075 - Distance Vector Multicast Routing Protocol, November 1988.

[22] A. Adams, J. Nicholas, and W. Siadak. Internet Draft - Protocol Independent Multicast - Dense Mode (PIM-DM) : Protocol Specification (Revised), draft-ietf-pim-dm-new-v2-05.txt, June 2004.

[23] J. Moy. Multicast Routing Extensions for OSPF. *Communications of the ACM*, 37(8) :61–66, August 1994.

[24] J. Moy. RFC-2328 - OSPF Version 2, April 1998.

[25] A. Ballardie, P. Francis, and J. Crowcroft. Core Based Trees (CBT) : An Architecture for Scalable Inter-domain Multicast Routing. In *Proceedings of ACM Sigcomm'93*, pages 85–95, September 1993. San Fransisco, CA, USA.

[26] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C.-G. Liu, and L. Wei. The PIM Architecture for Wide-Area Multicast Routing. *IEEE/ACM Transactions on Networking*, 4(2) :153–162, April 1996.

[27] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei. RFC-2362 - Protocol Independent Multicast - Sparse Mode (PIM-SM) : Protocol Specification, June 1998.

[28] D. Meyer and W. Fenner (editors). RFC-3618 - The Multicast Source Discovery Protocol (MSDP), October 2003.

[29] H. Holbrook and D. Cheriton. IP Multicast Channels : EXPRESS Support for Large-scale Single-Source Applications. In *Proceedings of ACM Sigcomm'99*, September 1999. Cambridge, MA, USA.

[30] H. Holbrook and B. Cain. Internet Draft - Source-Specific Multicast for IP, draft-ietf-ssm-arch-06.txt, September 2004.

[31] L. Sahasrabuddhe and B. Mukherjee. Multicast routing algorithms and protocols : a tutorial. *IEEE Network*, 14(1) :90–102, Jan-Feb 2000.

[32] K. Almeroth. The Evolution of Multicast : From the MBone to Inter-Domain Multicast to Internet2 Deployment. *IEEE Network*, 14(1) :10–20, Jan-Feb 2000.

[33] K. Almeroth. A Long-Term Analysis of Growth and Usage Patterns in the Multicast Backbone (MBone). In *Proceedings of IEEE Infocom'00*, volume 2, pages 824–833, March 2000. Tel Aviv, Israel.

[34] ITU-T Recommendation G.992.1 - Asymmetric Digital Subscriber Line (ADSL) transceivers, July 1999.

[35] ANSI T1.413 - Asymmetric Digital Subscriber Line (ADSL) Metallic Interface, 1998.

[36] IEEE Std 802 Part 11 : Wireless Lan Medium Access Control (MAC) And Physical Layer (PHY) Specifications, September 1999.

[37] Specification of the Bluetooth System. Core, Version 1.0 B, December 1999.

[38] J. Lansford, A. Stephens, and R. Nevo. Wi-Fi (802.11b) and Bluetooth : Enabling Coexistence. *IEEE Network*, 15(5) :20–27, Sept-Oct 2001.

[39] C. Perkins. RFC-2002 - IP Mobility Support (Obsoleted by RFC-3220 and RFC-3344), October 1996.

[40] C. Perkins (editor). RFC-3344 - IP Mobility Support for IPv4, August 2002.

[41] D. Johnson, C. Perkins, and J. Arkko. RFC-3775 - Mobility Support in IPv6, June 2004.

[42] S. Thomson and T. Narten. RFC-2462 - IPv6 Stateless Address Autoconfiguration, December 1998.

[43] P. Albitz and L. Cricket. *DNS and BIND*. Editions O'Reilly, 1997.

[44] F. Teraoka, K. Uehara, H. Sunahara, and J. Murai. VIP : A Protocol Providing Host Mobility. *Communications of the ACM*, 37(8) :67–75, August 1994.

[45] M. Ishiyama, M. Kunishi, K. Uehara, H. Esaki, and F. Teraoka. LINA : A New Approach to Mobility Support in Wide Area Networks. *IEICE Transactions on Communications*, E-84-B(8) :2076–2086, August 2001.

[46] M. Kunishi, M. Ishiyama, K. Uehara, H. Esaki, and F. Teraoka. LIN6 : A New Approach to Mobility Support in IPv6. In *Proceedings of WPMC'00*, November 2000. Bangkok, Thailand.

[47] F. Teraoka, M. Ishiyama, and M. Kunishi. Internet Draft - LIN6 : A Solution to Multihoming and Mobility in IPv6, draft-teraoka-multi6-lin6-00.txt, December 2003.

[48] J.-J. Pansiot, D. Grad, and S. Marc-Zwecker. Towards a Logical Addressing and Routing Sublayer. In *Proceedings of PROMS'95*, pages 438–451, October 1995. Salzbourg, Austria.

[49] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet Indirection Infrastructure. In *Proceedings of Sigcomm'02*, pages 73–86, August 2002. Pittsburgh, PA, USA.

[50] G. Xylomenos and G. Polyzos. IP Multicast for Mobile Hosts. *IEEE Communications Magazine*, 35(1) :54–58, January 1997.

[51] C. Bettstetter, A. Riedl, and G. Gessler. Interoperation of Mobile IPv6 and Protocol Independent Multicast Dense Mode. In *Proceedings of Wksp. WL Net. Mob. Comp.*, August 2000. Toronto, Canada.

[52] C. Jelger and T. Noel. Multicast for Mobile Hosts in IP Networks : Progress and Challenges. *IEEE Wireless Communications Magazine*, 9(5) :58–64, October 2002.

[53] I. Romdhani, M. Kellil, H.-Y. Lach, A. Bouabdallah, and H. Bettahar. Mobility-Aware RendezVous Point for Mobile Multicast Sources. In *Proceedings of WWIC*, pages 62–73, February 2004. Frankfurt, Germany.

[54] T. Harrison, C. Williamson, W. Mackrell, and R. Bunt. Mobile Multicast (MoM) Protocol : Multicast Support for Mobile Hosts. In *Proceedings of ACM Mobicom'97*, pages 151–160, September 1997. Budapest, Hungary.

[55] C. Williamson, T. Harrison, W. Mackrell, and R. Bunt. Performance Evaluation of the MoM Mobile Multicast Protocol. *ACM Mobile Networks and Applications*, 3(2) :189–201, 1998.

[56] C. Lin and K. Wang. Mobile Multicast Support in IP Networks. In *Proceedings of IEEE Infocom'00*, pages 1664–1672, March 2000. Tel Aviv, Israel.

[57] Y. Suh, H. Shin, and D. Kwon. An Efficient Multicast Routing Protocol in Wireless Mobile Networks. *Wireless Networks*, 7(5) :443–453, September 2001.

[58] C.-C. Tseng, K.-H. Chi, and T.-L. Huang. A new locality-based IP multicasting scheme for mobile hosts. *Computer Communications*, 24(5-6) :486–495, March 2001.

[59] B. Cain, S. Deering, I. Kouvelas, B. Fenner, and A. Thyagarajan. RFC-3376 - Internet Group Management Protocol, Version 3, October 2002.

[60] Y.-J. Suh, H.-S. Shin, and D.-H. Kwon. An Efficient Multicast Routing Protocol in Wireless Mobile Networks. *ACM Wireless Networks*, 7(5) :443–453, September 2001.

[61] T. Noel, D. Grad, and J.-J. Pansiot. Logical Framework for Group Communications with Mobile Hosts. In *Proceedings of AFRICOM-CCDC'98*, October 1998. Tunis, Tunisia.

[62] D. Grad, J.-J. Pansiot, and S. Marc-Zwecker. Distributed Computation of Reduced Multicast Trees. In *Proceedings of TDP'96*, pages 91–107, June 1996. Lalonde les Maures, France.

[63] C. Diot, B. Levine, B. Liles, H. Kassem, and D. Balensiefen. Deployment Issues for the IP Multicast Service and Architecture. *IEEE Networks*, 14(1) :78–88, January 2000.

[64] I. Stoica, T. Eugene, and H. Zhang. REUNITE : A Recursive Unicast Approach to Multicast. In *Proceedings of Infocom'00*, pages 1644–1653, March 2000. Tel Aviv, Israel.

[65] L. Costa, S. Fdida, and O. Duarte. Hop-by-hop Multicast Routing. In *Proceedings of Sigcomm'01*, pages 249–259, August 2001. San Diego, CA, USA.

[66] R. Boivie, N. Feldman, Y. Imai, W. Livens, D. Ooms, and O. Paridaens. Internet Draft - Explicit Multicast (Xcast) Basic Specification, draft-ooms-xcast-basic-spec-05.txt, August 2003.

[67] R. Vida, L. Costa, and S. Fdida. Mobile hop-by-hop Multicast Routing. *Computer Networks*, 44(6) :789–812, April 2004.

[68] K.-T. Kim, J.-H. Kim, D.-K. Kim, and J. Lee. Efficient Explicit Mobile Multicast Routing Protocol (E2MMP). In *Proceedings of ICITA'02*, pages 176–181, November 2002. Bathurst, Australia.

[69] Y. Imai, H. Kishimoto, M.-K. Shin, and Y.-H. Kim. XCAST6 : eXplicit Multicast on IPv6. In *Proceedings of IEEE/IPSJ SAINT'03*, January 2003. Orlando, USA.

[70] S. Floyd and V. Paxson. Difficulties in Simulating the Internet. *IEEE/ACM Transactions on Networking*, 9(4) :392–403, August 2001.

[71] K. Fall and S. Floyd. Simulation-based Comparisons of Tahoe, Reno, and Sack TCP. *ACM Computer Communication Review*, 29(3) :5–21, July 1996.

[72] W. Leland, M. Taqqu, W. Willinger, and D. Wilson. On the Self-Similar Nature of Ethernet Traffic (extended version). *IEEE/ACM Transactions on Networking*, 2(1) :1–15, February 1994.

[73] B. Mah. An Empirical Model of HTTP Network Traffic. In *Proceedings of IEEE Infocom'97*, pages 592–600, August 1997. Kobe, Japan.

[74] K. Pawlikowski, H. Jeong, and J. Lee. On Credibility of Simulation Studies of Telecommunications Networks. *IEEE Communications Magazine*, 40(1) :132–139, January 2002.

[75] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On Power-Law Relationship of the Internet Topology. In *Proceedings of ACM Sigcomm'99*, August 1999. Cambridge, MA, USA.

[76] A. Medina, I. Matta, and J. Byers. On the Origin of Power Laws in Internet Topologies. *ACM Computer Communication Review*, 30(2) :18–28, April 2000.

[77] K. Calvert, M. Doar, and E. Zegura. Modeling Internet Topology. *IEEE Communications Magazine*, 35(6) :160–163, June 1997.

[78] D. Magoni and J.-J. Pansiot. Evaluation of Internet Topology Generators by Power Law and Distance Indicators. In *Proceedings of IEEE Icon'02*, pages 401–406, August 2002. Singapore.

[79] J. Chuang and M. Sirbu. Pricing Multicast Communications : A Cost Based Approach. In *Proceedings of INET'98*, July 1998. Geneva, Switzerland.

[80] G. Phillips, S. Shenker, and H. Tangmurarunkit. Scaling of Multicast Trees : Comments on the Chuang-Sirbu Scaling Law. In *Proceedings of ACM Sigcomm'99*, August 1999. Cambridge, MA, USA.

[81] R. Chalmers and K. Almeroth. On the Topology of Multicast Trees. *IEEE/ACM Transactions on Networking*, 11(1) :153–165, February 2003.

[82] P. Van Mieghem, G. Hooghiemstra, and R. Van Der Hofstad. On the Efficiency of Multicast. *IEEE/ACM Transactions on Networking*, 9(6) :719–732, December 2001.

[83] The Network Simulator - NS-2 - http ://www.isi.edu/nsnam/ns/.

[84] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Ya Xu, and Haobo Yu. Advances in Network Simulation. *IEEE Computer*, 33(5) :59–67, May 2000.

[85] J.-J. Pansiot and D. Grad. On Routes and Multicast Trees in the Internet. *ACM Computer Communication Review*, 28(1) :41–50, January 1998.

[86] B. Fenner, H. He, B. Haberman, and H. Sandick. Internet Draft - IGMP/MLD-based Multicast Forwarding ("IGMP/MLD Proxying"), draft-ietf-magma-igmp-proxy-06.txt, April 2004.

[87] C. Jelger and T. Noel. Multipoint et Mobilité : Optimisations pour IPv6. In *Proceedings of Algotel'02*, pages 109–113, May 2002. Mèze, France.

[88] R. Koodli (editor). Internet Draft - Fast Handovers for Mobile IPv6 - draft-ietf-mipshop-fast-mipv6-01.txt, January 2004.

[89] C. Jelger and T. Noel. An Analysis of Multicast Delivery with Mobile Receivers. In *Proceedings of IEEE PIMRC'03*, volume 3, pages 2690–2695, September 2003. Beijing, China.

[90] C. Jelger and T. Noel. Supporting Mobile SSM Sources for IPv6. In *Proceedings of IEEE Globecom'02*, volume 2, pages 1693–1697, November 2002. Taipei, Taiwan.

[91] M. Handley, C. Perkins, and E. Whelan. RFC-2974 - Session Announcement Protocol, October 2000.

[92] M. Handley and V. Jacobson. RFC-2327 - Session Description Protocol, April 1998.

[93] B. Fenner, B. Haberman, H. Holbrook, and I. Kouvelas. Internet Draft - Multicast Source Notification of Interest Protocol (MSNIP) - draft-ietf-magma-msnip-05.txt, September 2004.

[94] A. Tanenbaum. *Réseaux (4ème édition)*. Pearson Education, 2003.

[95] G. Malkin. RFC-2453 - RIP Version 2, November 1998.

[96] S. Das, R. Castaneda, J. Yan, and R. Sengupta. Comparative Performance Evaluation of Routing Protocols for Mobile, Ad Hoc Networks. In *Proceedings of IC3N'98*, pages 153–161, October 1998. Lafayette, USA.

[97] S. Corson and J. Macker. RFC-2501 - Mobile Ad hoc Networking (MANET) : Routing Protocol Performance Issues and Evaluation Considerations, January 1999.

[98] E. Royer and C.-K. Toh. A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks. *IEEE Personal Communications*, 6(2) :46–55, April 1999.

[99] C. Perkins and E. Royer. Ad-hoc On-Demand Distance Vector Routing. In *Proceedings of the 2nd IEEE Worshop on Mobile Computing Systems and Applications*, pages 90–100, February 1999. New Orleans, USA.

[100] D. Johnson. Routing in Ad Hoc Networks of Mobile Hosts. In *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, pages 158–163, December 1994. Santa Cruz, USA.

[101] P. Jacquet, P. Muhletaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot. Optimized Link State Routing Protocol. In *Proceedings of IEEE INMIC'01*, pages 62–68, December 2001. Lahore, Pakistan.

[102] B. Bellur and R. Ogier. A reliable, Efficient Topology Broadcast Protocol for Dynamic Networks. In *Proceedings of Infocom'99*, pages 178–186, March 1999. New York, USA.

[103] D. Johnson, D. Maltz, and Y.-C. Hu. Internet Draft - The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR), draft-ietf-manet-dsr-09.txt, April 2003.

[104] C. Perkins, E. Belding-Royer, and S. Das. RFC-3561 - Ad hoc On-Demand Distance Vector (AODV) Routing, July 2003.

[105] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *Proceedings of Sigcomm'94*, pages 234–244, August 1994. London, UK.

[106] T. Clausen and P. Jacquet. RFC-3626 - Optimized Link State Routing Protocol (OLSR), October 2003.

[107] ETSI STC-RES10 Committee. Radio Equipment and Systems : High Performance Radio Local Area Network (HYPERLAN) Type 1, Functional Specifications, June 1996.

[108] A. Qayyum. *Analysis and Evaluation of Channel Access Schemes and Routing Protocols for Wireless Networks*, Thèse de doctorat en Informatique, Université Paris-Sud XI, 2000.

[109] R. Ogier, F. Templin, and M. Lewis. RFC-3684 - Topology Dissemination Based on Reverse-Path Forwarding (TBRPF), February 2004.

[110] S. Das, R. Castaneda, and J. Yan. Simulation-based Performance Evaluation of Routing Protocols for Mobile Ad Hoc Networks. *Mobile Networks and Applications*, 5(3) :179–189, 2000.

[111] S. Lee, M. Gerla, and C.-K. Toh. A Simulation Study of Table-Driven and On-Demand Routing Protocols for Mobile Ad-Hoc Networks. *IEEE Network*, 13(4) :48–54, 1999.

[112] D. Perkins, H. Hughes, and C. Owen. Factors Affecting the Performance of Ad Hoc Networks. In *Proceedings of IEEE ICC'02*, volume 4, pages 2048–2052, April-May 2002. New York, USA.

[113] D. Cavin, Y. Sasson, and A. Schiper. On the Accuracy of MANET Simulators. In *Proceedings of ACM POMC'02*, pages 38–43, October 2002. Toulouse, France.

[114] C. Cordeiro, H. Gossain, and D. Agrawal. Multicast over Wireless Mobile Ad Hoc Networks : Present and Future Directions. *IEEE Network*, 17(1) :52–59, Jan-Feb 2003.

[115] H. Gossain, C. Cordeiro, and D. Agrawal. Multicast : Wired to Wireless. *IEEE Communications Magazine*, 40(6) :116–123, June 2002.

[116] S. Lee, W. Su, J. Hsu, M. Gerla, and R. Bagrodia. A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols. In *Proceedings of IEEE Infocom'00*, pages 565–574, March 2000. Tel Aviv, Israel.

[117] E. Royer and C. Perkins. Multicast Operation of the Ad Hoc On-Demand Distance Vector Routing Protocol. In *Proceedings of ACM Mobicom*, pages 207–218, August 1999. Seattle, USA.

[118] A. Laouiti, P. Jacquet, P. Minet, L. Viennot, T. Clausen, and C. Adjih. Multicast Optimized Link State Routing - INRIA Research Report 4721, February 2003.

[119] C. Wu and Y. Tay. AMRIS : A Multicast Protocol for Ad Hoc Wireless Networks. In *Proceedings of IEEE Milcom'99*, November 1999. Atlantic City, NJ, USA.

[120] C. Chiang, M. Gerla, and L. Zhang. Forwarding Group Multicast Protocol (FGMP) for Multihop, Mobile Wireless Networks. *ACM Journal of Cluster Computing*, 1(2) :187–196, August 1998.

[121] S. Lee, W. Su, and M. Gerla. On-Demand Multicast Routing Protocol in Multihop Wireless Mobile Networks. *Mobile Networks and Applications*, 7(6) :441–453, December 2002.

[122] J. Garcia-Luna-Aceves and E. Madruga. The Core-Assisted Mesh Protocol. *IEEE Journal on Selected Areas in Communications*, 17(8) :1380–1394, August 1999.

[123] A. Ballardie. RFC-2189 - Core Based Trees (CBT version 2) Multicast Routing - Protocol Specification, September 1997.

[124] J. Xie, R. Talpade, A. McAuley, and M. Liu. AMRoute : Ad Hoc Multicast Routing Protocol. *Mobile Networks and Applications (Kluwer)*, 7(6) :429–439, December 2002.

[125] K. Chen and K. Nahrstedt. Effective Location-Guided Tree Construction Algorithms for Small Group Multicast in MANET. In *Proceedings of IEEE Infocom'02*, pages 1180–1189, June 2002. New York, USA.

[126] L. Ji and M. Corson. Differential Destination Multicast - A MANET Multicast Routing Protocol for Small Groups. In *Proceedings of IEEE Infocom'01*, pages 1192–1202, April 2001. Anchorage, Alaska.

[127] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, and M. Carney. RFC-3315 - Dynamic Host Configuration Protocol for IPv6 (DHCPv6), July 2003.

[128] S. Thomson and T. Narten. RFC-2462 - IPv6 Stateless Address Autoconfiguration, December 1998.

[129] A. Conta and S. Deering. RFC-2463 - Internet Control Message Protocol (ICMP6) for the Internet Protocol Version 6 (IPv6) Specification, December 1998.

[130] K. Weniger and M. Zitterbart. IPv6 Autoconfiguration in Large Scale Mobile Ad-Hoc Networks. In *Proceedings of European Wireless Conference 2002*, February 2002. Florence, Italy.

[131] R. Wakikawa, J. Malinen, C. Perkins, A. Nilsson, and A. Tuominen. Internet Connectivity for Mobile Ad hoc Networks. *Wirel. Comm. and Mobile Computing*, 2(5) :465–482, August 2002.

[132] J. Xi and C. Bettstetter. Wireless Multihop Internet Access : Gateway Discovery, Routing, and Addressing. In *Proceedings of 3GWireless*, May 2002. San Francisco, CA, USA.

[133] C. Jelger and T. Noel. Internet Draft - Gateway and address autoconfiguration for IPv6 ad hoc networks - draft-jelger-manet-gateway-autoconf-v6-02.txt, April 2004.

[134] Y. Rekhter and T. Li. RFC-1771 - A Border Gateway Protocol 4 (BGP-4), March 1995.

[135] G. Huston. RFC-3221 - Commentary on Inter-Domain Routing in the Internet, December 2001.

[136] H. Lundgren, E. Nordstrom, and C. Tschudin. The Gray Zone Problem in IEEE 802.11b based Ad hoc Networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 6(3) :104–105, July 2002.

[137] C. Bettstetter, G. Resta, and P. Santi. The Node Distribution of the Random Waypoint Mobility Model for Wireless Ad Hoc Networks. *IEEE Trans. Mobile Computing*, 2(3) :257–269, July-September 2003.

[138] J. Yoon, M. Liu, and B. Noble. Random Waypoint Considered Harmful. In *Proceedings of IEEE Infocom'03*, April 2003. San Francisco, CA, USA.

[139] W. Navidi, T. Camp, and N. Bauer. Improving the Accuracy of Random Waypoint Simulations Through Steady-State Initialization. In *Proceedings of the 15th Int. Conf. on Modeling and Simulation (MS'04)*, March 2004. Marina del Ray, CA, USA.

[140] R. Hekmat and P. Van Mieghem. Degree Distribution and Hopcount in Wireless Ad-Hoc Networks. In *Proceedings of IEEE ICON'03*, Sept-Oct. 2003. Sydney, Australia.

[141] S. Wee and J. Apostolopoulos. Secure Scalable Video Streaming for Wireless Networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, May 2001. Salt Lake City, UT, USA.

[142] J. Smith, R. Mohan, and C. Li. Transcoding Internet Content for Heterogeneous Client Devices. In *Proceedings of the IEEE International Symposium on Circuits And Systems (ISCAS'98)*, May 1998. Monterey, CA, USA.

[143] J. Flinn and M. Satyanarayanan. Energy-aware adaptation for Mobile Applications. In *Proceedings of the 17th ACM Symposium on Operating Systems Principles*, pages 48–63, December 1999. Charleston, USA.

Liste des publications

Publications internationales

T. Noel et C. Jelger, **A Unified Architecture to Support Unicast and Multicast Communications with Mobile Nodes**, *International Journal of Wireless and Mobile Computing (IJWMC)*, à paraître.

C. Jelger et T. Noel, **Performance Evaluation of Multicast Transmissions with Mobile Sources**, *Proceedings of IEEE ICON'03*, Sydney, Australie, pages 723-728, Septembre-Octobre 2003.

C. Jelger et T. Noel, **An Analysis of Multicast Delivery with Mobile Receivers**, *Proceedings of IEEE PIMRC'03*, Beijing, Chine, vol. 3, pages 2690-2695, Septembre 2003.

C. Jelger et T. Noel, **Supporting Mobile SSM Sources for IPv6**, *Proceedings of IEEE Globecom'02*, Taipei, Taiwan, vol. 2, pages 1693-1697, Novembre 2002.

C. Jelger et T. Noel, **Multicast for Mobile Hosts in IP Networks : Progress and Challenges**, *IEEE Wireless Communications Magazine*, vol. 9, no. 5, pages 58-64, Octobre 2002.

Publication nationale

C. Jelger, **Multipoint et Mobilité : Optimisations pour IPv6**, *Actes de Algotel'02*, Mèze, France, pages 109-113, Mai 2002.

Internet Drafts

C. Jelger et T. Noel, **Gateway and address autoconfiguration for IPv6 ad hoc networks**, *draft-jelger-manet-gateway-autoconf-v6-02.txt*, Avril 2004. Soumis au groupe de travail MANET.

Livrable dans le cadre du projet européen 6NET (IST-2001-32603)

C. Jelger et T. Noel, **Multicast with Mobile Hosts : Analysis and Performance Evaluation**, Deliverable 4.1.5 version 1, Décembre 2003. Disponible sur le site du projet http ://www.6net.org.

Lexique

Adresse MAC	Medium Access Control - cette adresse de 48 bits identifie de manière unique une interface de communication (par exemple une carte réseau Ethernet)
DNS	Domain Name System - système distribué sur l'Internet permettant la conversion d'un nom de machine en adresse IP et vice versa
EUI-64	Ethernet Unique Identifier - identificateur global d'interface de 64 bits proposé par l'IEEE
ICMP	Internet Control Message Protocol - protocole de contrôle utilisé dans l'Internet
IEEE	Institute of Electrical and Electronics Engineering - institut responsable de la normalisation de systèmes de communication pour les réseaux
IETF	Internet Engineering Task Force - organisme international chargé de la standardisation des protocoles de l'Internet
IP	Internet Protocol - protocole réseau utilisé actuellement dans l'Internet
IPv6	Internet Protocol version 6 - nouveau protocole réseau de l'Internet
Mobile IPv6	Protocole de gestion de la mobilité pour IPv6
MSSMSv6	Mobile SSM Sources for IPv6 - protocole permettant de gérer la mobilité de la source d'un canal SSM
OSI	Open Systems Interconnection - modèle historique utilisé pour définir l'architecture des systèmes de communication
SSM	Source Specific Multicast - modèle de communication multicast basé sur le concept de canal (S,G) dans lequel seule une source S envoie des données à une adresse de groupe G
RWP	Random WayPoint - modèle utilisé pour simuler la mobilité des nœuds d'un réseau ad hoc

Table des figures

Liste des tableaux