République Algérienne Démocratique et Populaire

Ministère de l'enseignement Supérieur et de la Recherche Scientifique
Faculté des Sciences Exactes
Département d'informatique

Mémoire

Présenté par

HEDJAZ Sabrine

En vue de l'obtention du diplôme de Magister

Filière : Informatique

Option : Cloud computing

Thème

Routage réactif à basse consommation de bande passante dans les réseaux mobiles ad hoc

---

Soutenu le 14/09/2014 devant le jury composé de :

BOUKERRAM Abdallah	Professeur	Univ.de Béjaia	Président.
TARI Abdelkamel	M.C.A	Univ.de Béjaia	Rapporteur.
MELIT Ali	Professeur	Univ.de Jijel	Examinateur.
SEMCHEDINE Fouzi	M.C.A	Univ.de Béjaia	Examinateur.
OMAR Mawloud	M.C.B	Univ.de Béjaia	Invité.

Année Universitaire : 2012-2013

Résumé

Nos travaux se situent dans le contexte des réseaux mobiles ad hoc, qui constituent une catégorie des réseaux sans fil pouvant être déployés sans infrastructure. Le routage dans les réseaux ad hoc présente de nombreux challenges dus tant aux contraintes technologiques qu'aux différents contextes applicatifs. Notre travail traite la problématique de routage à basse consommation de la bande passante. Le protocole proposé DS2R2P (on Demand Source Routing with Reduced Packet Protocol) est une extension du protocole DSR (on Demand Source Routing) pour les grands réseaux en réduisant la charge de communication. Notre protocole repose sur une politique de réduction de paquets en utilisant un mécanisme de hachage où les données sont transmises en utilisant des valeurs de tailles fixes et réduites. Les résultats de simulation obtenus montrent l'efficacité de notre protocole en termes de temps de transmission ainsi que de charge de communication.

Mots clés : Réseaux mobiles ad hoc, Protocole de routage, DSR, DS2R2P.

Abstract

Mobile ad-hoc networks (MANETs) are a specific type of wireless networks that can be deployed without pre-existing infrastructures. Routing in ad hoc networks presents many challenges due to technological constraints as well as different application contexts. Our work address the problem of routing with preservation of bandwidth. The proposed protocol DS2R2P (on Demand Source Routing with Reduced Packet Protocol) is an extension of DSR protocol (on Demand Source Routing) for large networks reducing packet control overhead. Our protocol is based on a policy of reducing packet sizes using hashing mechanisms, where data are transmitted with a fixed value and reduced header packets. Simulation results shows the efficiency of the proposed protocol in terms of transmission time and overhead.

Keywords : Mobile ad hoc networks, Routing protocol, DSR, DS2R2P.

Dédicaces

À mes parents adorés, qui ont toujours été là pour moi

Qui ont su par tant de sacrifices, me soutenir et m'encourager

Et qui m'ont donné tant d'amour

Mille mercis pour tous ce que vous avez faits pour moi

J'espère que vous êtes fière de moi.

À mes adorables et très chers frères et sœur que j'aime tant.

À toutes ma famille.

À toutes mes chères amies.

À tous mes collègues de la promotion Cloud Computing.

Sabrine

Remerciements

Avant tout je remercie le bon **Dieu** qui m'a donné le courage, la force et la volonté pour continuer. Merci de m'avoir éclairé le chemin de réussite.

Je tiens à exprimer ma profonde gratitude et mes vifs remerciements à **Dr. TARI Abdelkamel**, directeur de mémoire, pour l'énorme soutien scientifique et moral qu'il a su m'adresser pendant cette période.

Je remercie très vivement monsieur **Dr. OMAR Mawloud**, mon co-promoteur pour la confiance qu'il m'a accordée, pour ses précieuses directives et pour tous le temps qu'il m'a consacré pour la réussite de ce travail, merci beaucoup.

Je suis reconnaissante envers monsieur **Pr. BOUKERRAM Abdellah**, d'avoir présidé le jury. Je tiens également à remercier **Pr. MELIT Ali** ainsi que **Dr. SEMCHEDINE Fouzi**, d'avoir accepté de juger notre travail.

Mes vifs remerciements vont également à tous ceux qui ont, de loin ou de près contribuer à l'accomplissement de ce travail.

Je remercie aussi mes professeurs, mes collègues, mes amies et toutes les personnes qui m'ont aidé durant mes études universitaires.

Enfin, ma reconnaissance s'adresse à ma famille qui a su m'apporter, sans relâcher leur soutien durant toutes ces longues années d'études.

À tous, merci du fond du cœur.

Table des matières

Liste des figures

Introduction générale

Le développement technologique qu'a vu le monde d'aujourd'hui a touché tous les domaines, particulièrement le secteur de la communication qui connaît une évolution considérable par l'apparition de la technologie sans-fil. On distingue deux grandes familles de réseaux sans fil : les réseaux avec infrastructure et les réseaux sans infrastructure (ou ad hoc). Dans les réseaux avec infrastructure, les communications s'effectuent via une station de base fixe. Cette approche est utilisée dans les réseaux sans fil traditionnel comme les réseaux GSM (Global System for Mobile communications) et les réseaux locaux sans fil. Cependant, un réseau ad hoc est constitué d'un ensemble d'unités mobiles communiquant via un médium radio et ne requiert ni infrastructure fixe ni administration centralisée.

Un réseau ad hoc est composé de nœuds mobiles qui peuvent communiquer directement entre eux s'ils sont situés à portée radio. La portée étant relativement limitée, le déploiement d'un réseau à grande échelle nécessite que le réseau soit à multi-saut, c'est-à-dire que des nœuds intermédiaires fassent office de point de relais. Les réseaux mobiles ad hoc, grâce à leur auto-organisation et à l'absence d'infrastructure, peuvent facilement être déployés dans de nombreux domaines. Un tel réseau se caractérise également par leurs faibles ressources sur la totalité de la ligne de communication. Le médium de communication offre une bande passante limitée par rapport aux réseaux filaires. Les nœuds mobiles disposent de ressources matérielles limitées et hétérogènes en termes de batterie et de puissance de calcul. Les interconnexions

des noeuds sont instables à cause d'une part de l'utilisation de la technologie sans fil et d'autre part, de la mobilité fréquente des noeuds. Toutes ces spécificités imposent des contraintes supplémentaires aux concepteurs et aux développeurs de services dédiés aux réseaux ad hoc et les poussent à réfléchir à de nouvelles approches. Les protocoles de routage doivent donc s'ajuster aux contraintes des réseaux ad hoc. Un protocole de routage doit être efficace en bande passante, énergie consommée, etc. Ces protocoles doivent s'adapter rapidement aux changements de topologie.

À cause de la mobilité des nœuds il est très difficile de localiser une destination à un instant donné, les protocoles de routage conçus pour les réseaux statiques sont inadaptés pour ce type de réseaux. Plusieurs protocoles de routage pour les réseaux ad hoc ont été développés. Chaque protocole essaye de maximiser les performances du réseau en minimisant le délai de livraison des paquets, l'utilisation de la bande passante et la consommation d'énergie. Les algorithmes de routage dans les réseaux ad hoc peuvent être classés en trois catégories, les protocoles proactifs, les protocoles réactifs et les protocoles hybrides. Nous nous focalisons dans le cadre de ce travail aux protocoles réactifs.

Parmi les protocoles réactifs, DSR (Dynamic Source Routing) est l'un des plus étudiés. DSR est un protocole de routage par la source. L'émetteur des données doit fournir la route (la liste des noeuds à traverser) nécessaire pour atteindre la destination. Le chemin source est présent dans l'en-tête de chaque paquet de données. La source diffuse un paquet de requête qui va atteindre tous les nœuds du réseau. La destination recherchée répond à la source pour lui fournir la route. L'un des principaux avantages du DSR réside dans son fonctionnement purement réactif car les mécanismes de routage ne sont déployés qu'en cas de besoin en réduisant ainsi la consommation de la bande passante. L'utilisation du routage par la source (toutes les informations nécessaires au routage d'un paquet de données sont contenues dans celui-ci) permettant d'éviter les boucles ainsi que la mémorisation des informations de routage sur les nœuds intermédiaires. Cependant, l'utilisation du routage source

implique une surcharge importante de paquets notamment dans le cas des grands réseaux vus avec la route est incluse dans chaque en-tête de paquet de données.

Ce travail rentre dans le cadre de l'étude du problème de routage dans les réseaux mobiles ad hoc. Son objectif est de proposer un protocole de routage réactif à basse consommation de bande passante, et nous avons proposé DS2R2P (on Demand Source Routing with Reduced Packets Protocol), un protocole qui minimise la charge des messages de contrôle lors de l'acheminement des données en réduisant la taille de l'entête des paquets. Cela réduit la charge de transmission, le temps de délivrance de données et la consommation de bande passante.

Ce mémoire est organisé en trois chapitres. Le premier chapitre présente un état de l'art sur les protocoles de routages réactifs. Le second chapitre décrit le protocole DS2R2P que nous avons proposé. Dans le troisième chapitre nous présentons les résultats de simulation en comparant notre protocole aux protocoles : DSR et l'extension DSR (Automatic Route Shortening). Nous clôturons ce mémoire avec une conclusion et perspectives.

1

Etat de l'art sur les protocoles de routage réactifs

1.1 Introduction

Les protocoles de routage dans les réseaux ad hoc doivent être assurés l'acheminement des données et maintiennent la connexion entre les nœuds du réseau. Le protocole de routage doit prendre en considération les caractéristiques des réseaux ad hoc. Il doit minimiser les messages de contrôles (overhead) nécessaires à l'établissement et à la maintenance des routes afin de réduire la charge du réseau. Un protocole de routage doit s'adapter rapidement au changement de la topologie. La limitation des ressources (bande passante, énergie, capacité du mémoire, etc.) doit être prise en compte par les protocoles de routage. En effet, l'absence d'infrastructure dans ces réseaux, impose un fonctionnement distribué.

Plusieurs travaux de recherche sur les protocoles de routages dans les réseaux

ad hoc ont été faits : l'approche la plus intuitive dans la conception des protocoles de routage consiste à l'adaptation des protocoles traditionnels (dans les réseaux filaires) au contexte des réseaux ad hoc, ce qui a donné la naissance des protocoles proactifs. Dans un protocole de routage proactif, chaque nœud échange périodiquement sa connaissance de la topologie du réseau à ses nœuds voisins. Cela fait que les informations de routage sont toujours disponibles, mais le réseau est chargé par un trafic de contrôle dont une partie importante est inutile. D'autres protocoles de routage sont conçu pour minimiser la charge de contrôle des protocoles de routage proactifs, en ne maintenant des informations que concernant les chemins actifs. Ce sont les protocoles réactifs. Les protocoles de routages hybrides combinent les deux approches précédentes afin de tirer les avantages des deux catégories, tout en réduisant leurs inconvénients. D'autres classes sont à citer à savoir les protocoles de routage géographiques, hiérarchique, etc. Dans ce mémoire, on se focalise sur les protocoles de routage réactifs.

Dans ce chapitre, nous commencerons par la présentation des différentes caractéristiques des protocoles de routage réactifs. Dans un premier temps, nous proposerons une classification des protocoles de routage réactifs. Ensuite, nous décrirons chaque catégorie ainsi que les protocoles de routage à laquelle ils appartiennent. Enfin, une discussion sur les différents protocoles de routage présentés dans ce chapitre avec l'objectif d'optimisation de la consommation de la bande passante dans les réseaux ad hoc sera fournie.

1.2 Protocoles de routage réactifs

Les protocoles de routage réactifs créent et maintiennent les routes selon les besoins, lorsqu'un nœud vient communiquer avec une station à distance, il lance une procédure de découverte de route qui va inonder le réseau avec un paquet de requête et lui fournit les informations nécessaires pour atteindre la destination. Cette technique peut ne pas inonder le réseau par des paquets de contrôles et de ne pas

maintenir les routes non actives. S'il y a lieu d'une défaillance de lien, une procédure de maintenance de lien sera lancée.

1.3 Classification des protocoles réactifs

Les protocoles réactifs réduisent la charge des paquets de contrôles, comparés aux protocoles proactifs, surtout si le réseau est très dynamique. Mais aussi la découverte de routes en cas de coupure génère une charge supplémentaire. Le problème de ces protocoles c'est qu'ils ont un délai initial avant de commencer la transmission des paquets, provoqué par la procédure de découverte de route. En effet, le routage dans les réseaux ad hoc présente de nombreux challenges. De nombreux travaux de recherche sur les protocoles de routage dans les réseaux ad hoc ont été menés dès la fin des années 90. Les protocoles de routages réactifs proposés servent à résoudre la problématique de routage dans les réseaux ad hoc dont le routage réactif est le plus adapté à la plupart des caractéristiques de ces réseaux. Les solutions proposées sont de natures différentes, dû aux points de vues des concepteurs de ces protocoles. Nous pouvons distinguer quatre catégories des protocoles de routage réactifs selon l'axe de recherche suivi (cf. figure 1.1).

- **Protocoles de routage réactifs de base :** les recherches dans ce cas, concentrent sur l'amélioration des protocoles de routage réactifs de base, tel qu'AODV [1] et DSR [2]. Ces protocoles utilisent des mécanismes pour l'obtention et la maintenance des informations de routage. Plusieurs travaux ont été proposés, vise à l'adaptation et l'optimisation de ces protocoles par rapport à des contraintes particulières.
- **Le changement dans la topologie :** lorsque les protocoles de routage ad hoc doivent s'adapter rapidement aux changements relativement fréquents dans la topologie ; plusieurs protocoles ont été conçus afin de réduire le coût de maintenance des liens utilisant des différents mécanismes (réparation efficace des liens, la reconstruction préventive des routes, stabilité de lien, etc.).

- **Le coût de la découverte de route :** dans cet axe de recherche les protocoles consistent à minimiser le coût de l'opération de la découverte de route, tout en utilisant des mécanismes de localisation géographique, clustering, etc., afin de réduire les messages de contrôles.
- **Utilisation des mécanismes spécifiques :** dans cette classe, des nouveaux mécanismes et des approches multidisciplinaires sont utilisés (intelligence artificielle, théorie des jeux, QOS, etc.).

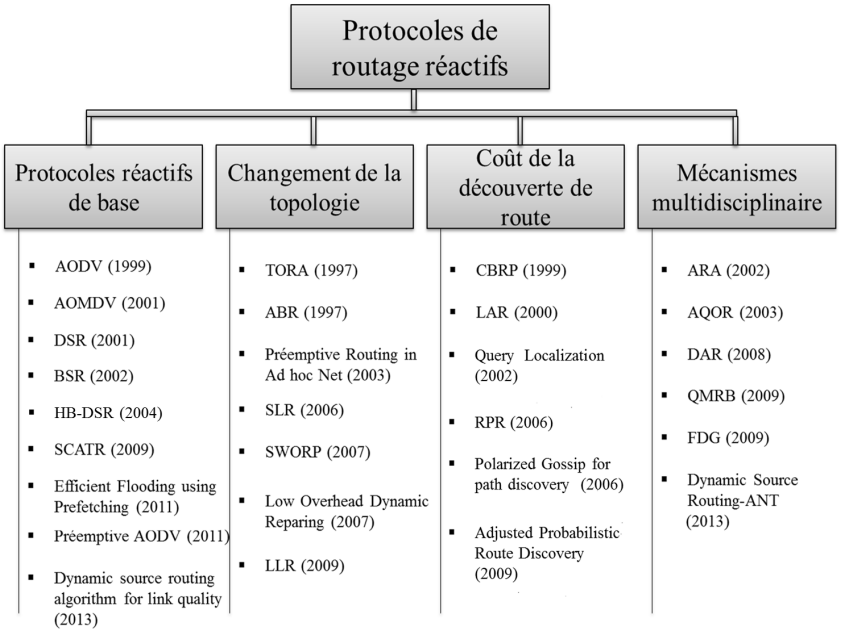


FIGURE 1.1 – Classification des protocoles de routage réactifs dans les réseaux ad hoc

1.3.1 Protocoles réactifs de base

Les protocoles réactifs découvrent le chemin quand un nœud désire envoyer un paquet vers un autre nœud du réseau, celui-ci invoque un mécanisme de découverte des chemins vers la destination. La route ainsi créée reste valide tant que le nœud final est joignable ou jusqu'à ce que la route ne soit plus utilisée. Le mécanisme de découverte d'une route est basé principalement sur deux algorithmes à savoir :

- La méthode d'apprentissage en arrière (Backward Learning)

- La méthode de routage source (Source Routing)

Dans la méthode d'apprentissage en arrière, le nœud source qui est à la recherche d'un chemin vers la destination ; diffuse par inondation une requête sur le réseau. Lors de la réception de la requête, les nœuds intermédiaires essaient de faire apprendre le chemin au nœud source. Une fois la destination est atteinte, elle peut envoyer une réponse en utilisant le chemin tracé par la requête, un chemin *full duplex* est alors établi entre le nœud source et le nœud destination.

Par contre, dans la méthode de routage source, la source de données détermine la séquence complète des nœuds à travers lesquels les paquets de données sont envoyés. En effet, afin d'envoyer un paquet de données à un autre nœud, l'émetteur construit une route source et l'inclut en tête du paquet. La construction se fait en spécifiant l'adresse de chaque nœud à travers lequel le paquet va passer pour atteindre la destination. Par la suite, l'émetteur transmet le paquet à l'aide de son interface au premier nœud spécifié dans la route source. Un nœud qui reçoit le paquet et qui est différent de la destination, supprime son adresse de l'entête du paquet reçu et le transmet au nœud suivant identifier dans la route source. Ce processus se répète jusqu'à ce que le paquet atteigne sa destination finale.

Les protocoles de routages réactifs les plus représentatifs (standardisés) sont : DSR et AODV. Ces deux protocoles ont plusieurs implémentations indépendantes pour les différents systèmes d'exploitation. Dans cette classe on va détailler le fonctionnement de ces protocoles avec quelques travaux proposés visant l'adaptation et l'optimisation de ces protocoles selon plusieurs critères.

Ad hoc on-demand distance vector

Le protocole AODV (Ad hoc On-demand Distance Vector) [1] a été développé par Perkins et Royer, comme amélioration de l'algorithme DSDV (Destination sequensed distance vector) [3]. Le protocole AODV réduit le nombre de diffusions des messages et cela en créant les routes lors du besoin, contrairement au DSDV qui maintien la

totalité des routes. AODV utilise les principes de numéros de séquence afin d'éviter la présence de boucles et de maintenir la consistance des informations de routage. À cause de la mobilité des nœuds dans les réseaux ad hoc, les routes maintenues par certains nœuds deviennent invalides. Les numéros de séquences permettent d'utiliser les routes les plus fraîches. Chaque nœud les met à jour chaque fois qu'une nouvelle information pérennante d'un message RREQ, RREP ou RERR, il incrémente son propre numéro de séquence dans les circonstances suivantes :

- Il est lui-même le nœud destination et offre une nouvelle route pour l'atteindre.
- Il reçoit un message AODV contenant des nouvelles informations sur le numéro de séquence d'un nœud destination.
- Le chemin vers une destination n'est plus valide.

Ce protocole met en œuvre différentes opérations pour réaliser et maintenir le routage : gestion de la connectivité locale, phase de découverte des routes et maintenance des routes. La fonctionnalité de gestion de la connectivité locale est appliquée par les nœuds de la manière suivante. Chaque nœud émet périodiquement un paquet nommé HELLO. À la réception de ce paquet, les nœuds apprennent la présence des nœuds voisins. La connectivité locale est modifiée dans les cas suivants : un nœud reçoit un paquet HELLO transmis par un nouveau voisin ou un nœud ne reçoit plus de paquets HELLO durant un laps de temps défini. La phase de découverte des routes par le protocole AODV est la suivante. Un nœud source cherche à envoyer un paquet de données à une destination, il vérifie sa table de routage pour voir s'il y a un chemin valide au nœud destinataire. Si un chemin existe, le paquet est transmis vers le prochain nœud, sinon la phase de découverte des routes est engagée. Le paquet est mis en file d'attente le temps d'obtenir une route, puis la source diffuse un paquet de création de route (RREQ : Route Request) à ses voisins immédiats et ces nœuds diffusent plus loin à leurs voisins jusqu'à ce que la demande atteigne soit un nœud intermédiaire qui a un chemin à la destination ou le nœud destinataire lui-même. Le paquet de création de route (RREQ) contient *l'adresse IP du nœud source*, *numéro*

de séquence courant (du nœud source) , *l'adresse IP du nœud destinataire*, *numéro de séquence de la destination* (contient la dernière valeur connue du numéro de séquence associé au nœud destination ; cette valeur recopiée de la table de routage. Si le numéro de séquence n'est pas connu, la valeur nulle sera prise par défaut) et le *nombre de sauts parcourus*. Quand un nœud intermédiaire reçoit le paquet de la requête, il vérifie dans sa table historique si cette requête a été déjà vue et traitée. Si le paquet est dupliqué, le nœud doit l'ignorer et arrêter le traitement. Dans le cas contraire le couple (@ source, ID de requête) sera inscrit dans la table historique pour rejeter les futures doublons, et le nœud continue le traitement en cherchant la destination dans sa table de routage. S'il possède une route récente, à noter qu'une route est récente si le numéro de séquence de la destination dans la table est supérieur ou égal au numéro de séquence dans le paquet RREQ. Dans ce cas, le nœud envoie un paquet de réponse (RREP) à la source lui indiquant comment atteindre la destination. Autrement, le nœud ne connaît pas la route vers la destination ; il incrémente le nombre de sauts et rediffuse le paquet. Quand un nœud intermédiaire envoie le paquet RREQ à un voisin, il sauvegarde aussi l'identificateur du nœud à partir duquel la première copie de la requête est reçue. Cette information est utilisée pour construire le chemin inverse qui sera traversé par le paquet de réponse de route (RREP : Route Reply) de manière unicast. Puisque le paquet de réponse de route RREP va être envoyé à la source, les nœuds appartenant au chemin de retour vont modifier les tables de routage suivant le chemin contenu dans le paquet de réponse. L'opération générale d'AODV est représenté dans la figure 1.2 AODV utilise les messages de contrôle HELLO qui permettent de vérifier la connectivité et l'activité des routes. Un nœud détermine l'activité d'une route en écoutant périodiquement les messages HELLO transmis par ses voisins. Si pendant un laps de temps, trois messages HELLO ne sont pas reçus consécutivement, le nœud considère que le lien vers ce voisin est cassé. Il envoie un message d'erreur RERR à chacun de ses voisins ascendants pour assurer la suppression de cette partie particulière du chemin.

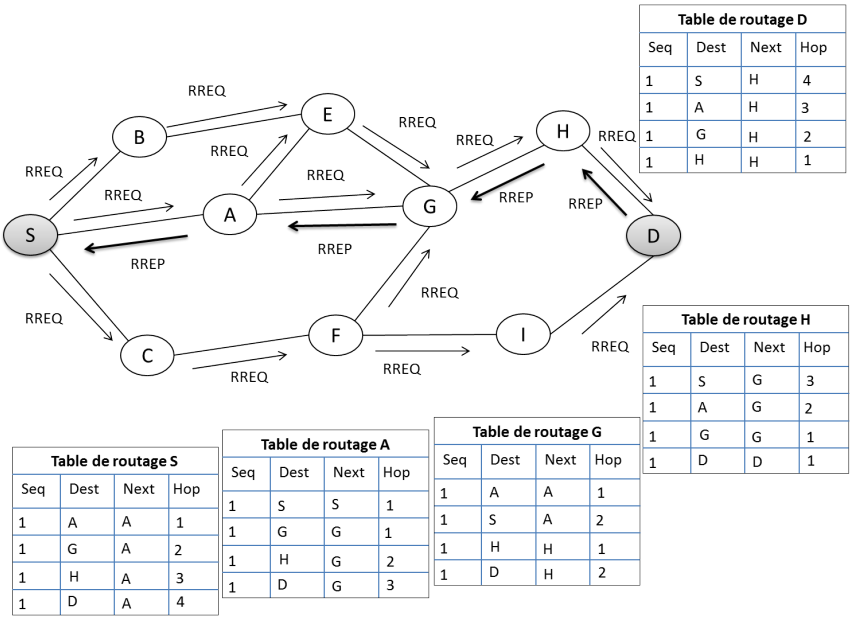


FIGURE 1.2 – Protocole de routage AODV

Une fois que le message atteint au nœud source, il entame une nouvelle phase de découverte de route.

Ad hoc on-demand multiple distance vector

AOMDV (Ad hoc On-demand Multiple Distance Vector) [4] est une extension du protocole AODV. L'idée d'AOMDV est de trouver multiples routes (sans boucles et lien disjoints) vers la même destination. Pour garder trace de plusieurs chemins, à chaque destination est associée une liste de sauts prochains avec le nombre de sauts correspondant. Pour éviter la formation des boucles, un nœud n'accepte que les chemins alternatifs (définis par des RREQ ou des RREP) dont le nombre de sauts est inférieur à celui annoncé. Pour trouver des chemins *lien disjoints* entre une paire de nœuds, chaque nœud intermédiaire ne supprime pas tous les paquets RREQ dupliqués, mais il accepte ceux arrivant via différents voisins du nœud source. Si ce nœud intermédiaire maintient un chemin valide vers la destination, il répond au nœud source par RREP. Pour le nœud destination, il répond à k copies du RREQ arrivant via des voisins différents. Après le premier saut, les paquets RREP

suivent les chemins inverses qui sont des nœuds disjoints et par conséquent lien disjoints. Notons que les trajectoires des paquets RREP peuvent s'en croiser à un nœud intermédiaire, mais chaque RREP prend un chemin inverse différent vers la source pour assurer la disjonction des liens.

Combining on-demand and opportunistic routing for intermittently connected networks

Boice et al. [5] Proposent un protocole de routage qui prend en considération la possibilité de connectivité intermittente dans les réseaux mobile ad hoc. (SCaTR Combining on-demand and opportunistic routing for intermittently connected networks). Il utilise les informations des anciennes connectivités définissant des nœuds proxy pour router le trafic vers la destination en absence de routes directes. Il est basé sur le protocole AODV de manière que lorsque le réseau est entièrement connecté, il fonctionne comme le protocole AODV. Lorsque le réseau est partitionné, les nœuds proxy sont créés à base de distance entre la source et la destination. Un nœud plus proche à la destination est annoncé comme proxy de destination et considéré comme un tampon des messages sur la route jusqu'à ce que la destination soit découverte ou un autre nœud est choisi comme un meilleur proxy. Ce protocole est constitué de plusieurs phases : maintenance de table de contact, la découverte de routes, la sélection de la route, la découverte de proxy. Chaque nœud dans le réseau maintient une table de contact contenant une mesure de la distance en fonction du temps par rapport aux autres nœuds du réseau. Chaque entré de la table comprennent une adresse destination et sa valeur de contact courant. Les nœuds maintiennent ces tables avec l'information qui est superposée sur les messages de contrôle. Quand un nœud reçoit un message HELLO par un voisin, il reçoit également la table de contact de ce nœud. Il utilise cette table pour apporter des modifications à sa propre table de contact. Comme la maintenance de ces informations est couteuse, SCaTR initialise la table de contacts à la demande. Pendant la phase de découverte de route s'il y a aucune route établie vers la destination, une demande de proxy PREQ est transmise pour

trouver le proxy destination le plus proche. Le paquet PREQ pourrait également être envoyé pour des multiples destinations et une réponse proxy a été transmise. Le paquet PREP contient les informations à jour de la valeur de contact pour la destination, son espace tampon restant et le nombre de messages stockés pour des paires source-destination particuliers à partir desquelles il a reçu PREQ. La source rassemble tous les paquets PREP, compare les valeurs de contact obtenus et met à jour sa table avec la valeur la plus élevée, ainsi que la route où les paquets de données sont transmis. L'opération générale de SCaTR est illustrée sur la figure 1.3. Le

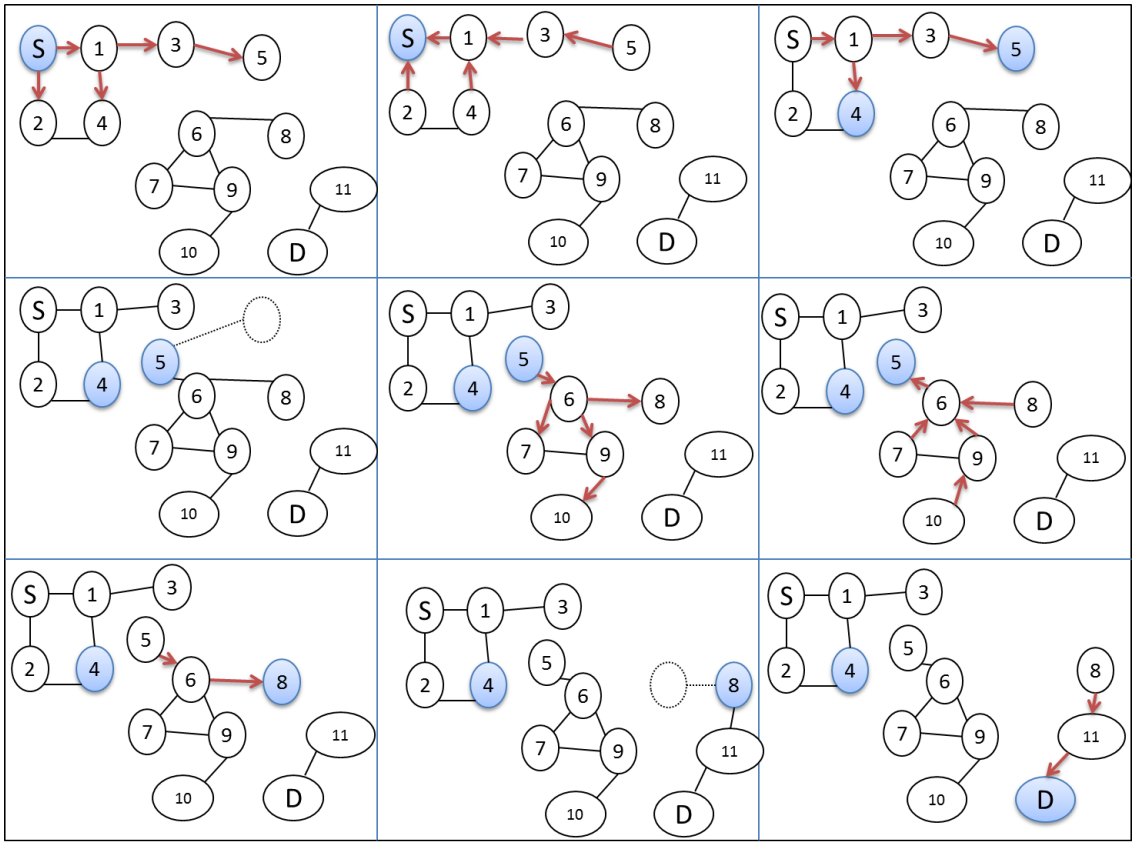


FIGURE 1.3 – Illustration du fonctionnement SCATR

nœud source (S) possède des données à envoyer à la destination (D). Le nœud source initiait premièrement la découverte de proxy dans sa partition locale. Après avoir trouvé les deux nœuds avec les meilleures valeurs de contact pour une destination donnée, la source sélectionne ces deux nœuds en tant que proxy et leur envoie des

données pour les enregistrer. Lorsque l'un des proxys se joint à une nouvelle partition, il initie le processus de découverte de proxy et sélectionne le meilleur proxy destination dans cette partition. Enfin, un proxy atteint la partition contenant la destination et délivre les données.

Preemptive AOMDV routing for mobile Ad hoc networks

Subbiah et Ramesh ont proposé une extension du protocole AODV (Preemptive AOMDV Routing for mobile ad hoc networks) [6] afin de réduire la charge produite due à la détection des défaillances des liens et de la maintenance des routes. L'idée d'AOMDV est de trouver multiples routes vers la même destination. Le nœud destination envoie plusieurs RREP un pour chaque chemin [4]. En se basant sur la qualité de signal, cette solution peut prédire que le lien peut se rompre. Le paquet RERR d'AODV nécessite un changement contenant des informations sur le type d'erreur (deux types : *lien est rompu*, *lien peut se rompre*). Si un nœud est dans la zone préventive, un paquet RERR est envoyé vers la source (cf. figure 1.4). La source cherche une autre route et change le chemin, sinon dans le pire des cas, elle initialise une nouvelle opération de découverte de routes.

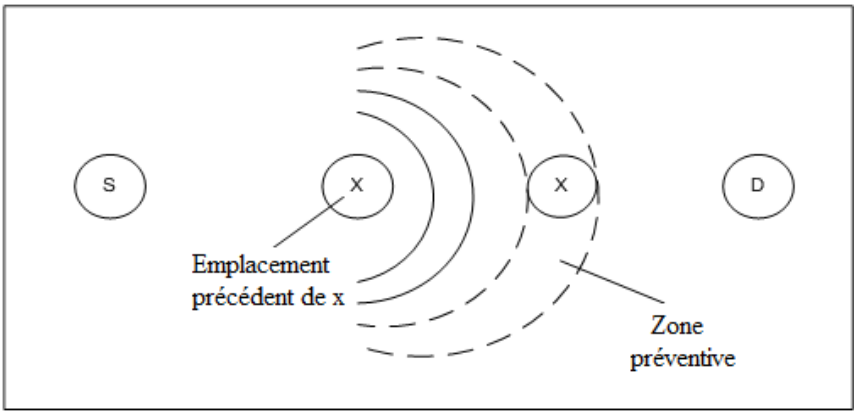


FIGURE 1.4 – La zone préventive

Dynamic source Routing

Le protocole DSR (Dynamic Source Routing) est un protocole de routage réactif simple et efficace conçu spécifiquement pour une utilisation dans les réseaux sans fil ad hoc multi-hop. Ce protocole a été élaboré dans le cadre du projet Monarch (Mobile Network Architectures) mené par le département informatique de l'université de Rice [28]. Le protocole DSR est basé sur l'utilisation de la technique du routage par la source. Dans cette technique l'émetteur des paquets détermine une séquence complète des nœuds par lesquels doivent passer les paquets pour atteindre la destination. Ainsi, les nœuds intermédiaires relaient le paquet selon la route indiquée. DSR utilise un en-tête d'options qui peut être inclus dans tout paquet IP pour véhiculer le trafic de contrôle. Cet en-tête DSR suit immédiatement l'entête IP du paquet (dans IPv4) ou l'option Saut-Par-Saut (Hop-By-Hop définie dans IPv6) si elle est présente (cf. figure 1.5). Une ou plusieurs options DSR peuvent suivre l'en-tête DSR. Chaque option est caractérisée par un type. Les plus utilisées sont l'option Route Source (la liste de nœuds à traverser), Route Request (requête de route), Route Reply (réponse de route) et Route Error (erreur de route). Le fonctionnement basique de DSR s'avère assez simple à mettre en œuvre. Il met en place uniquement deux phases : la phase de découverte des routes, et la phase de maintenance de ces mêmes chemins. L'opération de découverte de route, permet à n'importe quel nœud

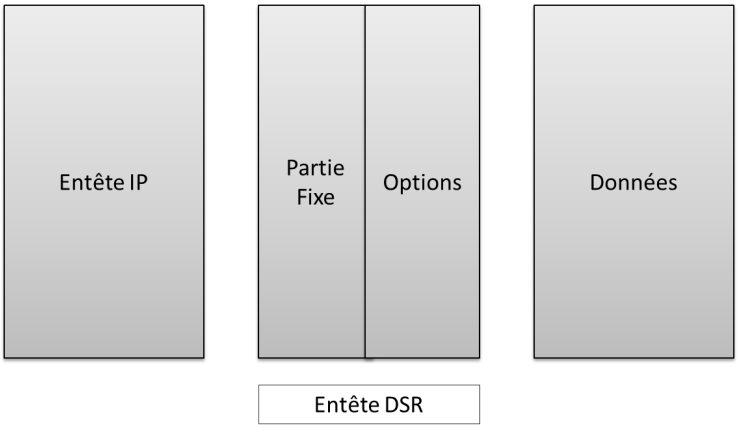


FIGURE 1.5 – Entête DSR dans un paquet IP

de réseau ad hoc de découvrir dynamiquement un chemin vers un nœud quelconque du réseau. Ainsi, DSR étant un protocole réactif, un nœud source va chercher une route uniquement s'il veut émettre un paquet vers un nœud destinataire, et qu'il ne possède aucune route vers celui-ci dans son cache. La source diffuse un paquet de requête (Route Request) contenant l'adresse de la destination recherchée, une liste dans laquelle les adresses des nœuds traversées sont conservées ainsi qu'un identifiant de requête (cf. figure 1.6). Lorsqu'un nœud intermédiaire reçoit le paquet de

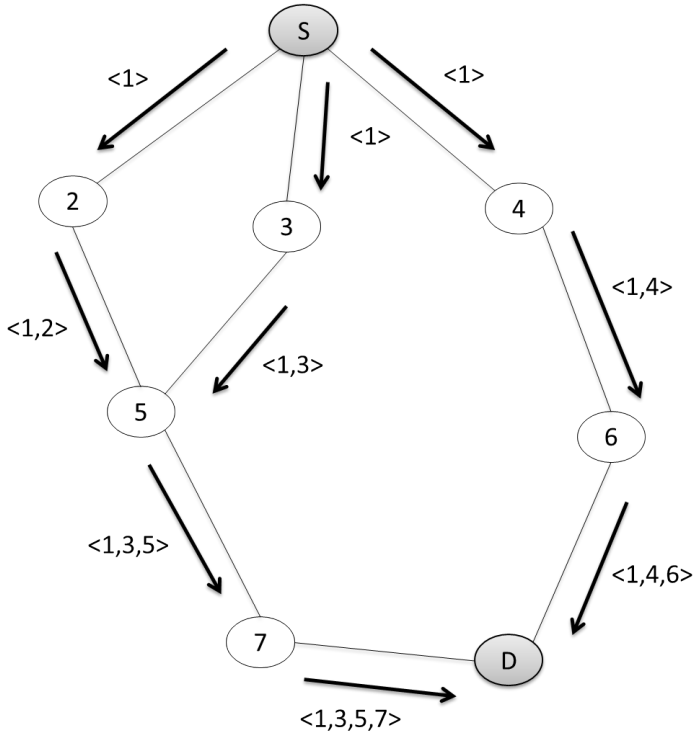


FIGURE 1.6 – Découverte de route dans DSR (Envoi de paquet RREQ)

requête, il vérifie qu'il ne pas déjà traité grâce à l'identifiant de requête, et en vérifiant que son adresse n'est pas déjà dans la liste des nœuds traversés. Si c'est le cas, il s'enregistrait dans le chemin (la liste), et le propager à tous ses voisins. Sinon le paquet est supprimé. Lorsque la destination envisagée est atteinte, elle renvoie un paquet de réponse (Route Reply), vers la source (cf. figure 1.7).

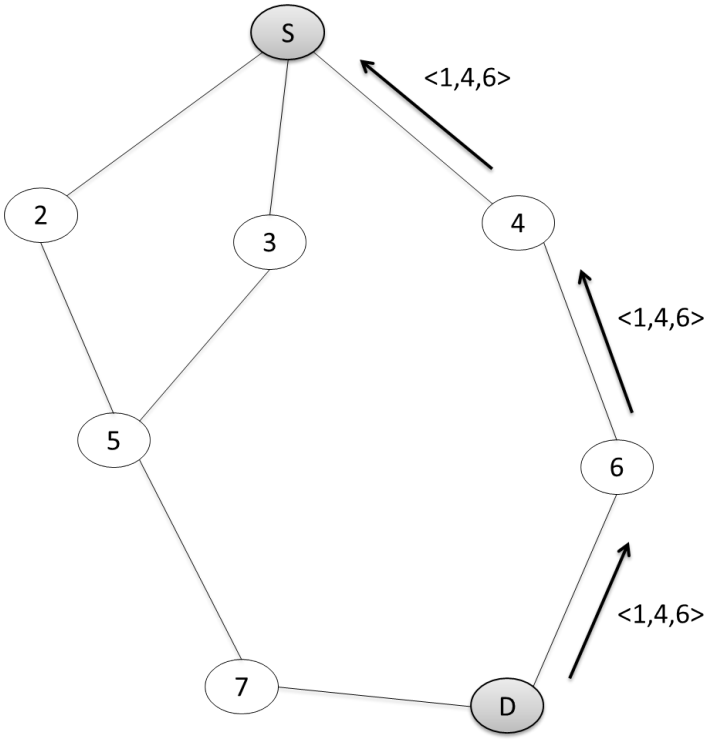


FIGURE 1.7 – Découverte de route dans DSR (Envoi de paquet RREP)

Optimisations sur DSR

De nombreuses optimisations ont été apporté à l'algorithme DSR de base, au niveau des phases de découverte et de maintenance de route pour minimiser le trafic de contrôle et accroître les performances du protocole. Les optimisations décrites ci-après sont définies dans [28].

– **Écoute Active (Caching overheard routing information) :** lorsque qu'un nœud transmet un paquet de données ou qu'il écoute le support, il peut ajouter à son propre cache les informations de routage utiles de ce paquet. L'utilité de ces informations dépend aux caractéristiques de directivité du support physique ainsi que le protocole Mac utilisé. On peut citer les cas suivants :

- un nœud S, trouve la route [S,E,F,J,D] vers le nœud D, donc le nœud S peut être ajouté la route [S,E,F] vers le nœud F à son propre cache .
- si un nœud K, renvoi un paquet de requête RREQ [S,C,G] destiné à un autre

nœud, le nœud K peut ajouter la route [K,G,C,S] à son propre cache.

- si un nœud F transfère un paquet de réponse RREP [S,E,F,G,D] le nœud F peut ajouter la route [F,G,D] vers le nœud D à son propre cache.
- si un nœud E transfère un paquet de données [S,E,F,G,D] le nœud E peut ajouter la route [E,F,G,D] vers le nœud D à son propre cache.
- si un nœud X écoute le nœud F envoyer un paquet de données [S,E,F,J,D], le nœud X peut ajouter un chemin [X,F,G,D], vers D à son propre cache.

– **Réponse du cache (Replying to Route Requests Using Cached Routes) :** lorsqu'un nœud reçoit une requête de route pour laquelle, il envoie directement une réponse à la source et ne rediffuse pas la requête. Ainsi la découverte de route est accélérée et la charge d'overhead sur le réseau est diminuée. Touts fois avans de transmettre un paquet de réponse qui a été généré en utilisant des informations à partir de son cache, un nœud doit vérifier que le chemin résultat étant envoyé dans la réponse ne contient pas des nœuds en double. La figure 1.8 illustre un cas où le chemin de réponse contient des nœuds en double.

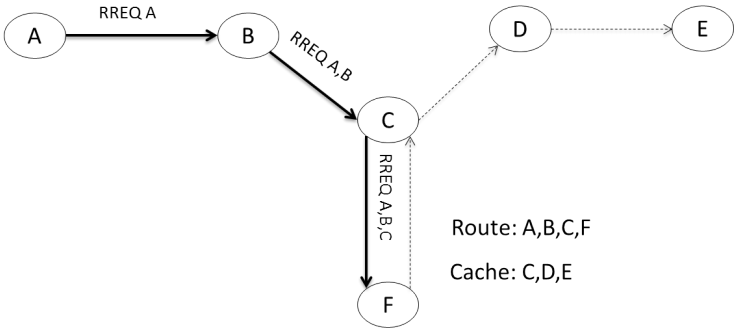


FIGURE 1.8 – Replying to Route Requests Using Cached Routes

– **Limitation de la porté de requête :** cette optimisation consiste à ajuster le TTL du paquet IP, qui contient la requête de route. Le but est d'éviter l'inondation du réseau avec des paquets de requête. Deux types de découverte de route en déroulent :

- **réponse des voisins (Non-propagating Route Request) :** quand un nœud entreprend une découverte de routes, il envoie en premier lieu une requête en positionnant le TTL (du paquet IP qui la contient) à 1, afin qu'elle ne soit pas rediffusée par ses voisins. Si aucune réponse n'a été reçue après expiration d'un timer, la source déclenche une requête avec un TTL égal à la taille maximale d'une route. Cette optimisation prévient l'inondation inutile du réseau quand un des voisins est la destination recherchée ou qu'il possède la route adéquate.
- **recherche incrémentale (Expanding Ring Search) [31] :** le nœud commence par envoyer une requête de route *Non-propagating* avec un TTL égal à 1. Si le nœud ne reçoit pas de réponse, il double la valeur du TTL lors de sa requête suivante. Il poursuit ce processus jusqu'à ce qu'il reçoive une réponse de route. Avec cette technique, le nœud explore progressivement le réseau pour ne pas inonder systématiquement le réseau.

– **Récupération de route (Salvaging) :** le nœud détectant l'inaccessibilité du nœud suivant consulte son cache pour rechercher un autre chemin menant à la destination. S'il possède une route, il renvoie un message d'erreur vers la source en précisant la modification apportée au chemin et il transmet le paquet de données avec la nouvelle route. Sinon, le nœud peut lancer une découverte de route ciblée vers le nœud destination. Ainsi, on évite de repartir de la source.

– **Annonce de route erroné (Increased spreading of route error messages) :** lorsqu'une source reçoit un message d'erreur de route, elle le joint à sa nouvelle requête de route. Ainsi elle ne recevra pas de réponse contenant le chemin erroné.

– **Annonce de route erroné (Les paquets de données en attente) :** lorsqu'un nœud intermédiaire transmettant un paquet détecte que le lien de saut suivant sur la route est rompu. En plus, que ce nœud doit manipuler ce paquet tel que défini pour la maintenance de route. Le nœud doit manipuler les

paquets dans les files d'attente ou dans le tampon de maintenance de la même façon.

– **Raccourcissement automatique du route(Automatic route shortening) :** le mécanisme de raccourcissement automatique de route se base sur l'écoute des nœuds des paquets qui circulent dans le réseau. Si un nœud écoute un paquet destiné à un nœud donné et trouve que son adresse se trouve plus tard dans la séquence des nœuds relais, le nœud doit envoyer un *Gratuitous Route Reply* à la source du paquet en lui indiquant un chemin plus court qui peut être utilisé. La figure 1.9 illustre une situation qui nécessite un *Gratuitous Route Reply* de la part du nœud D puisqu'il arrive à entendre la transmission du noeud B vers C du paquet provenant du noeud A.

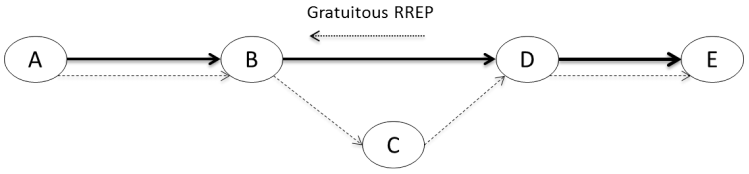


FIGURE 1.9 – Cas d'un automatic route shortening dans DSR

Backup source routing

Dans ce travail, les auteurs proposent un algorithme BSR (Backup Source Routing) [36]. BSR est une extension du protocole DSR, l'idée clé est de réduire le nombre d'opérations de découverte de routes après une rupture de lien. Pour cela les nœuds maintiennent le chemin principal le plus court mais également un chemin de secours plus stable. Ce protocole définit une nouvelle métrique de routage *Route durability*, et développe une méthode d'approximation pour mesurer la métrique par une fonction heuristique de coût. Le mécanisme de propagation utilisé par DSR doit être modifié puisqu'une grande quantité de messages dupliqués ont été éliminée. Ce qui rend difficile l'établissement des routes de secours au niveau des nœuds intermédiaires. Chaque nœud intermédiaire sélectionne deux routes, le

premier chemin est obtenu selon le premier paquet RREQ reçu (selon le délai), et un autre chemin plus stable est sélectionné. Lorsqu'un nœud intermédiaire reçoit un paquet RREQ dont la cache de ce nœud contient des routes de secoure vers la destination, une réponse RREP va être générée. Si le nœud destination reçoit plusieurs paquets RREQ, il sélectionne la route primaire ainsi que d'autres routes plus stables. En cas de défaillance de lien, un paquet RERR est envoyé contenant des informations de défaillance. Lorsque le nœud source reçoit le paquet RERR, il supprime les liens rompus et lance une reconstruction de routes de secours basant sur les informations à jours qui ont été reçues. Dans le pire des cas, une procédure de découverte de routes va être déclenchée.

Hash based dynamic source routing

L'utilisation de routage par la source permet d'éviter les boucles, en plus de cette façon seule la source doit maintenir une route cohérente vers la destination. Cependant, l'utilisation de routage par la source implique une surcharge importante par paquet de données. Dans ce travail l'auteur propose une extension du protocole DSR ; HB-DSR (Hash Based Dynamic Source Routing) [7]. L'idée de HB-DSR est de remplacer la route source par un vecteur de taille bien précis (bloom filter, contient des 0 et des 1) en utilisant le mécanisme de *bloom filter* (structure de données probabilistes). Dans HB-DSR, le nœud source calcule le bloom filter de la liste des adresses nœuds constituant la route source, insérer le bloom filter dans le paquet de données et envoie ce dernier vers la destination. Chaque nœud intermédiaire reçoit le paquet de données doit faire un teste sur la liste de tous ses voisins utilisant le même bloom filter. Un nœud voisin qui appartient au bloom filter est considéré comme un element de l'ensemble des nœuds constituant la route source. Dans le cas où plusieurs voisins appartiennent au bloom filter, le paquet de données doit être dupliqué et envoyé vers tous ces derniers. Si aucun nœud voisin appartient au bloom filter, le paquet doit être supprimé (paquet mal acheminé). Ce processus

est exécuté de bout en bout par l'ensemble des nœuds intermédiaires jusqu'à ce que le paquet de données atteigne la destination. Dans cette solution, les résultats montrent une amélioration en charge de contrôle par paquet en remplaçant le chemin source par le bloom filter. Cependant, le routage dans HB-DSR génère une charge supplémentaire, cela est dû à la structure probabiliste de bloom filtre (paquet mal acheminé), notamment dans le cas des réseaux de grande densité.

Efficient flooding using prefetching

Quoique le mécanisme du cache de routage dans DSR serve à réduire l'inondation de RREQ dans le réseau, mais la gestion de ces caches nécessite une stratégie efficace, notamment dans un environnement très dynamique. Afin de résoudre ce problème, Shobha et al. proposent une nouvelle technique de préchargement *Prefetching* (Efficient flooding using prefetching in on demand routing protocols for Manet) [8]. Les routes dans le cache sont marquées par un *Time-stamp* pendant la phase de découverte. L'obsolescence de routes stockées dans le cache peut être facilement enlevée. Le *Time-stamp* affecté à chaque entrée du cache est sélectionné à base de la mobilité du nœud et prend une valeur égale à la durée de pause du réseau. Le calcul des nouveaux chemins est lancé automatiquement juste avant que la validité des routes soit expirée, ce qui évite l'obsolescence des routes du cache. Seuls les chemins utilisés fréquemment sont préchargés avant que leurs *Time-stamp* soit expirée utilisant une inondation sélective.

Dynamic source routing algorithme for route link quality

Dans les réseaux mobiles ad hoc, la métrique de routage (nombre de saut) est la plus utilisée pour la sélection des chemins. Les protocoles de routage utilisant cette métrique arrivent à choisir aléatoirement un parmi plusieurs chemins de même longueur (nombre de sauts), par conséquent la grande différence entre les chemins en termes de débit effectif (throughput) a été ignorée. Les auteurs dans [32], proposent une solution (analysis the anhanced dynamic source routing algorithme for route

link quality) qui prend en considération la qualité de lien en termes d'ETT (Expected Transmission Time). Cela pour sélectionner les routes de haut throughput. La méthode aide à trouver un chemin avec le nombre minimum de retransmissions nécessaires pour délivrer un paquet en calculant le taux de perte des paquets entre les nœuds voisins. Afin de calculer ETT, un paquet *probe* est diffusé périodiquement entre chaque paire de nœuds. Chaque nœud maintient le nombre de paquets *probe* qui a été réçus pendant 10 secondes. Cette valeur a été envoyée dans un paquet *probe*. Ensuite chaque nœud calcule le taux de perte (le nombre prévu de retransmissions nécessaires pour la délivrance d'un paquet avec succès). Cette valeur peut être concidérée comme métrique de qualité de liens entre deux nœuds. Par conséquent le protocole de routage va choisir un chemin de routage avec le nombre minimum de retransmissions nécessaires à la délivrance du paquet. ETT est calculée à chaque fois qu'un nœud reçoit un paquet *probe*.

1.3.2 Changement de la topologie

La topologie des réseaux ad hoc change rapidement et aléatoirement, ceci est causé par la mobilité arbitraire des nœuds du réseau. Ces nœuds peuvent quitter le réseau ou des nouveaux nœuds peuvent le rejoindre. Ce changement de la topologie change les routes entre les nœuds et provoque la perte des paquets. Aussi la maintenance des routes dans les protocoles de routage traditionnels consomme énormément de bande passante, chargé le réseau et augmenté le temps de transmission des paquets. Plusieurs protocoles de routage réactifs ont été proposés pour contourner ce problème. Ces protocoles proposent des solutions pour diminuer l'impact d'une rupture de route sur la bande passante du réseau.

Plusieurs protocoles considèrent la stabilité de route comme le facteur le plus important pour sélectionner une route, ce qui réduit la probabilité de défaillance des liens. Si les chemins de qualité plus élevée sont choisis, ils sont susceptibles de vivre plus longtemps en réduisant le nombre de découvertes de route. Parmi ces

protocoles :

Associativity-Based Routing

Le protocole ABR (Associativity-Based Routing) [9] privilégie les nœuds les plus stables. Pour déterminer cette stabilité, chaque nœud émet périodiquement des messages de contrôles HELLO à ses voisins pour faire connaître sa présence. Un nœud calcule la stabilité d'un voisin en fonction du temps qu'il passe dans son voisinage. La phase de découverte de chemins dans ABR est semblable à celle dans DSR. Un nœud diffusait une requête BQ (Broadcast query) afin de trouver des nœuds qui mènent vers la destination. Un nœud intermédiaire ajoute leur adresse et leur valeur d'associativité au paquet BQ reçu. Chaque paquet BQ (Broadcast query) reçu par la destination contient les valeurs d'associativité de tous les nœuds constituant le chemin reliant la source et la destination. Le nœud destination choisit le meilleur chemin selon la valeur de degré d'associativité et envoie un message (BQ-reply) au nœud source. Dans ABR un nœud qui détecte la rupture d'un lien initie localement une découverte de chemins.

Long Life time Route Discovery

Dans le protocole (Long Life time Route Discovery) [10], Cheng et Heinzelman ont été proposé une solution consiste à choisir la route de plus long duré de vie. Deux algorithmes ont été proposés : g-LLR est l'algorithme global fonctionné d'une manière centralisé dont l'idée clé est de choisir le lien avec la plus longue durée de vie pour chaque paire des nœuds dans le réseau. Cet algorithme nécessite une connaissance globale sur la durée de vie de tous les liens du réseau, ça ce n'est pas pratique pour implémenter directement sur les protocoles de routage. Cependant le LLR optimal obtenu par cet algorithme peut être utilisé comme benchmark pour évaluer la performance des approches LLR distribués. D-LLR (LLR distribué) est un algorithme distribué qui sélectionne les routes de plus longues durées parmi les plus courts chemins (en longueur) utilisant des informations locales. D-LLR fournit

une découverte Best-Effort résultant deux routes LLR, un primaire LLR (est un LLR pour le plus court chemin) et un deuxième auxiliaire (qui est plus long d'un saut que le premier) dans la même procédure de découverte. La procédure générale de d-LLR consiste à utiliser un paquet LLR-Req qui contient toujours deux routes primaires et secondaires (au début contient seulement le nœud source). Au niveau de chaque nœud intermédiaire la durée de vie des deux routes a été recalculée. Le nœud destination choisissait les deux routes et envoyait un paquet de réponse. Cette solution peut implémenter sur les protocoles de routage comme extension sans toucher les autres couches, la structure des caches ou des tables de routage.
D'autres protocoles proposent des solutions utilisant la maintenance préventive de chemin, le coût de détection d'une fracture de chemin est éliminé, si un autre chemin se trouve avec succès avant les défaillances des chemins. En outre, le coût pour découvrir un autre chemin est réduit (ou éliminé), si la découverte du chemin initiée avant que le chemin courant est effectivement rompu. Plusieurs protocoles ont été proposés :

Preemptive routing in ad hoc networks

L'algorithme inventé par Goff et al. (Preemptive routing in ad hoc networks)[11] initialisé la récupération (maintenance du chemin) avant de détecter qu'un lien est susceptible de se rompre. Cet algorithme maintient la connectivité par le passage préventivement a un chemin de meilleure qualité lorsque la qualité du premier devient soupçonnée. Cette technique est similaire au soft-handoff (transfert intercellulaire sans coupure). Ainsi en combinant le meilleur des protocoles à la demande et des protocoles proactifs ; la surcharge est maintenue petite car les mises à jour ne déclenchent que par des chemins actifs qui sont susceptibles d'être rompus. Alors que les algorithmes à la demande minimisent la surcharge en initiant la découverte de route lorsque si nécessaire. L'algorithme se compose de deux éléments : (i) la détection d'un trajet qui est susceptible d'être déconnecté à bientôt. (ii) chercher le

meilleur chemin de commutation. Pour les protocoles à la demande en remplaçant la défaillance d'un chemin par la probabilité de défaillance comme mécanisme de déclenchement pour la découverte de route. Une composante essentielle de ce schème proposé, consiste à détecter lorsque la qualité d'un chemin n'est plus acceptable (ce qui génère une préemption d'avertissement). La qualité de trajet peut intégrer plusieurs critères tels que l'intensité du signal, taux de collision, etc.

Stable Weight Based On-demand Routing

Wang et al. propose le protocole (Stable Weight Based On-demand Routing) [12]. Dans ce protocole le poids de lien est calculé selon trois facteurs : temps d'expiration de route, compteur d'erreur, et compteur de sauts. La procédure de découverte de route est similaire au protocole DSR avec l'utilisation de numéro de séquence. Chaque nœud envoie un paquet RREQ à ses voisins avec les informations sur lesquelles le nœud destination sélectionné la route de plus grand poids. Un nœud peut calculer le temps d'expiration entre eux et le nœud émetteur utilisant des informations du système de positionnement global GPS. Le nœud destinataire choisit par la suite la route avec le plus grand poids et envoie un RREP vers le nœud source. Avec l'utilisation des informations de localisation GPS, on peut prédire que deux nœuds mobiles peuvent se déconnecter, ce fait qu'un paquet RERR est envoyé vers le nœud source. Ce dernier choisit une autre route s'il existe, sinon une nouvelle opération de découverte de route est initialisée.

D'autres protocoles ont été conçus principalement pour minimiser l'effet des changements de la topologie dans les réseaux ad hoc en utilisant des mécanismes de réparation efficaces de lien. Parmi ces protocoles :

Temporally-ordered routing algorithm

Le protocole TORA (Temporally-Ordered Routing Algorithm) [13] a été inventé par Vincent Park et M.Scott Corson. L'objectif principal de TORA est que les messages de contrôles sont limités à l'ensemble des nœuds proches au lieu du change-

ment de la topologie. Afin de s'adapter à la mobilité des réseaux ad hoc, le protocole stocke plusieurs chemins vers une même destination, ce qui fait que beaucoup de changements de topologie n'auront pas d'effets sur le routage des données, à moins que tous les chemins qui mènent vers la destination seront perdus (rompus). TORA avait trois types de messages : QRY (message de création de route), UPD (message pour la création et la maintenance de route), CLR (message de suppression de route). L'algorithme TORA appartient à la classe des algorithmes appelée *inversement de Liens* (Link Revesral). Il est basé sur le principe des algorithmes qui essaient de maintenir la propriété appelée *orientation destination* des graphes acycliques orientés (DAG : Directed Acyclic Graph). Un graphe acyclique orienté (DAG) est orientée destination s'il y a toujours un chemin possible vers une destination spécifiée. TORA utilise la notion de taille de nœud. Chaque nœud possède une taille qu'il échange avec l'ensemble de ses voisins directs. Cette nouvelle notion est utilisée dans l'orientation des liens du réseau. Le lien peut être un ascendant ou un aval basé sur la taille du nœud. Un lien est toujours orienté du nœud qui a la plus grande taille vers le nœud qui a la plus petite taille. Le processus de découverte de route pour une destination

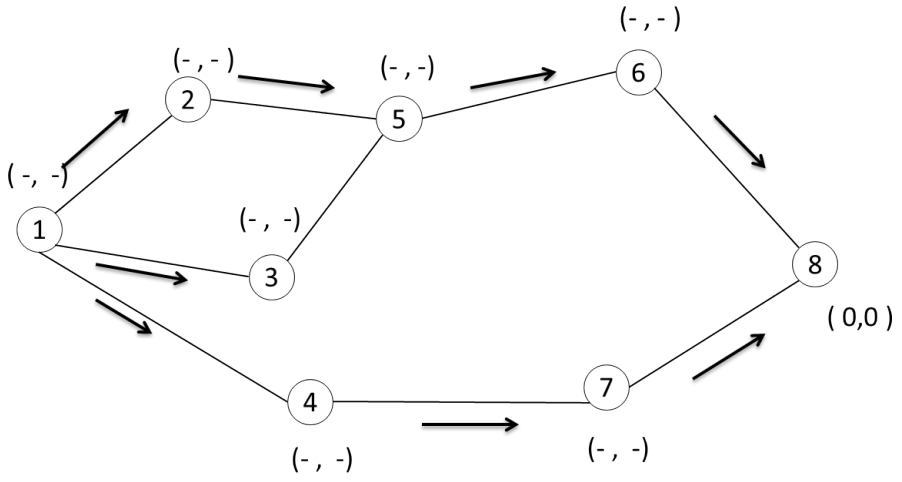


FIGURE 1.10 – La propagation du paquet de requête

créer un DAG initié par la source orienté vers une destination bien définie. Le nœud source diffuse un paquet QRY (cf. figure 1.10). Un nœud qui a reçu le paquet QRY

et a une taille indéfinie (ou un nœud qui n'a pas un lien sortant) rediffuse le paquet à ses voisins, un nœud qui a une taille définie répond par l'envoi d'un paquet UPD qui contient sa propre taille. Un nœud qui a reçu un paquet UPD, il affecte la valeur de taille contenant dans le paquet reçu incrémenté de *un* a sa propre taille, à condition que cette valeur soit la plus petite par rapport aux autres voisins (cf. figure 1.11). Le

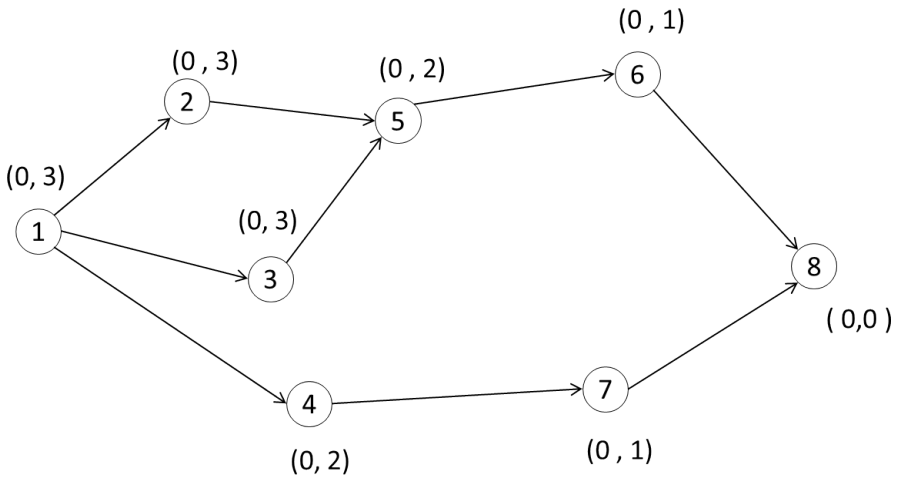


FIGURE 1.11 – Les tailles des noeuds après la réception du paquet UPD ((x,y) :le niveau de référence, la taille du noeud)

graphe DAG devient non orientée destination si un lien ou plus devient défaillant. Dans ce cas, l'algorithme utilise le concept *d'inversement de lien*. Ce concept assure la transformation du graphe précède en un graphe orienté destination durant un temps fini. Quand un nœud perd son dernier lien sortant à cause d'une défaillance, il lance un nouveau niveau de référence, cela est effectué comme suit : le nœud ajuste sa taille pour qu'elle représente le maximum de taille des nœuds voisins. Le nœud transmet par la suite un paquet UPD contenant la nouvelle taille. Par conséquent tous les liens vont être orientés de ce nœud vers ses voisins car la taille de nœud est devenue la plus grande taille. La diffusion du paquet UPD inverse le sens d'un ensemble de liens qui participent dans les chemins, où une défaillance est détectée : ce qui reconstruit le DAG et indique à la source l'invalidité des chemins rompus. La force principale du protocole de TORA est son approche à manipuler les échecs de

lien (cf. figure 1.12). La fonction de suppression du protocole TORA est effectuée en diffusant un paquet CLR dans le réseau, cela afin de supprimer les routes invalides.

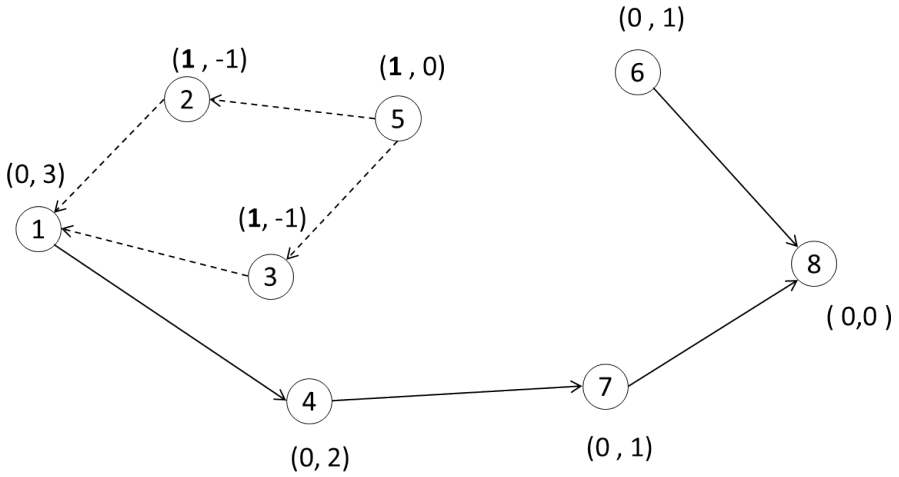


FIGURE 1.12 – La réaction du protocole TORA de la défaillance du lien (5,6)

Source Routing with Local Recovery

Sengul et Kravets proposent une solution de récupération locale de route sans besoin d'inonder le réseau. (Source routing with local recovery) [14] est une implémentation de *Bypass Routing*. En cas de défaillance de lien, une procédure de récupération locale (local recovery) a été lancée utilisant des informations sur l'état de lien de la couche MAC et des patches locaux sont créés contournant le lien rompu. Le nœud qui détecte la défaillance vérifiée dans sa propre cache locale s'il existe d'autres routes mènent à la destination. Si c'est le cas il envoie les paquets sur la route alternative (Salvage patch) (cf. figure 1.13 (1)), sinon un (bypass recovery) est initié enquêtant tous les voisins (vers la destination). En se basant sur les réponses de ces nœuds, la route sera reconstruite utilisant des informations de connectivité reçues (cf. figure 1.13 (2)). La couche MAC offrit les informations de connectivité pour les voisins à un saut et ces informations toujours mettent à jour en cas de communication par l'écoute de voisinages. Lorsque les paquets atteignent la destination, les nouvelles informations de route sont ajoutées à un paquet d'erreur renforcée. Ce

paquet est envoyé par la suite vers le nœud source indiquant la nouvelle route.

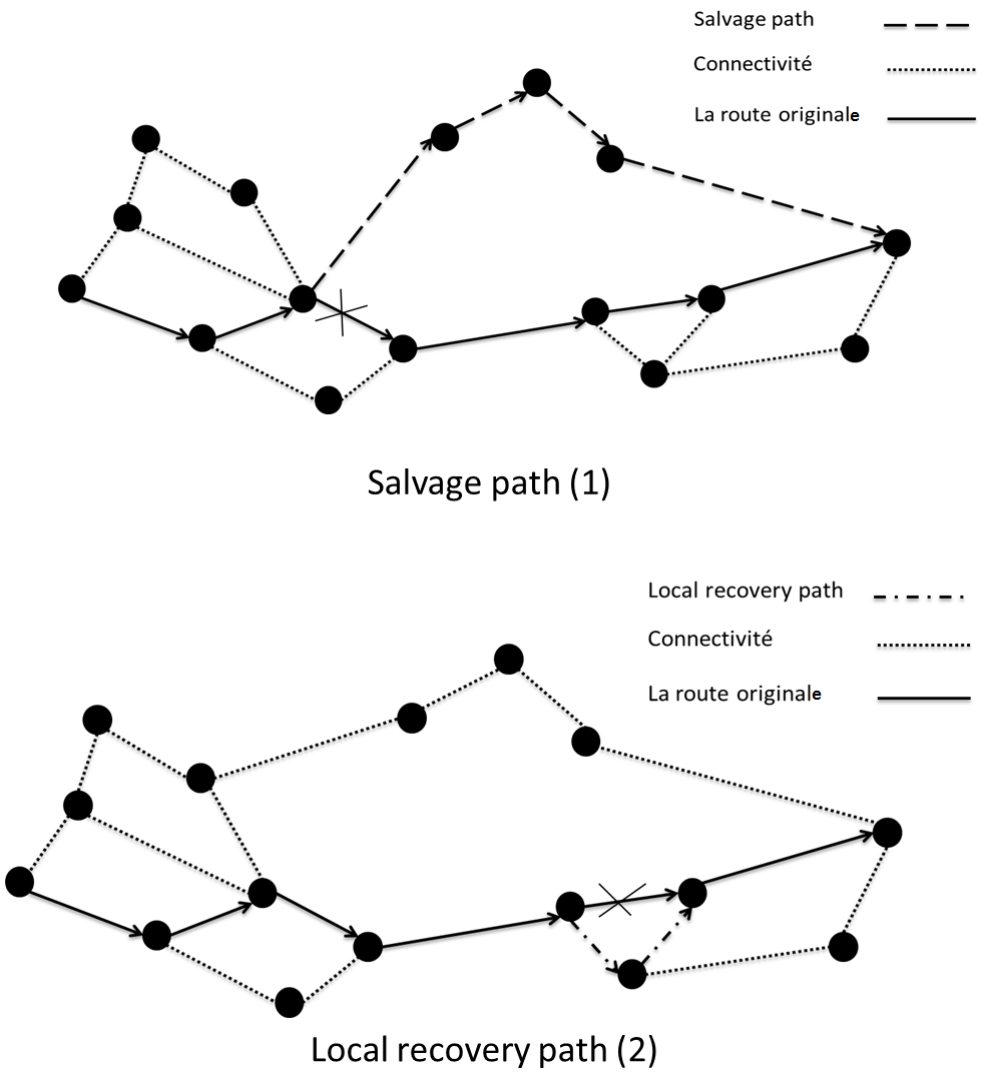
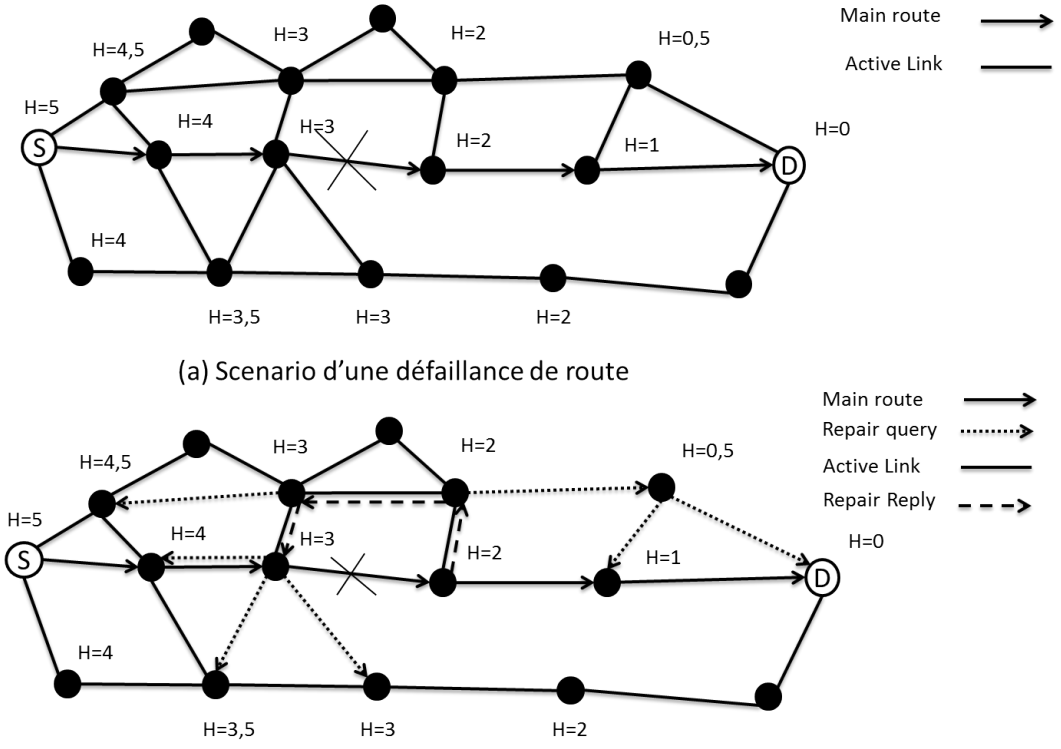


FIGURE 1.13 – Source Routing with Local Recovery

Low Overhead Dynamic Route Reparing Mecanism

Le protocole proposé par Yu et al. (Low overhead dynamic route reparing mecanism) [15] réparés dynamiquement les routes défaillantes utilisant des informations offertes par les nœuds (L'écoute sur la route de communication).Ce protocole remplaçait intelligemment les liens défaillants avec d'autres adjacents à la route principale. Pour la construction d'une route, le nœud source diffuse un paquet MREQ (Main

route Request), jusqu'au ce que le paquet atteigne la destination. Le nœud destination va répondu par un paquet MREP (Main Reply), le nombre de saut (utiliser ultérieurement pour la réparation de route) enregistrer dans MREP, (le nombre de saut de nœud destination est zéro $h = 0$). Lorsqu'il y a une défaillance, un paquet REPQ (Repair Query) a été envoyé contentant le nombre de saut du nœud qui détecte la défaillance de lien. Chaque nœud reçoit le paquet REPQ vérifié s'il a une route vers la destination. Si c'est le cas il envoie un paquet de réponse REPR vers le nœud qui envoie la requête de réparation (pour reconstruire la route), sinon il vérifiait le nombre de sauts du paquet REPQ avec son propre nombre de sauts. Si le nombre de saut de ce nœud est plus petit de ceux de REPQ, il propage le paquet vers la destination, sinon le paquet est supprimé afin que le paquet atteigne le nœud destination (cf. figure 1.14).



(a) Scenario d'une défaillance de route

(b) Scenario de réparation de route

FIGURE 1.14 – Illustration de procédure de réparation de route

1.3.3 Coût de la découverte de route

Les protocoles de routage réactifs créent et maintiennent les routes selon les besoins. Lorsque le réseau a besoin d'une route, une procédure de découverte globale de routes est lancée. Cette opération basée sur un mécanisme d'inondation résultant des problèmes de redondance, contention, et de collision dans le réseau ce qui augmente la surcharge de transmission. Dans cette direction de recherche, plusieurs protocoles de routage réactifs ont été proposés, visent à minimiser efficacement les inondations pendant la phase de découverte de route, ainsi que la surcharge de mise à jour durant le changement de la topologie utilisant plusieurs mécanismes (localisation, clustering, etc.).

Cluster based routing protocol

La solution proposée par Mingliang et al. est le protocole CBRP (Cluster Based Routing Protocol) [16]. Dans son fonctionnement de base, les nœuds sont divisés en groupes (Clusters). Un cluster est constitué d'un groupe de nœuds avec l'un d'eux élus comme chef cluster. Un cluster est identifié par l'ID de chef cluster. Les groupes sont formés comme suit : Quand un nœud entre dans le réseau entrer dans l'état indécis (C-UNDECIDED), déclenche un timer (u-timer) et envoie le message HELLO. Quand un chef de groupe reçoit ce message il répond avec un message HELLO immédiatement. Quand le nœud indécis reçoit ce message il passe son état en membre(C-MEMBER). Si le délai de garde du nœud indécis est expiré, alors il devient un chef de groupe si la table des voisins de nœud a au moins un lien bidirectionnel à un certain voisin. Autrement il reste dans l'état indécis et répète la procédure encore. Chaque nœud met à jour une table des voisins. Pour chaque voisin, la table des voisins d'un nœud contient le mode du lien (uni ou bidirectionnel) et l'état du voisin (chef cluster ou membre). Un chef cluster garde des informations sur les membres de son groupe et également met à jour une table d'adjacence du groupe qui contient des informations sur les clusters voisins. Pour chaque cluster voisin, une

entrée dans cette table contient la passerelle par laquelle le groupe peut être accédé (cf. figure 1.15). Le routage dans CBRP est basé sur le routage par la source, il

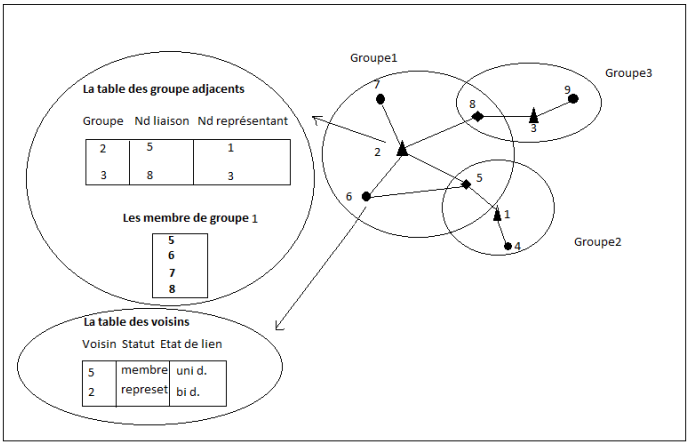


FIGURE 1.15 – L'organisation du réseaux dans CBRP

consiste à deux phases : découverte de route et routage réel des paquets. Quand le nœud source doit envoyer des données à la destination, il diffuse par inondation des paquets de demande de route seulement aux chefs de clusters voisins. À la réception de la demande un chef cluster vérifie si la destination est dans son groupe. Si oui, alors il envoie la demande directement à la destination, autrement il l'envoie à tous ses chefs de cluster adjacents. L'adresse de chef cluster est enregistrée dans un paquet ainsi un chef de groupe supprime un paquet de demande qu'il a déjà vu. Quand le nœud destination reçoit le paquet de demande, il répond avec la route qui a été enregistrée dans le paquet de demande. Si la source ne reçoit pas de réponse au cours d'une période de temps donnée, elle essaye d'envoyer la demande de route de nouveau.

Location Aided Routing

Le protocole LAR (Location Aided Routing) [17] a été proposé par Young-Bae Ko et al., il est similaire au DSR mais LAR utilisé les informations de localisation fournies par le système de positionnement global appelé GPS (Global Positioning

System). La caractéristique la plus importante de ce protocole est de limiter l'inondation des paquets de requête de route. Il utilise les informations de localisation pour prédire la localisation courante des nœuds destination. Deux approches sont proposées : Le premier schéma emploie directement les concepts de la *zone prévue* et la *zone de demande*. Dans ce cas, un nœud source utilise l'emplacement et les informations de la mobilité dont elle détient déjà sur le nœud destination pour prédire ce que l'on appelle la *zone prévue* de la destination. Cette zone prend la forme d'un cercle dont la destination est le centre et le rayon de la zone représente l'erreur de prédiction d'emplacement prévu à cause des erreurs GPS. Puis la source définit ce qu'on appelle la *zone de demande*. Cette zone est le plus petit rectangle qui comprend à la fois le nœud source et la zone de destination prévue. Lorsque le nœud source envoie le paquet de requête de route vers la destination, seuls les nœuds intermédiaires à l'intérieur de la zone de demande transmettent ce paquet, les autres nœuds jetés le paquet (cf. figure 1.16). Dans le second schéma, le nœud

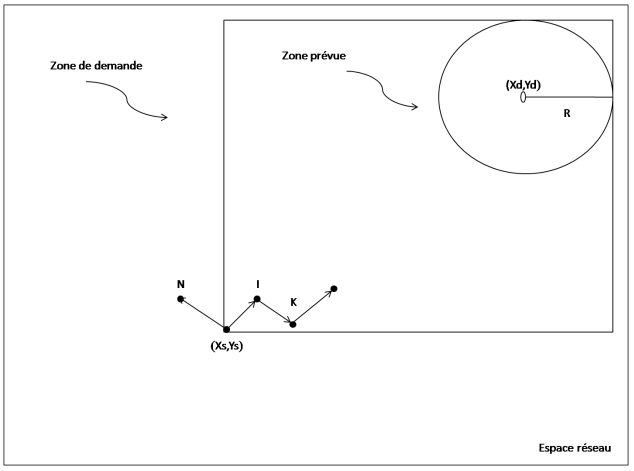


FIGURE 1.16 – Schéma 1 de LAR

source utilise l'emplacement disponible et les informations de la mobilité pour mesurer la distance jusqu'à l'emplacement de destination prévue. Ensuite il envoie un paquet de demande de route qui comprend cette distance. À la réception de paquet de requête de route, le nœud vérifié si la distance entre lui et le nœud destination

est inférieure à la valeur de distance incluse dans le paquet de demande de route. Si c'est le cas, il met à jour le paquet de sa propre distance et transmettre le paquet ; par ailleurs, le paquet est rejeté (cf. figure 1.17).

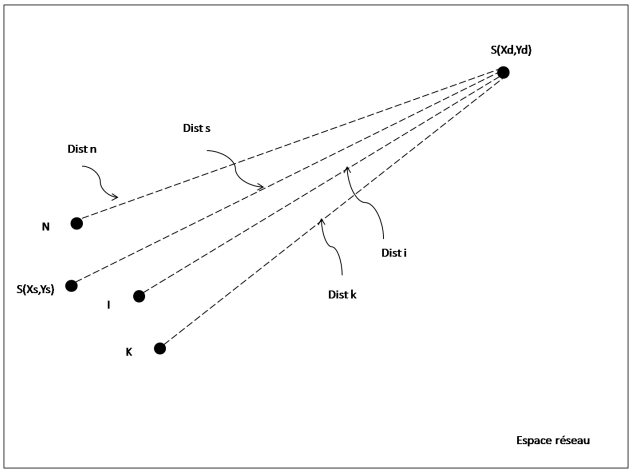


FIGURE 1.17 – Schéma 2 de LAR

Query localization techniques for on demand routing protocols

La technique de localisation de requête (Query localization) proposé dans [22], consiste à utiliser les anciennes informations de routage pour localiser l'inondation de requête dans une région limitée du réseau. Deux approches de localisation de requête ont été proposées : dans la première approche (Path locality) ; le protocole maintient un ensemble des nœuds (Pold) qui inclut les nœuds appartiennent à la dernière route valide. Dans le processus de découverte de route un paquet de requête est transmis dans le réseau. Si le paquet passe par un nœud qui n'appartient pas à l'ensemble (Pold), un compteur est incrémenté par *un*. Le paquet a été supprimé dès que la valeur du compteur dépasse une valeur prédéfinie k. La deuxième approche (Node locality) : ce mécanisme est similaire au premier, la seule différence est : si la requête passe par des nœuds appartient à l'ensemble (Pold), le compteur sera réinitialisé à *zéro*, sinon le compteur est incrémenté par un. Le paquet a été supprimé, dès que la valeur du compteur dépasse une valeur prédéfini k. Dans cette technique les routes défaillantes sont préférées (localiser la découverte de route dans une région limitée

du réseau) ; cela réduit la congestion et minimiser la surcharge. Cependant, ces liens peuvent être rompus à nouveau. Une optimisation a été proposée dans [21], afin de résoudre ce problème ainsi d'améliorer la technique de (Query localization).

Recycled path routing in mobile ad hoc networks

Eisbrener et al. présente une nouvelle stratégie de découverte de route RPR (Recycled path routing in mobile ad hoc networks) [18]. Ce protocole basé sur les routes expiré enregistré dans la cache de routage utilisant un mécanisme d'inondation contrôlée vers la destination sans besoin d'utiliser les informations de localisation. Dans RPR un paquet RREQ est envoyé sans inonder tous le réseau utilisant deux concepts (hot) et (cold). Chaque nœud reçoit le paquet RREQ, vérifié si le nœud destination a été apparu au moins dans l'une des routes expirées de sa propre cache, si c'est le cas, le nœud est considéré (hot) (un nœud plus proche à la destination est considéré plus chaud qu'un autre plus loin) et il rediffuse le RREQ, sinon le nœud est considéré (cold). Si le nœud est plus froid par rapport à un seuil le paquet RREQ a été supprimé. Ce qui minimise la zone d'inondation dans le réseau et tout à fait réduit la surcharge (redondance des RREQ).

The polarized gossip protocol for path discovery in manets

Dans ce travail Belardi propose le protocole (The polarized gossip protocol for path discovery in manets) [19] pour la découverte de route dont le nœud transmit le paquet avec des probabilités prédéfinies. Au contraire aux algorithmes (Gossip classic) qui transmit des messages avec la même probabilité. Dans ce travail la probabilité dépend à la localisation des nœuds et la distance entre eux. Chaque nœud (Polarizing node) a deux (Gossiping probabilities) : Pf (Forward probability) et Pb (Backward probability). Le message est transmis avec une probabilité Pf lorsque le nœud reçoit ce message à partir d'au moins un nœud qui est plus loin à la destination que le nœud lui-même. Dans le cas contraire, il transmet le message avec une probabilité Pb (cf. figure 1.18). La probabilité Pf est plus grande que la valeur

critique Pc, au contraire la probabilité Pb est plus petite. Le but de cette technique c'est que toutes les transmissions vers la mauvaise direction seront éteintes, tandis que les autres survivent.

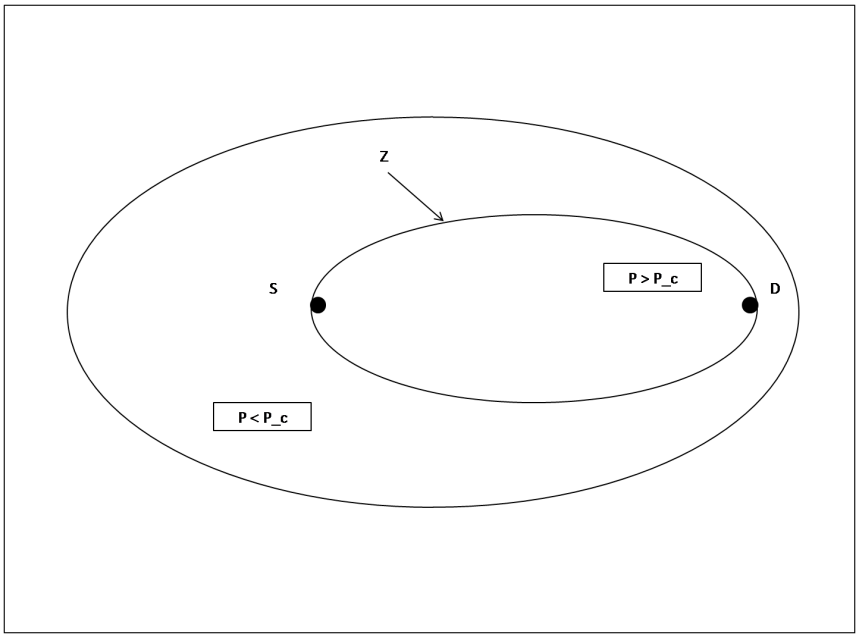


FIGURE 1.18 – Schéma de partition des nœuds dans le réseau

Adjusted probabilistic route discovery

Dans cette solution (Adjusted probabilistic route discovery) [20] ; les auteurs essayent de minimiser la surcharge dans le réseau réduisant la rediffusion des requêtes (RREQ). Deux méthodes probabilistes ont été utilisées afin de minimiser le nombre des paquets RREQ, utilisant une valeur de probabilité fixe et prédéfinie. Au contraire aux autres méthodes probabilistes, cette technique n'utilise pas des informations de localisation, mais basé sur les informations de la topologie. Dans le premier schéma (2P-scheme), deux groupes ont été défini basant sur des informations de voisinage. Si un nœud est situé dans une région clairsemée, il est attribué au groupe1, sinon si le nœud est situé dans une région dense, il est attribué au groupe2. Les nœuds dans le groupe1 sont alloués une plus grande probabilité de transmission par rapport au groupe2. Si un nœud reçoit un paquet RREQ, il génère une valeur aléatoire, et

comparait cette valeur avec la probabilité de transmission dans le paquet RREQ. Si la probabilité de transmission est plus grande que celle de la valeur aléatoire, le nœud rediffuse le RREQ sinon le paquet RREQ a été supprimé. Dans le deuxième schéma (3P-scheme) les nœuds sont classifiés en trois groupes. Les nœuds situés dans une région clairsemée, ils sont attribués au groupe1, les nœuds situés dans une région moyenne dense, ils sont attribués au groupe2, le groupe3 associé aux nœuds qui sont situés dans une région dense.

1.3.4 Mécanismes multidisciplinaire

Ces dernières années, des nouveaux algorithmes de routage ont été proposés utilisant des mécanismes multidisciplinaires (Intelligence artificielle, Théorie des jeux, Qos, etc.), pour les réseaux mobiles ad hoc. Ces algorithmes proposent des solutions afin de prendre en compte les nouvelles exigences des applications (utilisateurs) dans les réseaux MANETs. Ces techniques semblent donc appropriés pour gérer le routage dans les réseaux ad hoc et obtenir de meilleures performances. Il y a des protocoles issus du domaine d'intelligence artificielle, parmi ces protocoles :

The Ant-colony based routing algorithms

Guns et al. présentent une nouvelle technique ARA (the Ant-colony based routing algorithms) [23] pour le routage ad hoc utilisant des concepts de (Swarm intelligence) et des techniques d'optimisation par colonie de fourmis (ACO). ARA utilise des agents fourmis pour découvrir et maintenir les routes entre les nœuds du réseau. L'idée de base des algorithmes d'optimisation par colonie de fourmis est inspirée du comportement des fourmis réel quand elles recherchent la nourriture. Les fourmis réussissent toujours à trouver le chemin le plus court pour atteindre la nourriture. Ce comportement est utilisé dans ARA. (cf. figure 1.19). La phase de découverte des routes est la suivante : la création des nouvelles liaisons nécessite l'utilisation d'un forward ANT (FANT) et un backward ANT (BANT). FANT pour maintenir

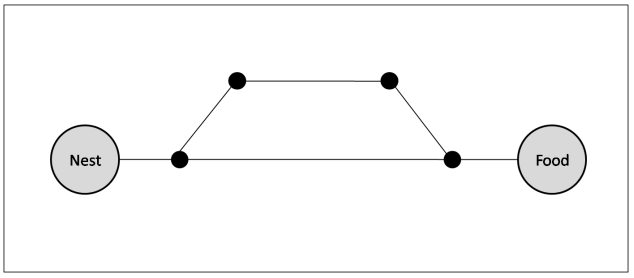


FIGURE 1.19 – Toute fourmi prend le plus court chemin après un temps

la phéromone vers le nœud destination. En revanche, une BANT pour maintenir la phéromone vers le nœud source. Le FANT est un petit paquet avec un numéro de séquence unique. Les nœuds sont capables de distingués les paquets dupliqués à base de numéro de séquence et l'adresse source de FANT. Un FANT est diffusé par l'émetteur et il sera relayé par les voisins de l'expéditeur (cf. figure 1.20). Un nœud reçoit un FANT pour la première fois, créée un enregistrement dans sa table de routage. Le nœud interprète l'adresse source de FANT comme adresse destination, l'adresse du nœud précède comme le saut suivant et calcule la valeur de phéromone en fonction du nombre de sauts nécessaires pour qu'un FANT atteignait le nœud. Ensuite, les nœuds relayés le FANT à ses voisins. Les FANT dupliqués sont détruits

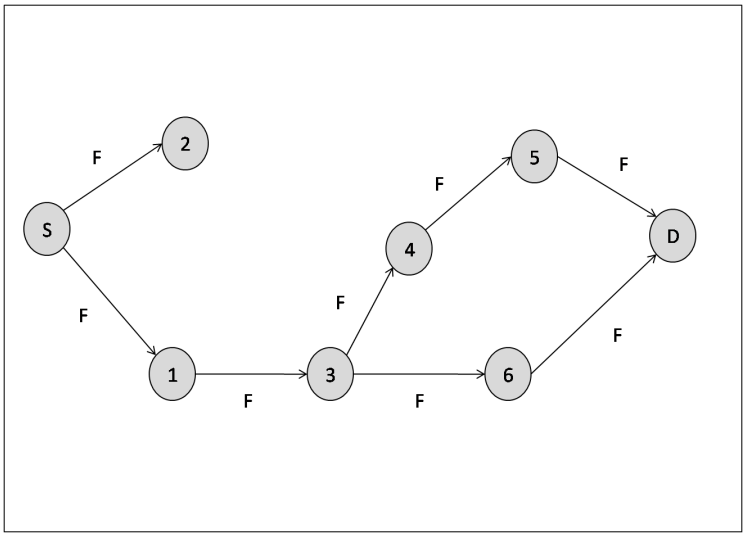


FIGURE 1.20 – Phase de découverte de route ; Le nœud source " S " envoi un FANT vers le nœud " D ", et il est relié par les autres nœuds

par les nœuds. Lorsque le paquet FANT atteint le nœud destination, il est traité d'une façon spécifique. Le nœud destination extrait les informations des FANT, et les détruit. Par la suite, il crée un BANT et l'envoi au nœud source (cf. figure 1.21). Le BANT effectue les mêmes tâches que le FANT. Lorsque le nœud source reçoit un BANT depuis le nœud destination, la route est établie et les paquets de données peuvent être envoyés. L'algorithme ARA n'a pas besoin à des paquets spécifiques

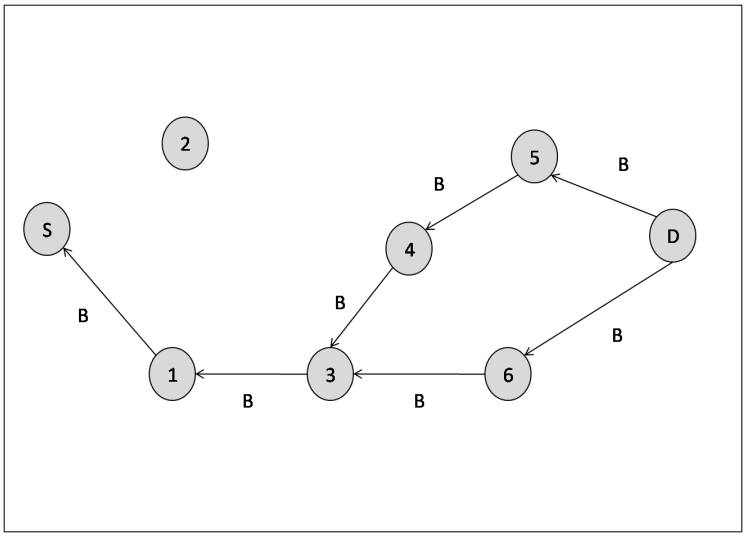


FIGURE 1.21 – Phase de découverte de route ; BANT envoyé par la destination vers le nœud source

pour la maintenance de route, car il utilise des paquets de données transmettent pour maintenir le trajet. La valeur de phéromone pour un trajet est augmentée à chaque fois le paquet de données est transmis le long du trajet mais diminuée avec le temps où aucun paquet n'est transmis. La surcharge de l'ARA est très faible car il n'y a pas des tables de routage qui sont échangées entre les nœuds contrairement à d'autres algorithmes de routage, le FANT et BANT ne transmettent pas beaucoup d'informations de routage, seulement le numéro de séquence. ARA besoin seulement des informations de l'entête IP des paquets de données.

Distributed Ant Routing

Rosati et al. propose un algorithme de routage distribué basée sur le comportement des colonies de fourmis (Distributed Ant Routing) [24]. Les auteurs visent à concevoir un algorithme intégrant les caractéristiques principales de nombreuses existantes approches. L'objectif principal de conception de DAR est de minimiser la complexité des calculs. Chaque nœud contient les tables de routage qui sont stochastiques avec le prochain saut étant choisi en fonction de probabilités pondérées. Ces probabilités sont calculées à base des traces de phéromone laissées par les fourmis. La transmission des fourmis servent à trouver de nouveaux chemins. Si plusieurs chemins sont disponibles à un nœud, le saut suivant peut-être sélectionné soit de manière aléatoire ou choisi le plus optimal. Les paquets de données sont transmis de façon déterministe en sélectionnant la plus forte probabilité à chaque nœud sur le chemin. Ainsi, un chemin global est créé à l'aide des informations hop by hop locale.

Game theoretic approach in routing protocol for manets

M. Naserian et al. propose une approche FDG (Forwarding Dilemma Game) [25] pour réduire le nombre des messages diffusés appliquant le mécanisme de théorie de jeu. Ce mécanisme utilisé comme modèle d'interaction entre utilisateurs pour éliminer les nœuds égoïstes et coordonner entre les nœuds dans le réseau ad hoc. Dans le FDG, les nœuds ce sont les joueurs du jeu. Un joueur avoir deux stratégies : (transmet le paquet) ou le (supprimer).Le FDG contient trois composants : (1) Nombre de joueurs N (Nombre de nœud qui reçoit les paquets diffusés), (2) coût de transfert, (3) facteur de gain de réseau. La stratégie de (Nash equilibrium) employé pour fournir la probabilité de transmission des messages d'inondation. Afin de permettre aux nœuds de découvrir le nombre des joueurs de FDG, un protocole de découvert de voisin (NDP) doit être introduit. Dans NDP les nœuds utilisent les messages d'accès au support ou les messages HELLO (comme FDG implémenté sur le protocole AODV, les messages HELLO sont utilisés). FDG limité le nombre des nœuds participant

à la découverte de route sans perturber la connectivité. Dans son fonctionnement de base : lorsque le nœud source diffuse le paquet de requête (également insérer le nombre de ses voisins). Lorsqu'un nœud intermédiaire va transmettre un paquet, il doit insérer le nombre et les ID de ses nœuds voisins dans le paquet de requête. Si un nœud intermédiaire reçoit un paquet de requête, il utilise ces informations pour calculer la probabilité de transmission, et il comparer cette valeur avec une valeur aléatoire générée, pour décider transmet ou supprimé le message (fonction d'utilité), ce qui minimise le nombre des messages de contrôle.

Les protocoles de routage meilleur effort, actuellement en place dans les réseaux MANETs, ne sont pas suffisants pour garantir une certaine qualité de service aux utilisateurs. Ces protocoles doivent évoluer et prendre en compte les contraintes imposées par les applications. Les protocoles de routage à QoS ont ainsi fait leur apparition dans les réseaux MANETs prenant en compte de nombreuses contraintes de QoS telles que le délai, la bande passante, la fiabilité, l'économie d'énergie, etc.

Ad hoc Qos on-demand routing in mobile ad hoc networks

Xue et Ganz propose AQOR (Ad hoc Qos On-demand Routing in mobile ad hoc networks) [26]. Ce protocole de routage réactif permettant le support de Qos en termes de bande passante et le délai de bout en bout. Le mécanisme AQOR estime la bande passante et le délai de bout en bout et utilisait ces paramètres pour prendre des décisions d'admission et de réservation de ressources. AQOR intègre la découverte de route à la demande, les fonctions de signalisations pour la réservation des ressources et le routage saut par saut pour fournir un support de qualité de service dans les réseaux ad hoc. Comme la plupart des violations de Qos sont détectés au niveau de la destination, la surcharge de routage généré peut être réduite déclenchant le processus de récupération de route au niveau de la destination. Les routes sont découvertes à la demande par un mécanisme d'inondation limité. Les valeurs de bande passante et le délai de bout en bout demandé sont exprimés à l'intérieur de paquet de requête.

Ces valeurs sont calculées à base de la capacité de liaison disponible et de la bande passante utilisée par le flux. Si cette demande est acceptée, le nœud met à jour sa table de routage avec statut (explored) et diffuse le paquet de requête à tous ses voisins. Toutefois, si aucune réponse n'est reçue dans un temps spécifié, l'entrée de la route dans la table sera supprimée et les réponses arrivant en retard sont tous simplement ignorés. Afin de réduire encore les messages de contrôles, les paquets ont un temps TTL (time-to-live) qui bloque les paquets de se déplacer inutilement à travers le réseau.

Qos routing with traffic distribution in mobile ad hoc networks

Les auteurs de [27] utilisent une épine dorsale de routage mobile (mobile routing backbone) pour le support Qos dans un MANET. L'épine dorsale de routage mobile (MRB) distribue de façon dynamique le trafic dans le réseau et sélectionne la route avec la meilleure qualité de service entre une paire source-destination. Le schéma proposé classifiait les nœuds de réseau en : les nœuds de routage Qos (QRN), les nœuds de routage simple (SRN), ou les nœuds émetteurs-récepteurs (TN). QRNs possèdent des garanties de qualité de service, SRNs simplement acheminer les paquets à travers le réseau, tandis que les TNs envoyée et recevoir des paquets mais ne peut pas les relier. MRB est formé par ces différents types de nœuds, alors qu'il n'est pas indispensable que tous les nœuds du réseau joignait le MRB. La classification des nœuds pour le MRB calculé selon les quatre métriques de support de Qos pour chaque paire de nœud : (1) capacité de ressource statique (calculé par taille des files d'attente des paquets, vitesse de l'unité centrale, la puissance de la batterie, et le maximum de la bande passante disponible), (2) disponibilité des ressources dynamiques (indique la charge courante de l'utilisation des ressources d'un nœud, le taux d'utilisation des ressources statiques sont utilisé pour calculer la disponibilité des ressources dynamiques), (3) qualité de voisinage (NQ)(le nombre de nœud qui peut transmettre les paquets avec succès), et (4) la qualité de la liaison et de la stabilité

(LQS) (la puissance de signal reçu et la stabilité statique de ces liens).Une fois que le MRB est mis en place, la découverte de route est lancée utilisant les RREQs le long de MRB.

Ant dtnamic source routing

Istikmal propose Ant-Dsr (une extension du protocole DSR) [35], un protocole basé sur le comportement des colonies de fourmis. DSR-Ant utilisait des agents fourmis pour la découverte de route utilisant la notion des phéromones afin de trouver le plus court chemin. Cela parce que les protocoles de routage réactif ne peuvent pas trouver le plus court chemin dans tous les cas (collision, etc.). Initialement m fourmis sont réparties aléatoirement sur les n nœuds du réseau. Pour chaque fourmi, une liste qui modélise la mémoire de la fourmi doit être défini. Initialement cette liste contient les nœud de départ. Les pistes de phéromone sont initialisées avec une petite constante positive. L'idée de cet algorithme est que chaque fourmi k placés sur le nœud i à l'instant t va choisir le nœud j de destination en fonction de la visibilité de ce nœud et de la quantité de phéromones déposés sur l'arc reliant les deux nœuds. À la fin d'un cycle (chaque fourmi a parcouru les nœuds du source à la destination), les variables de phéromones sont mis à jour (en d'autres termes, chaque fourmi se fait son tour en sens inverse tout en déposant les phéromones). On observe quelles fourmis a trouvées le tour de longueur minimum. Si ce trajet est le meilleur que le meilleur chemin qui a été trouvé jusqu'ici, on le mémorise et les mémoires des fourmis sont éffacés. Les fourmis recommencent un nouveau tour, toujours au départ de la ville sur laquelle ont été placés au début de l'algorithme. L'algorithme s'arrête après un nombre de cycles égale à une constance $Ncmax$, si à partir d'un insant toutes les fourmis font le même tour, l'algorithme s'interrompe. L'algorithme donne en retour le meilleur tour mémorisé. Les résultats obtebnus montrent une amélioration en temps de transmission et en débit effectif (throughput) par rapport à DSR, cependant l'algorithme Ant-Dsr génère une charge de contrôle supplémentaire afin

de trouver le bon chemin.

1.4 Discussion

La recherche dans le domaine des réseaux mobiles ad hoc est en plein essor. Plusieurs protocoles de routage ont été développés ces dernières années. Parmi les méthodes les plus utilisées pour la différenciation des protocoles de routage dans les MANETs, on retrouve celle basée sur la façon d'acquisition et du maintien d'informations de routage par les nœuds mobiles. En utilisant cette méthode, les protocoles de routage peuvent être classifiés en des protocoles proactifs, réactifs ou hybrides [29]. Dans ce mémoire on a mis l'accent sur les protocoles réactifs, ces protocoles se basant sur la découverte et la maintenance des routes ; suite à un besoin une procédure de découverte globale de routes est lancée. Les protocoles de routage réactifs présentés ci-avant partagent l'objectif commun de minimisation de consommation de la bande passante. Le trafic résiduel engendré par les informations de routage réduit la bande passante du réseau. De nombreuses méthodes ont été apportées pour réduire l'impact des informations de routage sur la bande passante du réseau.

Les protocoles de routage réactifs les plus connus (standardisé) sont : AODV [1] et DSR [2]. L'avantage de DSR est que les nœuds peuvent stocker des routes multiples dans leurs caches de routage ce qui signifie que le nœud source peut vérifier son cache de routage pour une route valide avant de lancer une nouvelle découverte de route. Ceci est très bénéfique dans le réseau avec une faible mobilité car les routes stocker dans la cache de routage seront valables pour une période plus long. Un autre avantage de DSR est qu'il ne nécessite pas l'inondation périodique des messages HELLO, cela économise une quantité considérable de la bande passante dans le réseau [30]. Avec DSR les nœuds de transit n'ont pas besoin de maintenir les informations de mise à jour pour envoyer les paquets de données, puisque ces derniers contiennent tous les informations de routage. Cependant, la taille de message devient assis grand à cause de l'inclusion des adresses des nœuds, ce qui rend DSR n'est pas

efficace pour les grands réseaux. Plusieurs protocoles ont été apportés, améliorant le protocole DSR. Pour adapter le protocole DSR au changement de la topologie, un protocole [36] BSR (Backup source routing) est développée, l'idée clé est de réduire le nombre d'opérations de découverte de route après une rupture de lien. Dans [7], les auteurs proposent une extension du protocole DSR afin de réduire la charge de contrôle par paquet. Comme la gestion des caches de routage nécessite une stratégie efficace notamment dans un environnement très dynamique. Afin de résoudre ce problème, Shobha et al propose une extension de DSR dans [8] en utilisant une nouvelle technique de préchargement (Prefetching). D'autres travaux ont été proposé dernièrement afin d'améliorer les performances de cache de routage ; la solution proposée dans [33] consiste à la diffusion d'un paquet parcourant le réseau afin de collecter les informations de la topologie ; chaque nœud du réseau peut mettre à jour son propre cache à l'aide de ces informations. Cela peut réduire le nombre des opérations de découverte déclenchés ainsi que le taux de perte. Dans [34] le protocole ASOR (Active Source and Opportunistic Routing scheme) a été proposé afin d'éviter le problème du cache vicié et de minimiser le temps de reconstruction des routes, cela par la réparation locale des chemins en cas de défaillance de la liaison entre les nœuds, le routage opportuniste utilisé afin d'atteindre la destination et le paramètre de durée de vie résiduelle de lien est considéré en reconstruisant la route.

AODV à un grand avantage concernant la surcharge des messages par rapport aux protocoles simples qui doivent garder la route entière du nœud source au nœud destination dans leurs messages. AODV agit assez rapidement aux changements topologiques du réseau, et de mettre à jour seulement les hôtes qui peuvent être affectés par le changement utilisant le message de RRER. Les messages HELLO qui sont responsables de la maintenance de route, sont également limités de sorte qu'ils ne créent pas une charge inutile dans le réseau. Cependant, le nœud peut subir des retards importants durant la construction et une défaillance de la liaison peut engager une autre découverte de route, ce qui introduit des retards supplémentaires et

consomme plus de bande passante si la taille du réseau augmente. Plusieurs travaux ont été proposés, comme extension de protocole AODV.

AOMDV [4] est une extension du protocole AODV. L'idée est de trouver multiples routes (sans boucles et lien disjoints) vers la même destination. Dans [5], SCATR un Framework de routage construit sur le protocole AODV, qui prend en considération la possibilité de connectivité intermittente utilisant un mécanisme de proxy. Une autre solution a été proposée par Subbiah et Ramesh dans [6], qui rend la reconstruction de routes préventives afin de réduire la charge produite due à la détection des défaillances des liens et de la maintenance des routes.

Dans les réseaux mobiles ad hoc, la mobilité des nœuds peut se produire des défaillances de liens et changer la topologie. Ce changement de la topologie rend les routes contenant ces liens invalides et provoque la perte des paquets. Aussi la maintenance des routes dans les protocoles de routage traditionnels consomme énormément de bande passante, chargé le réseau et augmenté le temps de transmission des paquets. Plusieurs solutions ont été proposées pour diminuer l'impact d'une rupture de route sur la bande passante du réseau. Des protocoles de routages réactifs basés sur la stabilité des liens ont été proposés afin de minimiser la probabilité qu'un lien puisse rompu, comme le protocole ABR [9] qui privilèges les liens les plus stables, et le protocole LLR [10] qui préfère les routes de plus longue durée de vie. D'autres solutions utilisant la maintenance préventive des routes ont été développé, sert à minimiser le cout de détection d'une fracture, si un autre chemin se trouve avec succès avant la défaillance des chemins. Une solution proposée dans [11] basant sur la qualité des liens pour prédire qu'il y a une défaillance de lien. SWORP [12] utilise les informations de localisation GPS pour détecter que deux nœuds peuvent se déconnecter. Plusieurs protocoles optés pour la réparation efficace des liens rompus en utilisant plusieurs mécanismes. Le protocole TORA [13] basé sur le mécanisme d'inversement de lien utilisant des graphes acycliques orientés (DAG). Dans SLR [14] une solution a été proposée pour la récupération locale des routes utilisant (Bybass

routing). Les auteurs du protocole (Low overhead dynamic route reparing mecanism) proposent une réparation simple des liens rompus basant sur les informations de voisinage.

D'autres protocoles ont été développés afin de réduire efficacement les inondations pendant la phase de découverte de route, ce qui minimise les redondances des collisions, et la surcharge. Les auteurs du protocole CBRP [16] proposent une solution basée sur le clustering afin de réduire le nombre de nœuds concernés par les informations de routage. Dans RPR [18] une nouvelle stratégie de découverte de route a été proposée basé sur les routes expirées dans les caches de routage. Le protocole LAR [17] limité la zone d'inondation utilisant les informations de localisation GPS. Le travail proposé dans [19] propose un (Gossip protocol). D'autres protocoles réduisent la charge de découverte de route sans besoin des informations de localisation comme les protocoles (Adjusted probabilistic route discovery) [20] et (Query localisation) [21].

Afin de gérer le routage dans les réseaux ad hoc et obtenir de meilleures performances. Plusieurs protocoles de routage ont été proposés issus des domaines différents. Des protocoles basés sur des techniques d'optimisation par colonie de fourmis comme ARA [23], DAR [24] et Ant-Dsr [35], d'autres basés sur le mécanisme de théorie de jeux comme FDJ [25].Les protocoles AQOR [26] et QMRB [27] sont des protocoles réactifs permettant le support de Qos.

1.5 Conclusion

Les protocoles de routage doivent s'adapter aux contraintes spécifiques des réseaux ad hoc comme la limitation de la bande passante. Dans ce chapitre nous avons réalisé un état de l'art, on se focalisant sur les protocoles réactifs dont ils sont plus adaptés aux caractéristiques des réseaux ad hoc. L'ensemble des protocoles présentés dans ce chapitre, illustré les principales techniques proposées pour diminuer la consommation de la bande passante tout en cherchant les chemins optimaux. Natu-

rellement, Il n'y a pas un seul bon protocole de routage pour tous les scénarios, car tous les protocoles sont basés sur des hypothèses spécifiques. En outre, les performances d'un protocole de routage dépendent fortement de plusieurs facteurs comme la mobilité des noeuds, le trafic de données, la taille du réseau, etc. Bien que ces protocoles résoudre beaucoup de problèmes de routages dans les réseaux ad hoc mais cette problématique reste présente de nombreux challenges et constitue un sujet d'étude très ouvertes.

2

Le Protocole DS2R2P

2.1 Introduction

De nombreux facteurs influencent la consommation de la bande passante d'un réseau. En effet, les informations de routage jouent un rôle important dans la diminution de la bande passante. Pour déterminer une route entre un nœud source et un nœud destination, les nœuds du réseau ont besoin d'informations de routage pour trouver un tel chemin. Les informations de routage consomment de la bande passante réduisant de ce fait la bande passante utile pour le fonctionnement du réseau. Cette diminution réduit la quantité de données transmises sur le réseau et augmente le risque de collisions. Dans la technique de routage par la source comme DSR, les nœuds intermédiaires n'ont pas besoin de maintenir les informations de mise à jour pour envoyer les paquets de données, puisque ces données contiennent toutes les informations du routage. Malgré cela, la charge de communication est très élevée du fait que l'émetteur des paquets embarque la séquence complète des

adresses des nœuds par lesquels doivent passer les paquets pour atteindre la destination. Le chemin complet entre la source et la destination est stocké dans l'entête de chaque paquet de données. Ainsi les nœuds intermédiaires relaient le paquet selon la route indiquée, ce qui implique une surcharge importante. Par conséquent, la taille de message devient assez grande, ce qui rend DSR inefficace pour les réseaux à grand échelle.

Dans ce chapitre, nous présentons notre protocole DS2R2P (on Demand Source Routing with Reduced Packets Protocol). L'objectif était de minimiser la charge de contrôle lors de l'acheminement des données. DS2R2P réduit la taille des paquets de données en utilisant un mécanisme de hachage ce qui réduit la charge de transmission et réduit le temps de réponse.

2.2 Le protocole DS2R2P

Dans son fonctionnement de base, le protocole DS2R2P similaire au protocole DSR. Dans ce dernier, le chemin complet entre la source et la destination est présenté dans l'entête de chaque paquet de données. Cependant, dans notre protocole le chemin entre la source et la destination est résumé par une valeur de taille fixe en utilisant un mécanisme de hachage, ce qui rend la taille d'un paquet de données plus réduite.

2.2.1 Les paquets et les structures de données utilisées dans DS2R2P

Les nœuds mobiles échangent quatre types de paquets de contrôle, à savoir : les paquets de requête RREQ, les paquets de réponse RREP et les paquets des erreurs RERR. Le même format des paquets RREQ, RREP et RERR définit pour le protocole DSR est adopté dans DS2R2P. Le format du paquet de données (pendant la phase de routage) est modifié, remplaçant les adresses du chemin du nœud source vers le nœud destinataire par une valeur de taille fixe et réduite obtenue à travers un

mécanisme de hachage. Pour les caches et les tables des requêtes, aucune modification n'a été faite sur le format défini dans DSR. Nous proposons aussi une nouvelle structure de données (table des chemins hachés) au niveau de chaque nœud afin de garder la trace des chemins de ces nœuds vers les différentes destinations.

Paquet de routage

Le paquet de routage DS2R2P est similaire à celui de DSR. Cependant, la liste d'adresses des nœuds constituant le chemin de routage est remplacée par une valeur de taille fixe (résultat du hachage du chemin original). Nous illustrons sur la figure 2.1 le format du paquet DS2R2P.

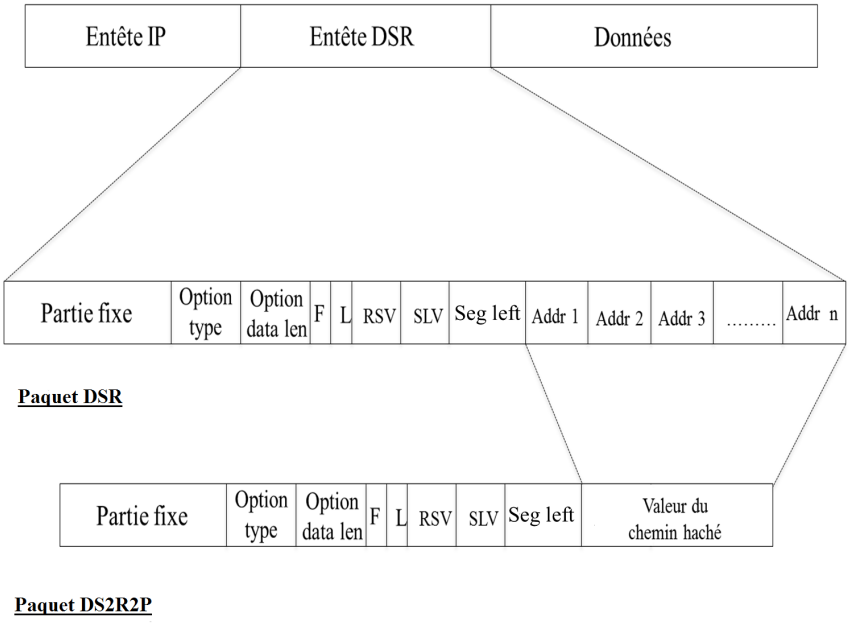


FIGURE 2.1 – Format du paquet de routage DSR vs DS2R2P

Table des chemins

C'est une structure servant à stocker les résultats de hachage calculés durant la phase de découverte, ainsi que les chemins correspondants. Cette table est indexée par la valeur de hachage des routes. L'entrée de cette table est constituée de trois champs : (1) *l'index*, qui représente la valeur de hachage (résultat obtenu en hachant

le chemin de routage), (2) *Destination* qui représente l'adresse du nœud destinataire et (3) *Chemin vers la destination*, qui représente le chemin dans son format original.

2.2.2 Fonction de hachage

Une fonction de hachage une fonction particulière qui à partir d'une donnée en entrée calcule une empreinte servant à identifier rapidement, bien qu'incomplètement, la donnée initiale. Une fonction de hachage est typiquement une fonction qui pour un ensemble de très grandes tailles renvoie des résultats aux spécifications précises optimisés pour des applications particulières. Cette sortie a une taille fixe qui varie selon les algorithmes. Comme une empreinte à une taille fixe et que celle-ci est inférieure à la taille des messages que l'on peut mettre à l'entrée de la fonction, alors il existe nécessairement des messages qui génèrent la même empreinte. Avec des entrées dont la taille n'est pas limitée, il existe une infinité de collisions. L'éfficacité de la fonction de hachage repose en partie sur la difficulté de trouver de tels conflits en un temps raisonnable.

Dans DSR, les adresses qui composent la route dans le paquet de données sont codées sur 32 bits pour IPV4 et 128 bits pour IPV6. Ainsi, afin de réduire la taille de l'entête, nous définissons la fonction de hachage suivante :

$$H(a_1, a_2,\ ...,\ a_\ell) = ((\sum_{i=1}^{\ell} a_i * c^{i \times e}) \bmod p) \bmod \alpha. \quad (1)$$

Tels que a_i est l'ensemble des identités des nœuds intermédiaires, ℓ est la longueur de la route, c et e sont des paramètres à fixer afin de minimiser le risque des collisions. p doit être premier et supérieur à α pour éviter le problème des multiples. α représente le taux de réduction qui est un paramètre très important à fixer selon les besoins de l'application. Comparant à DSR qui a une taille de l'entete du chemin égale à $TailleAdresseIP \times \ell$ bits, DS2R2P réduit cette valeur à $\log_2(\alpha)$ bits comportant l'information complète pour atteindre la destination. La taille du paquet de donnée $|Packet|$ est composé de deux parties : la taille de données $|Data|$ et la taille de l'entete $|Header|$, tel que pour DSR $|DSRHeader| = \ell \times TailleAdresseIP$ (la taille du

chemin complet menant à la destination). Cet entête se réduit de $TailleAdresseIP$ bits en passant d'un nœud à un autre. Pour DS2R2P : $|DS2R2PHeader| = \log_2(\alpha)$. On peut calculer le temps de transmission d'un paquet de données selon le protocole DSR avec la formule :

$$TT = \sum_{i=1}^{\ell-1} \frac{|Data|+|DSRHeader|-i*\frac{|DSRHeader|}{\ell}}{Débit}$$

On peut calculer le temps de transmission d'un paquet de données selon le protocole DS2R2P avec la formule :

$$TT = (\ell - 1)\frac{|Data|+|DS2R2PHeader|}{Débit}$$

Nous évaluons le temps de transmission de données en fonction de la longueur du chemin de routage (ℓ). Les figures 2.2 et 2.3 illustrent les résultats du temps de transmission en fonction de la longueur du chemin de routage pour taille du haché = 8 bits et taille du haché = 32 bits, respectivement.

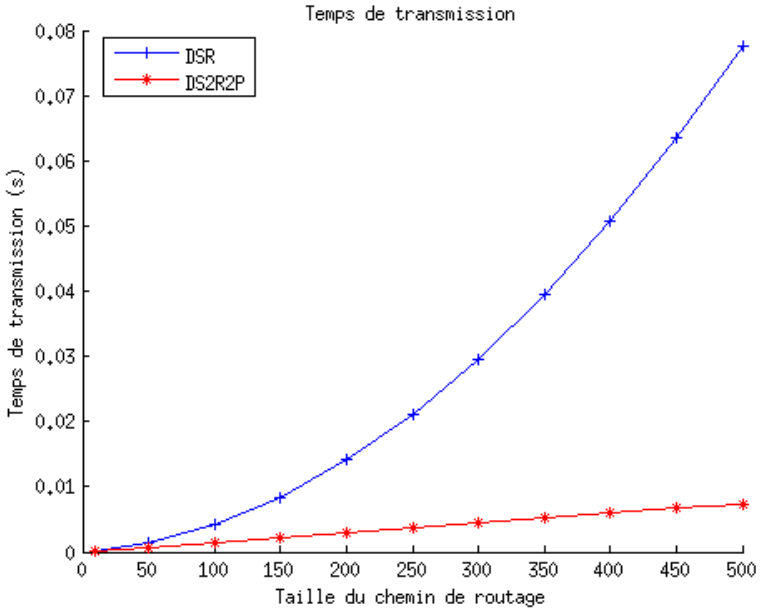


FIGURE 2.2 – Temps moyen de transmission en fonction de ℓ (DB=54MB/s, TailleAdresse = 4 Octets, Data=100 Octets, $\log_2(\alpha) = 1$ Octets)

Le temps moyen de transmission synthétise l'avantage de notre protocole en le comparant avec DSR, où l'écart de performance en termes de temps de transmission de données est largement grand. En plus, nous constatons d'après les résultats que

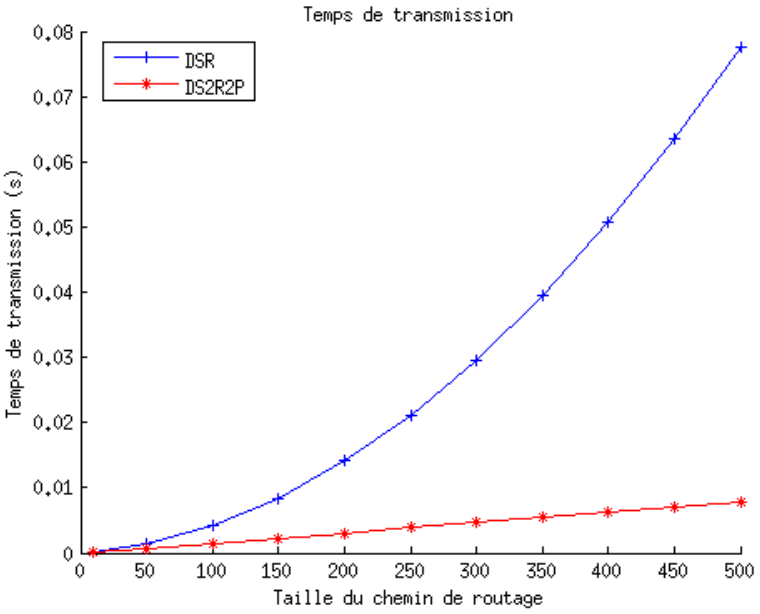


FIGURE 2.3 – Temps moyen de transmission en fonction de ℓ (DB=54MB/s, TailleAdresse = 4 Octets, Data=100 Octets, $\log_2(\alpha) = 4$ Octets)

la longueur du chemin de routage a un impact très faible sur les performances de DS2R2P.

2.2.3 Description du protocole DS2R2P

Le protocole DS2R2P s'exécute en trois phases : la découverte des routes, la maintenance et le routage des paquets de données. Nous décrivons chaque phase dans les sous-sections suivantes.

La découverte de route

L'opération de découverte de route dans DS2R2P est similaire à celle du DSR, une source cherche une route (diffuse un paquet RREQ) uniquement si elle veut émettre un paquet vers un nœud destinataire, et qu'elle ne possède aucune route qui mène vers celui-ci dans son cache. À la réception du paquet RREQ, le nœud destinataire applique une fonction de hachage sur sa propre adresse et la stocke dans sa table des chemins hachés. Dans le champ *index* est affecté le haché, dans

le champ *destination* est affectée l'adresse du nœud destinataire (dans ce cas c'est l'adresse du nœud lui-même) et dans le champ *chemin vers la destination* est affecté l'adresse du nœud destinataire (cf. algorithme 1). Ensuite, il répond avec un paquet RREP au nœud source en utilisant la route inverse. Chaque nœud intermédiaire en recevant le paquet RREP applique une fonction de hachage sur le chemin de ce nœud vers la destination et le stocke dans sa propre table, jusqu'à ce que le paquet RREP atteigne le nœud source (cf. algorithme 2).

Algorithm 1 Algorithme de traitement d'un paquet RREQ

```
1: Begin
2:     IF (Addr-node ≠ RREQ.Dest)
3:         IF ((Source-Adress, Request-id ∈ Route-request-table) OR
4:           (Addr-node ∈ RREQ.route))
5:           Discard(RREQ) ;
6:         ELSE
7:           Insert-req-table (RREQ.src, RREQ.route, RREQ.seq, RREQ.R-len) ;
8:           Add-node (RREQ.route, Addr-node) ;
9:           Broadcast(RREQ) ;
10:        EndIf
11:    ELSE
12:      RREP.route = RREQ.route ;
13:      Unicast (RREP) ;
14:      Insert-Hach-route-table (Hach(Addr-node), Addr-node, Addr-node) ;
15:    EndIf
16: END.
```

Algorithm 2 Algorithme de traitement d'un paquet RREP

```
1: Begin
2:     IF (Addr-node ≠ RREP.dest)
3:       Route1 = Route-current-to-Dest (RREP.route) ;
4:       Insert-Hach-route-table (Hach(Route1), RREP.dest, Route1) ;
5:       Unicast (RREP) ;
6:     ELSE
7:       Route-cache-insert (RREP.src, RREP.route) ;
8:       Insert-Hach-route-table (Hach(RREP.route), RREP.dest, RREP.route) ;
9:       Discard (RREP) ;
10:    EndIf
11: END.
```

Dans le cas où les liens sont unidirectionnels, le nœud destinataire envoie le paquet

RREP en utilisant un chemin de son cache s'il existe, sinon il lance une nouvelle découverte de routes. Lors de la réception du paquet RREP, le nœud source envoie seulement le premier paquet de données de la même façon que DSR. Cependant, il insère le chemin complet dans le premier paquet, applique une fonction de hachage sur la route et stocke le résultat dans la table des chemins hachés. Ensuite, il envoie le paquet au nœud destinataire. Chaque nœud intermédiaire recevant le paquet de données retire sa propre adresse du chemin de routage, applique une fonction de hachage sur le reste du chemin et stocke le résultat dans sa table ; ensuite il envoie le paquet vers le nœud suivant. Ce processus est exécuté de bout en bout par l'ensemble des nœuds intermédiaires jusqu'à ce que le paquet de données atteigne la destination. Les autres paquets de données seront envoyés par la suite en insérant la valeur du haché au lieu du chemin de routage.

La maintenance des routes

Comme un lien dans un chemin peut être rompu à cause de la mobilité, des fluctuations du canal de communication ou à cause des pannes des nœuds, il est nécessaire de mettre à jour les routes. Dans DS2R2P, si un nœud intermédiaire détecte la rupture d'un lien, il envoie un paquet RERR au nœud source à travers le chemin inverse (dans le cas des liens bidirectionnels). Les nœuds recevant le paquet RERR mettent à jour leurs caches ainsi que leurs tables des chemins hachés. Si le nœud source ne possède aucun chemin valide dans son cache, il initie le processus de découverte des routes (cf. algorithme 3). Dans le cas où les liens sont unidirectionnels, le nœud qui détecte la rupture de lien, envoie le paquet RERR en utilisant un chemin de son cache s'il existe, sinon par diffusion. Lors de la réception du paquet RERR, le nœud source met à jour son cache ainsi que sa table des chemins hachés et envoie un paquet RERR vers tous les nœuds constituant le chemin non valide afin que ces nœuds mettent à jour leurs caches ainsi que leurs tables.

Algorithm 3 Algorithme de traitement d'un paquet RERR

```
1: Begin
2:     Update current node cache ;
3:     IF (Address-node ≠ RERR.Dest)
4:           Unicast (RERR) ;
5:     ELSE
6:           Discard (RERR) ;
7:     EndIf
8: END.
```

Le routage

Algorithm 4 Algorithme de traitement d'un paquet de données

```
1: BEGIN
2: Source-node : Generating a packet with a routing header
3: Begin
4:     Route1 = Delate-home-adress (route-source) ;
5:     R-header.Hach-route = Hach (Route1) ;
6:     IP-source-Addr = Address-Source ;
7:     IP-Dest-Addr = Address-Dest ;
8:     Unicast (Packet) ;
9: End
10: Intermediate-node :
11: Begin
12:   IF (Address-node ≠ Paquet.Rout-header.IP-Dest-Addr)
13:     SearchTab (R-header.Hach-route, R-header.IP-Dest-Addr, R-header.Seg-
    left) ;
14:     Route1 = Delate-home-address (Hach-table.route) ;
15:     Routing-header.Hach-route = Hach (route1) ;
16:     Unicast (Packet) ;
17:   ELSE
18:     Get-data ;
19:     Discard (Packet) ;
20:   EndIf
21: End
22: END.
```

Dans le protocole DSR le nœud source envoie le paquet de données en ajoutant le chemin complet du nœud source vers le nœud destinataire. Cependant dans DS2R2P, le nœud source utilise un mécanisme afin de réduire la taille du paquet de données. Alors au lieu d'insérer le chemin complet dans les paquets des données, le nœud source retire sa propre adresse du chemin, hache le reste du chemin, insère

uniquement la valeur du haché et l'adresse destination dans les paquets de données et l'envoie vers le nœud suivant. Lors de la réception du paquet, le nœud récupère la valeur du haché, cherche dans sa table des chemins le sous-chemin qui correspond à la valeur du haché, à condition que le champ destination de la table doit correspondre à l'adresse du nœud destinataire et la taille du sous-chemin trouvé doit correspondre au nombre de sauts restants (la valeur du champ *seg-left* dans l'enete du paquet de données). Le nœud retire sa propre adresse du sous-chemin trouvé, calcule le haché du nouveau sous-chemin, remplace la valeur du haché par la nouvelle dans le paquet de données et l'envoie au nœud suivant. Ce processus est exécuté de bout en bout par l'ensemble des nœuds intermédiaires jusqu'à ce que le paquet de données atteigne la destination (cf. algorithme 4).

La figure 2.4 illustre un exemple d'une topologie de réseau. Nous supposons que le chemin choisi à la phase de découverte est $S \rightarrow B \rightarrow J \rightarrow F \rightarrow I \rightarrow D$. Ci-après la trace d'exécution du protocole DS2R2P.

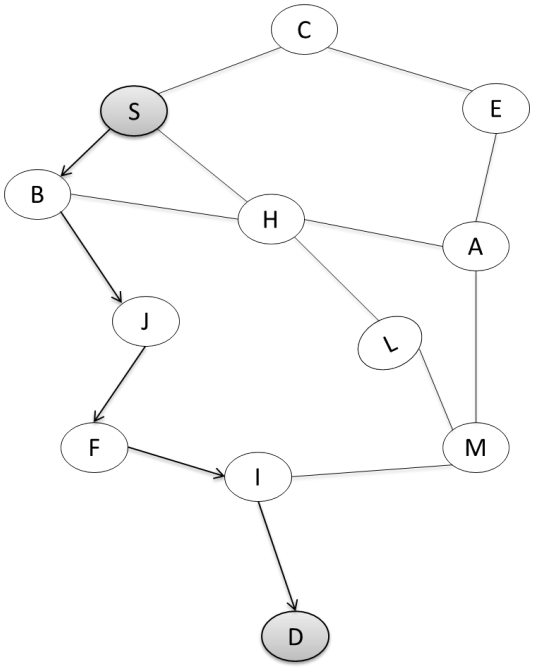


FIGURE 2.4 – Exemple d'exécution du protocole DS2R2P

1. À la réception du paquet $RREQ$, le nœud D répond avec un paquet $RREP$ contenant le chemin $S \rightarrow B \rightarrow J \rightarrow F \rightarrow I \rightarrow D$ vers le nœud S. Les traitements suivants sont effectués pendant la transmission du paquet $RREP$:
   - Le nœud D :
   - Calcule $H(D) = y_1$ et le stocke dans le champ *index* dans la table de hachage.
   - Destination $\leftarrow D$
   - Chemin vers la destination $\leftarrow D$.
   - Envoie le paquet $RREP$ vers le nœud I.
   - Le nœud I :
   - Calcule $H(I, D) = y_2$ et le stocke dans le champ *index* dans la table de hachage.
   - Destination $\leftarrow D$
   - Chemin vers la destination $\leftarrow I \rightarrow D$.
   - Envoie le paquet $RREP$ vers le nœud F.
   - Le nœud F :
   - Calcule $H(F, I, D) = y_3$ et le stocke dans le champ *index* dans la table de hachage.
   - Destination $\leftarrow D$
   - Chemin vers la destination $\leftarrow F \rightarrow I \rightarrow D$.
   - Envoie le paquet $RREP$ vers le nœud J.
   - Le nœud J :
   - Calcule $H(J, F, I, D) = y_4$ et le stocke dans le champ *index* dans la table de hachage.
   - Destination $\leftarrow D$
   - Chemin vers la destination $\leftarrow J \rightarrow F \rightarrow I \rightarrow D$.
   - Envoie le paquet $RREP$ vers le nœud B.
   - Le nœud B :
   - Calcule $H(B, J, F, I, D) = y_5$ et le stocke dans le champ *index* dans la table

de hachage.

– Destination $\leftarrow D$

– Chemin vers la destination $\leftarrow B \rightarrow J \rightarrow F \rightarrow I \rightarrow D$.

– Envoie le paquet $RREP$ vers le nœud S.

• Le nœud S :

– Calcule $H(S, B, J, F, I, D) = y_6$ et le stocke dans le champ *index* dans la table de hachage.

– Destination $\leftarrow D$

– Chemin vers la destination $\leftarrow S \rightarrow B \rightarrow J \rightarrow F \rightarrow I \rightarrow D$.

2. Lors du routage d'un paquet de données du nœud S vers le nœud D, les traitements suivants sont effectués :

• Le nœud S calcule $H(B, J, F, I, D) = y_5$, embarque cette valeur dans le paquet de données ainsi que l'adresse destination = D et l'envoie vers le nœud B.

• Lors de la réception de y_5, le nœud B :

– Cherche dans la table de hachage le chemin correspondant à *index*= y_5, destination= D, et le nombre de sauts du sous chemin = la valeur du seg-left = 4 (dans ce cas c'est $B \rightarrow J \rightarrow F \rightarrow I \rightarrow D$).

– Calcule $H(J, F, I, D) = y_4$, remplace y_5 par y_4 dans le paquet de données et l'envoie au nœud J.

• Lors de la réception de y_4, le nœud J :

– Cherche dans la table de hachage le chemin correspondant à *index*= y_4, destination= D, et le nombre de sauts du sous chemin = la valeur du seg-left = 3 (dans ce cas c'est $J \rightarrow F \rightarrow I \rightarrow D$).

– Calcule $H(F, I, D) = y_3$, remplace y_4 par y_3 dans le paquet de données et l'envoie au nœud F.

• Lors de la réception de y_3, le nœud J :

– Cherche dans la table de hachage le chemin correspondant à *index*= y_3,

destination= D, et le nombre de sauts du sous chemin = la valeur du seg-left = 2 (dans ce cas c'est $F \rightarrow I \rightarrow D$).

– Calcule $H(I, D) = y_2$, remplace y_3 par y_2 dans le paquet de données et l'envoie au nœud I.

• Lors de la réception de y_2, le nœud I :

– Cherche dans la table de hachage le chemin correspondant à *index*= y_2, destination= D, et le nombre de sauts du sous chemin = la valeur du seg-left = 1 (dans ce cas c'est $I \rightarrow D$).

– Calcule $H(D) = y_1$, remplace y_2 par y_1 dans le paquet de données et l'envoie au nœud D.

2.3 Conclusion

Dans DS2R2P, les données sont transmises en utilisant des paquets de tailles fixes et réduites, ce qui augmente les performances de routage en termes consommation de bande passante et aussi en termes de temps de réponse. Ces métriques sont très importantes si nous considérons un réseau ad hoc de taille importante. Nous considérons que le temps de calcul induit dans la fonction de hachage est négligeable par rapport au développement de la technologie actuelle. Quoique que les problèmes de collisions dans les fonctions de hachage puissent se produire générant des collisions dans la table des chemins. Ce problème peut être résolu en utilisant les informations dans le champ destination de la table des chemins hachés ainsi que celle dans le champ *seg-left* de l'entete de routage pour distinguer la bonne route parmi les chemins en collision.

3

Etude de performances

3.1 Introduction

L'objectif de ce chapitre est l'évaluation de performances de notre protocole DS2R2P par rapport au protocole DSR de base [2] et l'extension DSR (Automatic route shortening) (cf. page 20) qui est basé sur le principe de minimum de sauts, tel que le nouveau chemin avec moins de sauts est utilisé à la place du chemin actif. Nous commençons par la description de l'ensemble des paramètres de simulation ainsi que les différentes métriques de performances auxquelles nous sommes intéressés. Ensuite, nous présentons les résultats de comparaisons.

3.2 Environnement de simulation

Pour les simulations, nous avons utilisé l'environnement de programmation Matlab. Les simulations sont effectuées en comparant notre protocole aux protocoles DSR et l'extension DSR (Automatic route shortening). Le réseau est composé d'un nombre de nœuds répartis dans une zone de 1000 m^2. Les positions initiales des nœuds étant aléatoires sur la surface et ils se déplacent d'une façon arbitraire avec une vitesse de 1 m/s et une durée de pause de 60 s. Les nœuds sont configurés par un débit de 54 Mb/s. Pour chaque simulation, on sélectionne d'une manière aléatoire une source et une destination. Nous simulons l'acheminement des paquets de la source vers la destination pour les trois protocoles DSR, DS2R2P et l'extension DSR (Automatic route shortening). On considère la taille de données par paquet à 100 octets. Ce qui concerne le protocole DS2R2P, nous avons opté pour une valeur de $\alpha = 99$ (ce paramètre sera changé suivant la simulation effectuée). Nous avons mené des simulations avec des nœuds à :

- **Faible portée :** varié dans l'intervalle [55m, 75m].
- **Portée moyenne :** varié dans l'intervalle [125m, 200m].
- **Haute portée :** varié dans l'intervalle [350m, 400m].

Nous nous sommes intéressés à mesurer les métriques de performances suivantes :

- **Temps de transmission :** la moyenne de différence entre le temps d'arrivée d'un paquet de données à sa destination et le temps de son émission par le nœud source, pour tous les paquets de données reçu dans le réseau.
- **Overhead :** la charge de contrôle généré lors de l'acheminement des données. On s'intéresse à la charge de contrôle par paquet de données (la charge générée par les paquets RREQ, RREP et RERR est identique pour les trois protocoles).
- **Efficacité :** le rapport entre la quantité des données utiles délivrée sur la quantité totale des données envoyées (données et informations de routage), elle est définie : $\frac{Données}{Données+ChargeContrôle}$.

3.3 Les résultats de comparaison

3.3.1 Temps de transmission

Impact du débit sur le temps de transmission

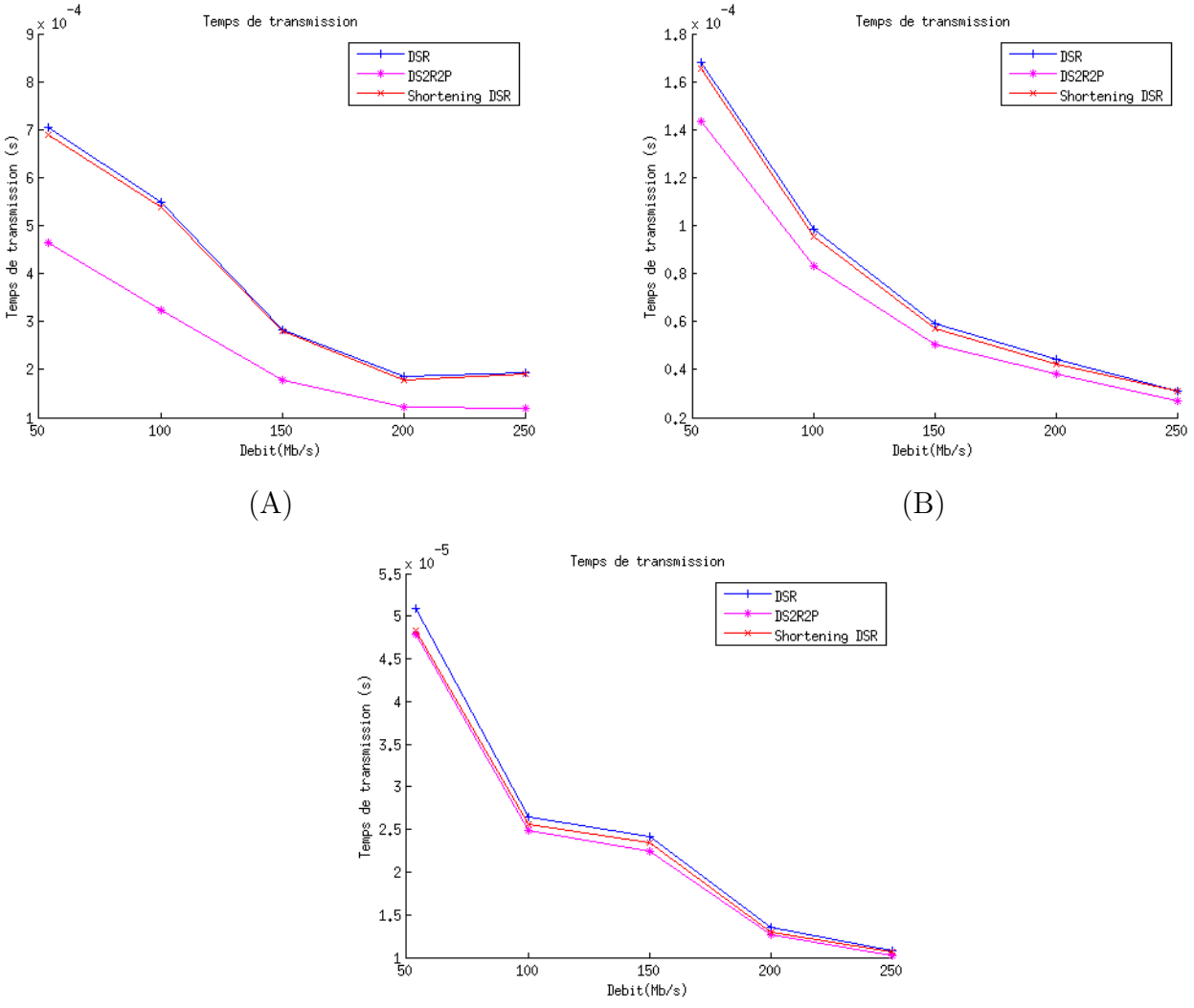


FIGURE 3.1 – Temps de transmission vs débit (Nombre de nœuds = 250, taille de haché =1 octet) (A) Cas d'une faible portée, (B) Cas d'une portée moyenne, (C) cas d'une haute portée

Sur la figure 3.1, on observe que dans le cas d'une faible portée, le temps de transmission de DSR est le plus élevé, suivi par l'extension DSR (Automatic route shortening). Cela est dû à la longueur des chemins du routage embarqué dans chaque

paquet de données. Plus le chemin de routage est long, plus la taille des paquets de

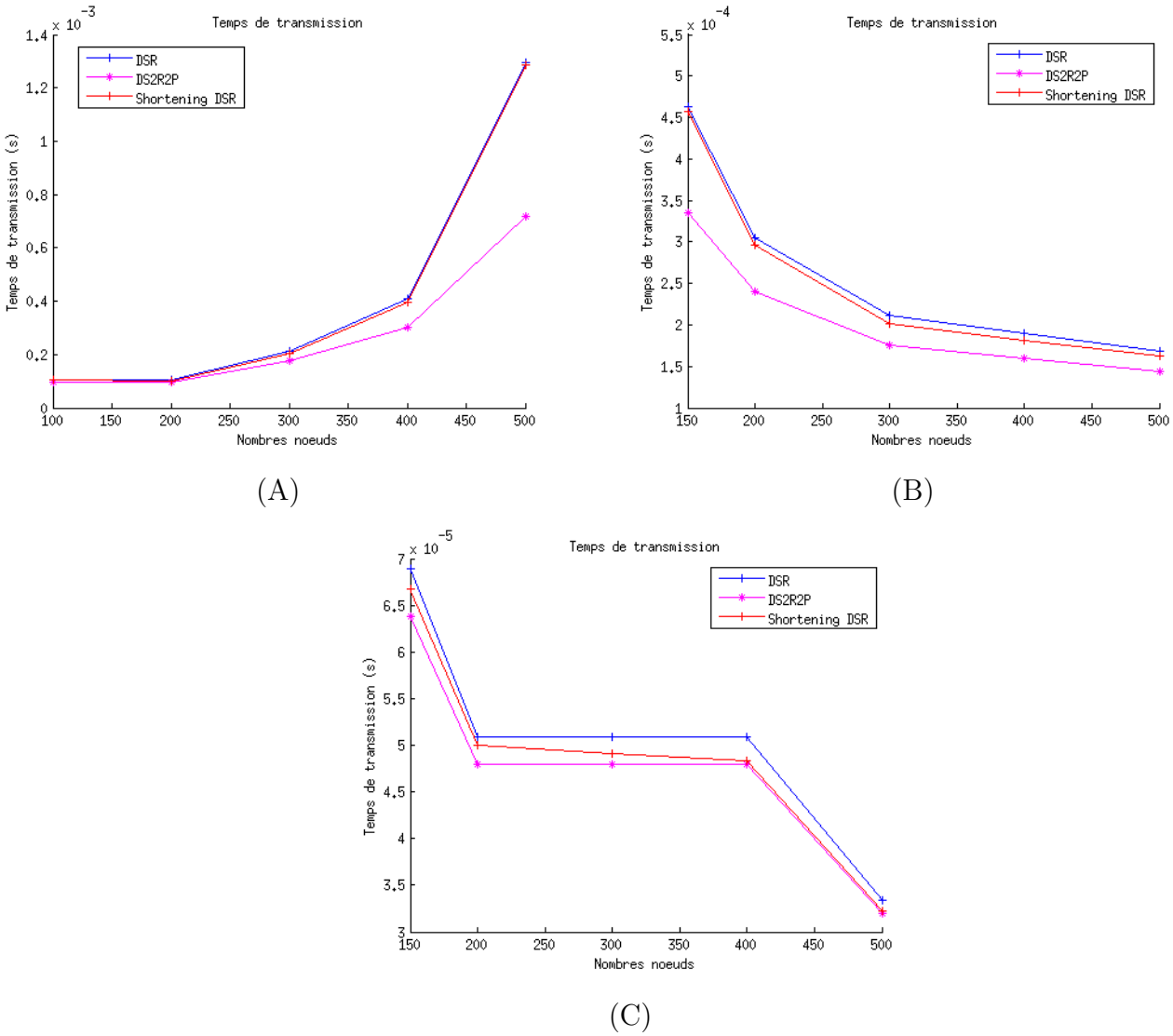


FIGURE 3.2 – Temps de transmission vs densité (Débit =54Mb/s, taille de haché =1 octet) (A) Cas d'une faible portée, (B) Cas d'une portée moyenne, (C) cas d'une haute portée

données est grande. Cependant, dans le protocole DS2R2P, le chemin de routage est résumé par une valeur de taille fixe (haché). Dans le cas d'une haute portée, le temps de transmission du protocole DSR et de l'extension DSR (Automatic route shortening) s'approche à celui de DS2R2P vu que la taille des chemins de routage devient beaucoup plus réduite.

Impact de la densité du réseau sur le temps de transmission

Sur la figure 3.2, on observe que le temps de transmission du protocole DS2R2P est inférieur à ceux de DSR et de l'extension DSR (Automatic route shortening). Dans le cas d'une faible portée, l'écart de performances entre le protocole DS2R2P

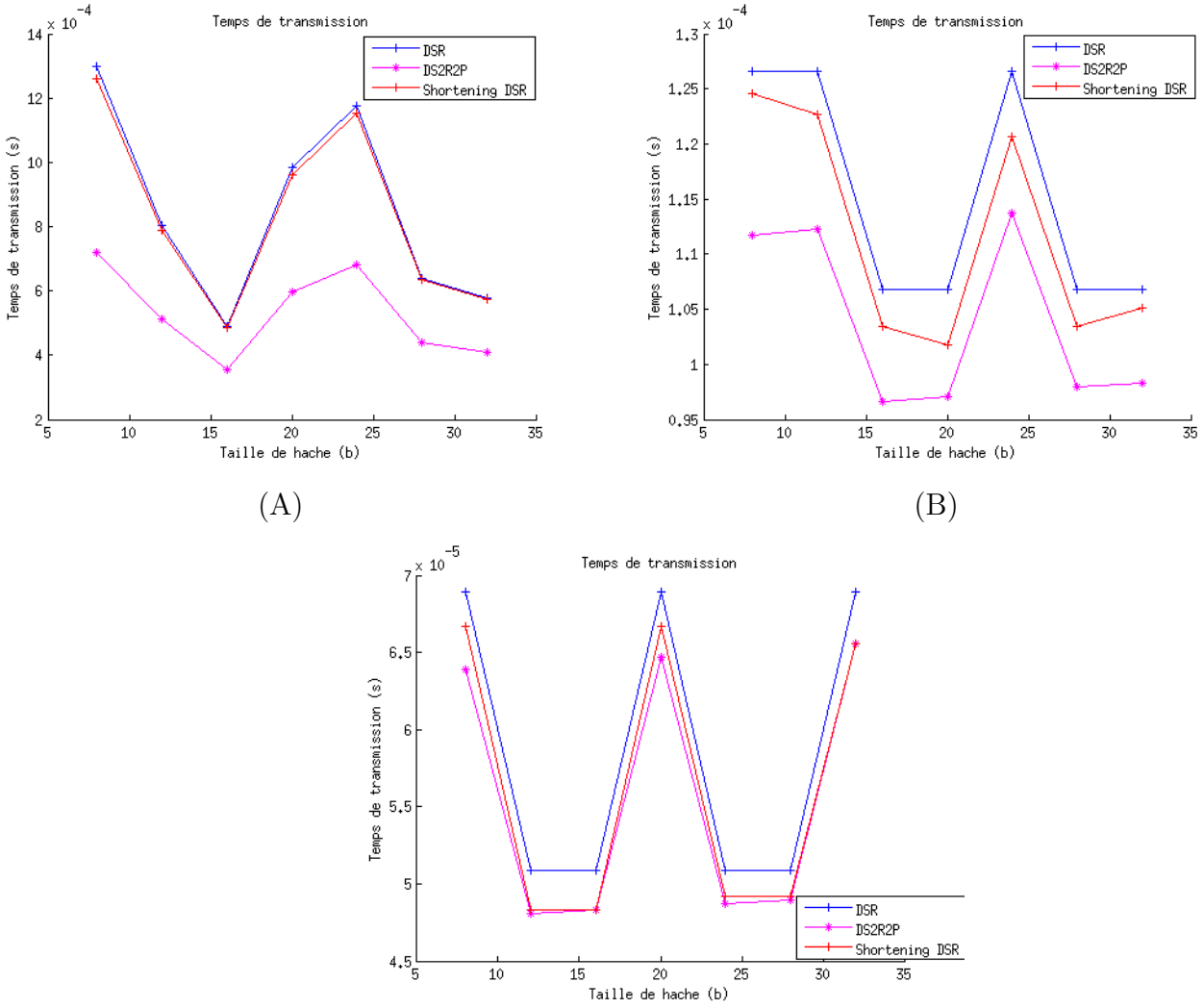


FIGURE 3.3 – Temps de transmission vs taille de haché (Débit =54Mb/s, Nombre de nœud=250) (A) Cas d'une faible portée, (B) Cas d'une portée moyenne, (C) cas d'une haute portée

et les deux autres protocoles augmente avec l'augmentation de la taille des chemins de routage (à l'exception du cas de densité de 100 jusqu'à 200 nœuds, où le temps de transmission des trois protocoles est presque égal, cela est dû au fait que le réseau

est partiellement déconnecté). Dans le cas d'une haute portée, on constate que le temps de transmission du protocole DSR et de l'extension DSR (Automatic route shortening) s'approche à celui de DS2R2P à cause de la taille réduite des chemins de routage.

Impact de la taille du haché sur le temps de transmission

Sur la figure 3.3, on constate que le temps de transmission de DS2R2P est inférieur à ceux de DSR et de l'extension DSR (Automatic route shortening) quelle que soit la taille de la valeur du haché. Cela est dû à la taille importante des chemins de routage embarqués dans les paquets de données, quoique le chemin de routage de l'extension DSR (Automatic route shortening) soit le plus court. Dans le cas d'une haute portée, le temps de transmission des deux protocoles DSR et l'extension DSR (Automatic route shortening) s'approche à celui de DS2R2P. On constate que dans le troisième cas (avec taille de haché = 28 bits), le temps de transmission des deux protocoles DS2R2P et l'extension DSR (Automatic route shortening) sont presque égaux, à cause de la taille réduite des chemins de routage.

3.3.2 Overhead

Impact de la densité du réseau sur la charge de contrôle

La figure 3.4 illustre les résultats de la charge de données en fonction de la densité du réseau. Nous constatons que dans tous les scénarios, la charge du contrôle de routage généré par DS2R2P est inférieure à ceux de DSR et de l'extension DSR (Automatic route shortening). Dans le cas d'une faible portée, les protocoles DSR et l'extension DSR (Automatic route shortening) génèrent plus de charge de contrôle que DS2R2P à cause de la taille importante des chemins de routage embarqués dans les paquets de données. L'écart de performances entre DS2R2P et les deux autres protocoles augmente avec l'augmentation de la taille des chemins de routage. Le gain en performances liées à DS2R2P est plus assistant. Dans le cas d'une haute portée,

l'overhead généré par les deux protocoles DSR et l'extension DSR (Automatic route shortening) s'approche de celui de DS2R2P.

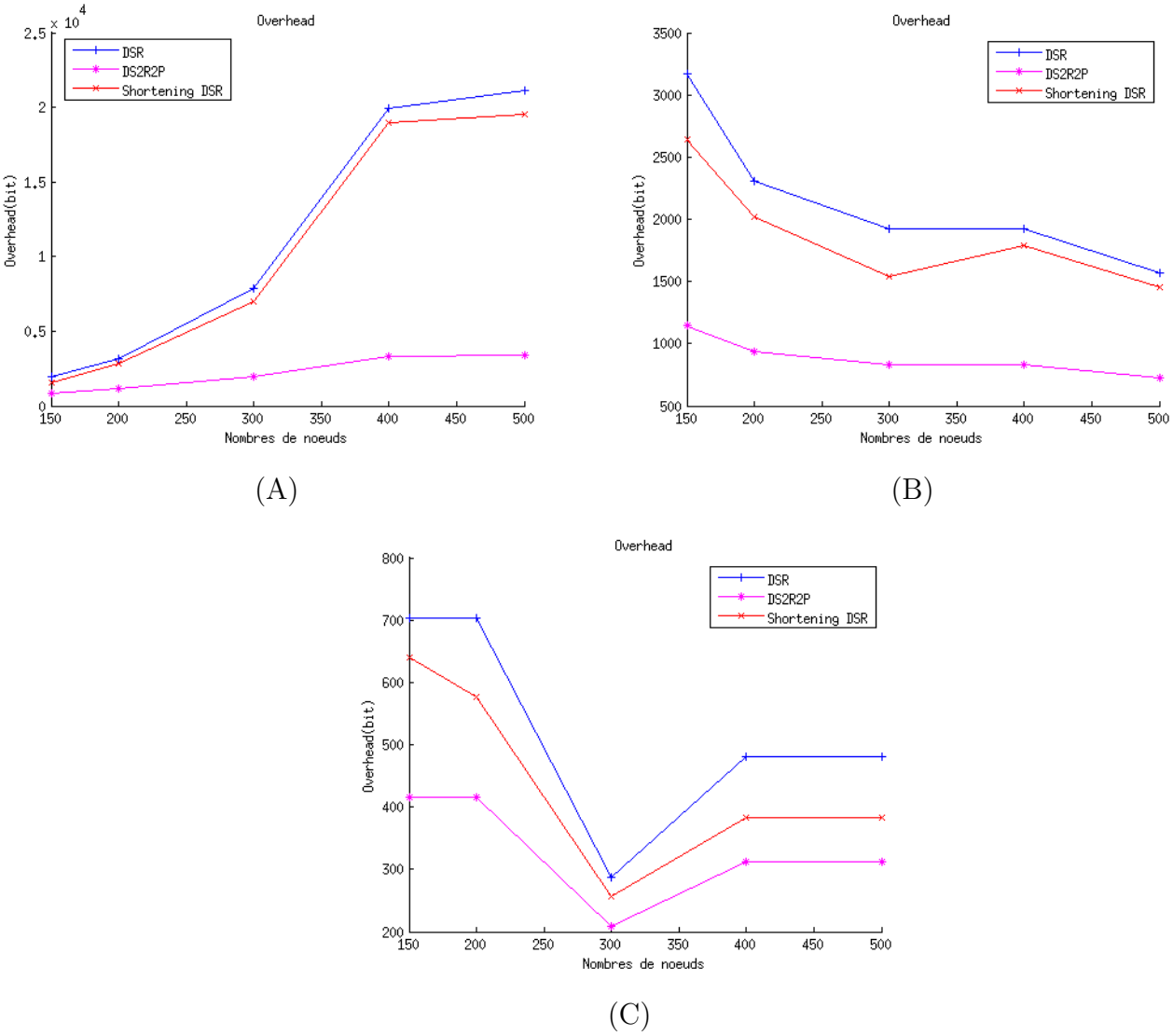


FIGURE 3.4 – Overhead vs densité (Débit =54Mb/s, Taille du haché = 1 octet) (A) Cas d'une faible portée, (B) Cas d'une portée moyenne, (C) cas d'une haute portée

Impact de la taille du haché sur la charge de contrôle

Le but de cette partie de simulation est d'étudier l'impact de la taille du haché sur la charge de routage générée en variant la taille de cette valeur d'un octet jusqu'à quatre octets (afin de minimiser le nombre des collisions dans la table des sous-chemins). Sur la figure 3.5, on observe que quelle que soit la taille de la valeur

du haché, le coût de DS2R2P est inférieure à ceux de DSR et de l'extension DSR (Automatic route shortening), notamment dans le cas d'une faible portée (les chemins de routage sont de tailles importantes). On constate, qu'à chaque fois la taille des chemins de routage augmente, la charge de contrôle générée par DS2R2P devient non considérable par rapport à ceux de DSR et l'extension DSR (Automatic route shortening). Cependant, dans le cas d'une haute portée, la différence entre DS2R2P et les deux autres protocoles est réduite.

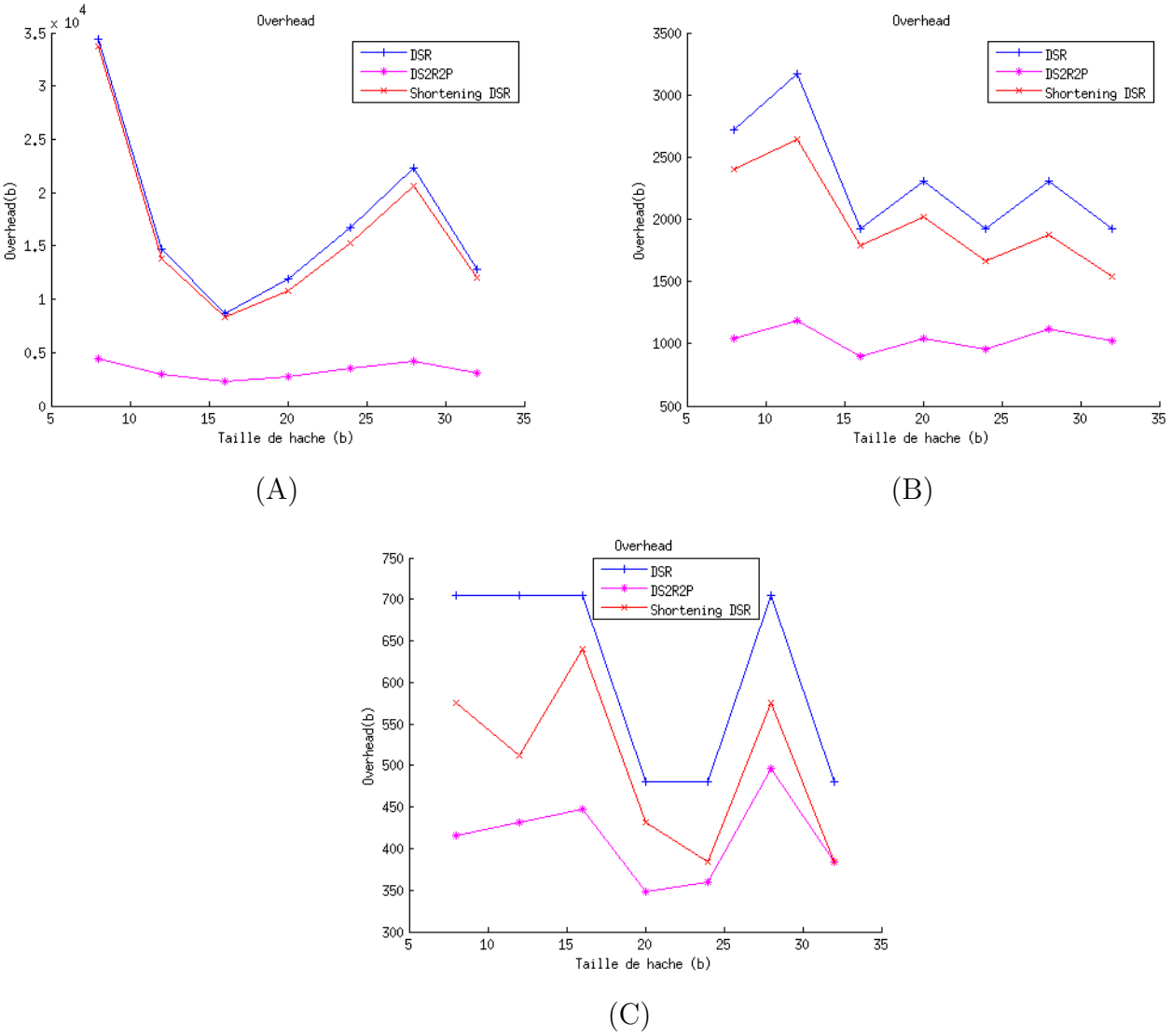


FIGURE 3.5 – Overhead vs taille du haché (Débit =54Mb/s, Nombre nœuds = 250) (A) Cas d'une faible portée, (B) Cas d'une portée moyenne, (C) cas d'une haute portée

3.3.3 Efficacité

Impact de la densité du réseau sur l'efficacité de routage

Sur la figure 3.6, on constate que le rapport d'efficacité de DS2R2P est le meilleur. Cela parce que dans les paquets de données DS2R2P, le chemin de routage est résumé par une valeur de taille fixe. Dans le premier cas, l'efficacité des trois protocoles a diminué avec l'augmentation de la densité du réseau. Cela est dû à l'augmentation de la taille des chemins de routage. Cependant, dans le cas d'une haute portée, on

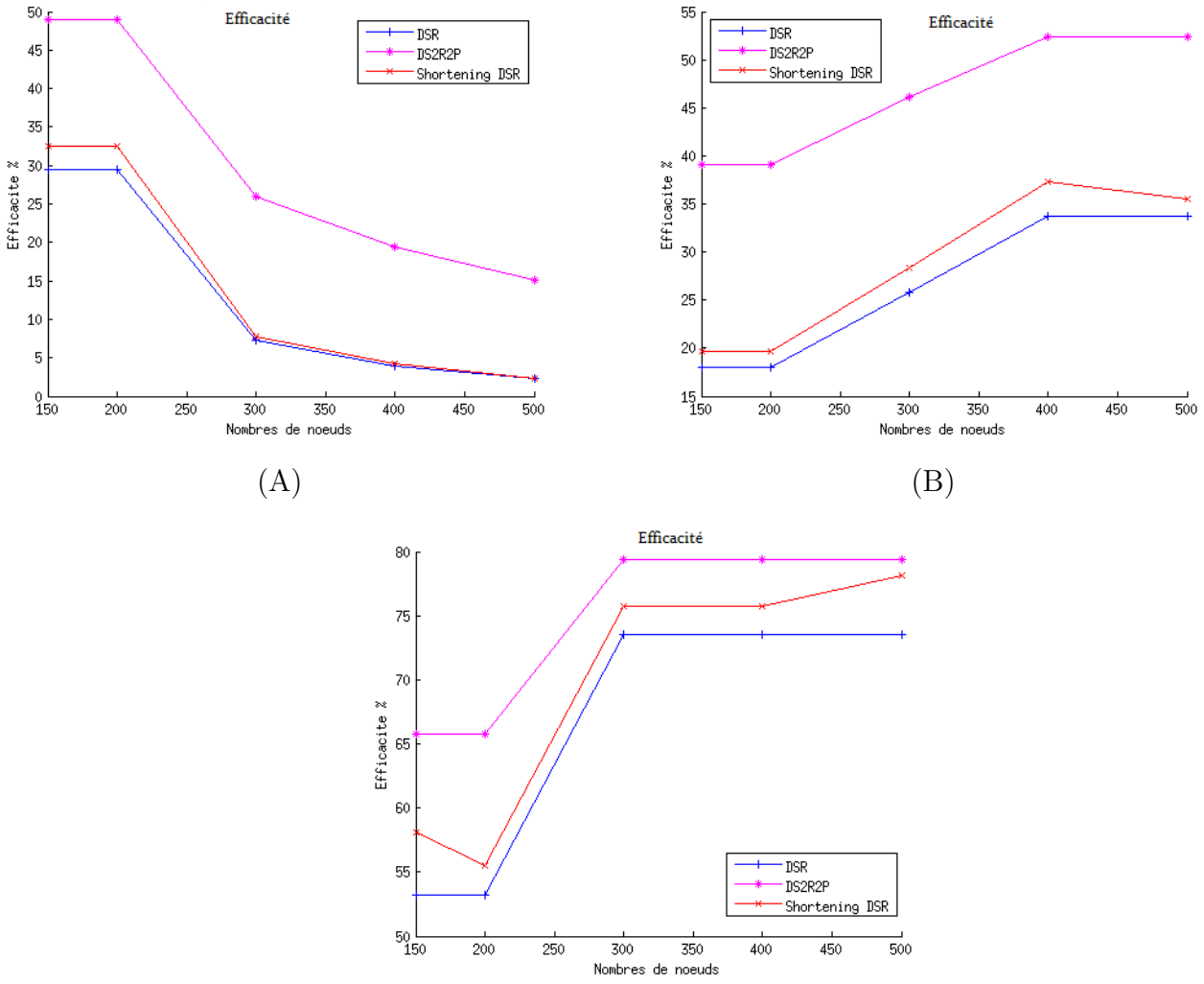


FIGURE 3.6 – Efficacité vs densité (Débit =54Mb/s, Taille du haché = 1 octets) (A) Cas d'une faible portée, (B) Cas d'une portée moyenne, (C) cas d'une haute portée

constate qu'à chaque fois la taille des chemins de routage se diminue, le rapport d'efficacité des trois protocoles augmente.

Impact de la taille du haché sur l'efficacité de routage

La figure 3.7 illustre l'impact de la taille du haché sur l'efficacité des trois protocoles. On remarque que l'efficacité de notre protocole est supérieure à celles de

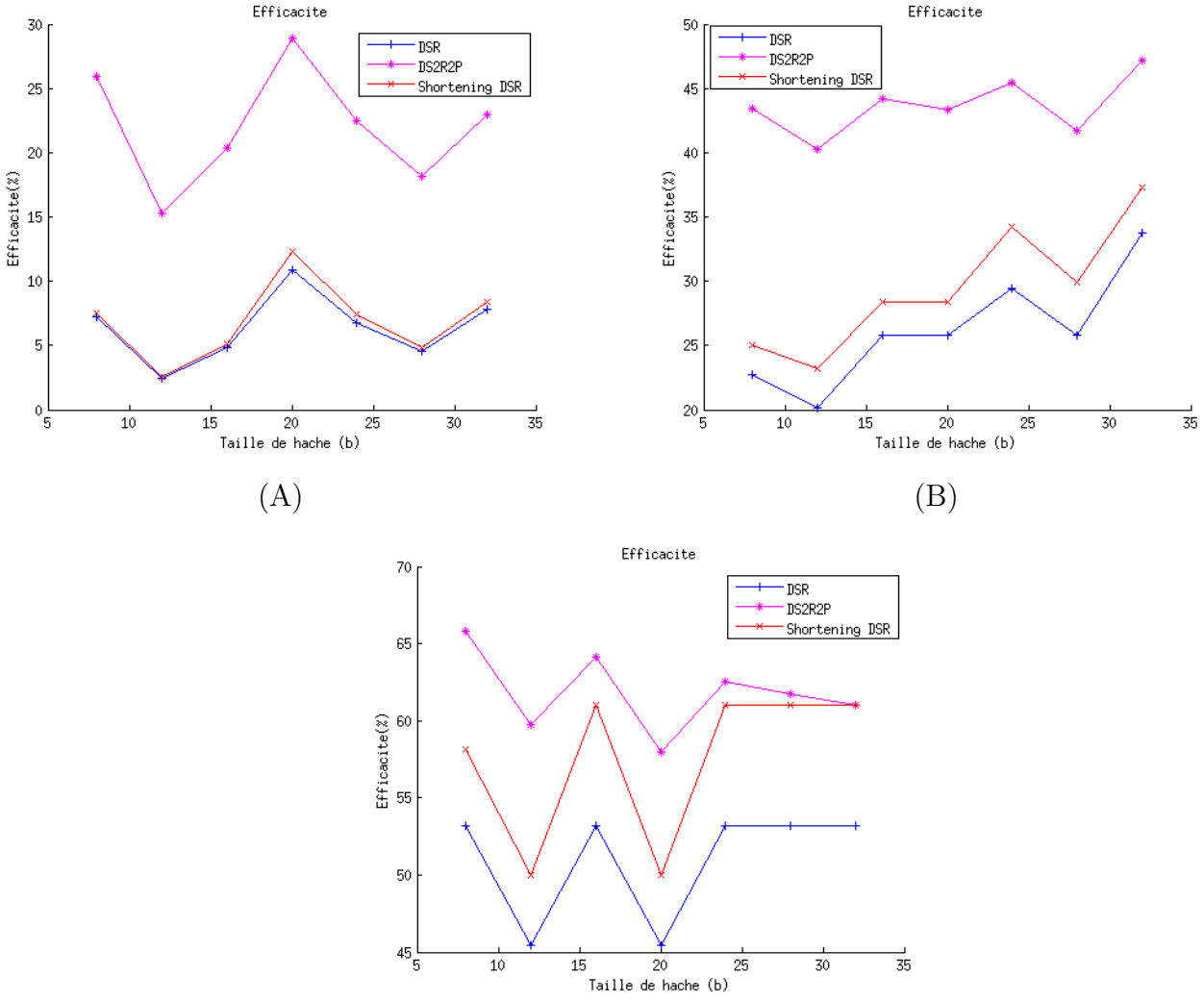


FIGURE 3.7 – Efficacité vs taille du haché (Débit =54Mb/s, Nombre de nœuds = 250) (A) Cas d'une faible portée, (B) Cas d'une portée moyenne, (C) cas d'une haute portée

DSR et de l'extension DSR (Automatic route shortening) quelle que soit la taille du

haché, notamment dans le premier et le deuxième cas, quoique l'accroissement de la taille du haché ait provoqué l'augmentation de la charge de routage de DS2R2P. Cette augmentation devient non considérable par rapport à la charge de contrôle de DSR et l'extension DSR (Automatic route shortening), cela est dû à la taille importante des chemins de routage. Cependant, dans le cas d'une haute portée, on constate que les performances de DSR et de l'extension DSR (Automatic route shortening) s'approchent à celui de DS2R2P. En effet, dans le troisième cas, l'efficacité des deux protocoles DS2R2P et l'extension DSR (Automatic route shortening) est presque égale, à cause de la taille réduite des chemins de routage.

3.4 Conclusion

Dans ce chapitre, nous avons présenté l'évaluation de performances de DSR, DS2R2P et l'extension DSR (Automatic route shortening) en termes de temps de transmission, charge de contrôle et d'efficacité. Les simulations ont été menées sous des conditions différentes de débit, densité et de taille de haché. Les résultats obtenus ont montré que DS2R2P minimise la charge de contrôle et le temps de délivrance des paquets lors de l'acheminement de données.

Conclusion générale

Dans les réseaux ad hoc, tout équipement peut être mis à contribution pour acheminer les données et chaque nœud participe au routage afin que tous les nœuds puissent créer un réseau autonome. Dans ce genre de réseau, la notion d'auto-organisation est très importante car chaque nœud participe à la survie du réseau. Dans ce mémoire, nous nous sommes intéressés aux travaux relatifs à la couche réseau, notamment aux protocoles de routage. L'étude et la mise en œuvre des protocoles de routage pour assurer la connexion des réseaux ad hoc sont des problèmes complexes.

Dans la conception des protocoles de routage pour les réseaux mobiles ad hoc, deux buts non compatibles doivent être atteints : d'une part il faut garder une vue consistante sur la topologie du réseau ce qui demande des fréquentes mises à jour à cause de la constante mobilité des nœuds et de l'instabilité du médium de communication sans fil (protocole proactif), et d'autre part il faut optimiser la consommation de la bande passante qui est une ressource limitée dans les réseaux mobiles ad hoc (protocole réactif). Il existe aussi les protocoles utilisant les deux procédés, ce sont les protocoles de routage hybrides. Plusieurs approches ont été proposées dans la littérature pour diminuer la consommation de la bande passante tout en cherchant

les chemins optimaux. Dans le chapitre 1, nous avons décrit plusieurs protocoles de routage réactif dont l'objectif commun est de minimiser la consommation de la bande passante en fournissant une classification des protocoles de routage réactifs selon l'axe de recherche suivi. Dans le chapitre 2, nous avons proposé le protocole DS2R2P (on Demand Source Routing with Reduced Packets Protocol). DS2R2P est un protocole de routage réactif, dont l'objectif de réduire la taille des paquets de routage source de DSR. DS2R2P réduits la taille des paquets en utilisant un mécanisme de hachage avant de transmettre le paquet de données, où les données sont transmises en utilisant des valeurs de tailles fixes et réduites. Il est à noter que le temps de calcul induit dans la fonction de hachage et le temps de recherche dans les tables des chemins hachés sont négligeables par rapport au développement de la technologie actuelle en termes de capacité de calcul et de stockage. Dans le dernier chapitre, nous avons réalisé une étude de performances de DS2R2P en le comparant aux protocoles DSR et l'extension DSR (Automatic Route Shortening). Les résultats obtenus ont montré l'efficacité de notre protocole en termes de temps de transmission lors de l'acheminement de données, ainsi que l'overhead généré.

Nos objectifs à court terme sont nombreux, un de ces objectifs est de réaliser d'autres tests sur d'autres environnements et sous d'autres conditions, en utilisant par exemple d'autres modèles de mobilités. Nous souhaitons aussi apporter dans nos futurs travaux quelques améliorations à cette première version en adaptant notre proposition pour d'autres extensions DSR, principalement : la réponse du cache (replayaing to route request using cached routes) et la récupération de route (salvaging).

Bibliographie

[1] C. Perkins, E. Royer, Ad hoc On-demand Distance Vector Routing, in : Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications (1999).

[2] D.B. Johnson, D.A. Maltz, J. Broch, DSR, the dynamic source routing protocol for multi-hop wireless ad hoc networks, in : C.E. Perkins Ad Hoc Networking, Addison-Wesley (2001).

[3] C. Perkins, P. Bhagwat, Highly dynamic destination-sequenced distance vector routing (DSDV) for mobile computers, in : ACM SIGCOMM (1994).

[4] K. Mahesh, Samir R.Das, On demand multipath distance vector routig in ad hoc networks (2001).

[5] J. Boice, J. Garcia-Luna-Aceves, K. Obraczka, Combining on-demand and opportunistic routing for intermittently connected networks, in : Ad Hoc Networks 7 (1) (2009).

[6] V. Ramesh, Dr. P. Subbaiah, Preemptive AOMDV Routing for mobile Ad hoc networks, in : Chennai and Dr. MGR University second International conference on sustainable energy and intelligent system (2011).

[7] Claude Castelluccia, Pars Mutaf, Hash-Based Dynamic Source Routing, in : International Federation for Information Processing (2004).

[8] Shobha et al, Efficient flooding using prefetching in on demand routing protocols for mobile ad hoc networks (2011).

[9] C.K. Toh, Associativity-based routing for ad-hoc mobile networks, in : Wireless Personal Communications Journal 4 (2) (1997).

[10] Z. Cheng, W. Heinzelman, Discovering long lifetime routes in mobile ad hoc networks, in : Ad Hoc Networks 6 (5) (2008).

[11] T. Goff, N.B. Abu-Ghazaleh, D.S. Phatak, R. Kahvecioglu, Preemptive routing in ad hoc networks, in : Journal of Parallel Distributed Computing 63 (2) (2003).

[12] N.-C. Wang, Y.-F. Huang, J.-C. Chen, A stable weight-based on demand routing protocol for mobile ad hoc networks, in : Information Sciences 177 (24) (2007).

[13] V.D. Park, M.S. Corson, A highly adaptive distributed routing algorithm for mobile wireless networks, in : Proceedings of INFOCOM (1997).

[14] C. Sengul, R. Kravets, Bypass routing, an on-demand local recovery protocol for ad hoc networks, in : Ad Hoc Networks 4 (3) (2006).

[15] Yu et al., a low overhead Dynamic route repairing mechanism for mobile ad hoc networks, in : computer communications 30 (2007).

[16] J. Mingliang et al. Internet Draft (CBRP), in : National univercity of singapore (1999).

[17] B. young, H. nitin, Location aided routing il mobile ad hoc networks, in : wireless networks 6 (2000).

[18] J. Eisbrener, G. Murphy, D. Eade, C. Pinnow, K. Begum, S. Park, S.M. Yoo, J.-H. Youn, Recycled path routing in mobile ad hoc networks, in : Computer Communications 29 (9) (2006)

[19] R. Beraldi, The polarized gossip protocol for path discovery in manets, in : Ad Hoc Networks 6 (1) (2008).

[20] J.-D. Abdulai, M. Ould-Khaoua, L. Mackenzie, Adjusted probabilistic route discovery in mobile ad hoc networks, in : Computers and Electrical Engineering 35 (1) (2009).

[21] Wenzheng et al., Overhead Analysis of query localization optimization and routing, in : copyright ACM (2011).

[22] R. Castaneda et al., Query localization techniques for on demand routing protocols in ad hoc networks, in : wireless networks 8, (2002).

[23] M. Gunes, U. Sorges, I. Bouazizi, Ara-the ant-colony based routing algorithms for manets, in : Proceedings of IEEE ICPP Workshop on Ad Hoc Networks (IWAHN) (2002).

[24] L. Rosati, M. Berioli, G. Reali, On ant routing algorithms in ad hoc networks with critical connectivity, in : Ad Hoc Networks 6 (6) (2008).

[25] M. Naserian, K. Tepe, Game theoretic approach in routing protocol for wireless ad hoc networks, in : Ad Hoc Networks (2009).

[26] Q. Xue, A. Ganz, Ad hoc QoS on-demand routing (AQOR) in mobile ad hoc networks, in : Journal of Parallel Distributed Computing 63 (2) (2003).

[27] Gabriel et al, Qos routing with traffic distribution in mobile ad hoc networks, in : computer communication 32 (2009).

[28] D.Jhonson, Y.Hu, D. Maltz, The dynamic source routing protocol (DSR) for mobile ad hoc network fot IPV4, in : working group (2007).

[29] Azzedine boukerche et al., Routing protocols in ad hoc networks : A survey, in : Computer networks 55 (2011).

[30] J. Rachit et S.Laxmi, Study and performance comparaison of AODV and DSR on the basis of path loss propagation models, in : International of advanced science and technology vol. 32. (2011).

[31] Johnson (D.) et Maltz (D.), Dynamic source routing in ad hoc wireless networks, in : Mobile Computing, éd. par Imielinski et Korth. Kluwer Academic Publishers (1996).

[32] Cancheng Huang et al., Analysis the anhanced dynamic source routing Algorithme for route link quality, in : ah doc network, Proceeding of the 9th international symposium on linear drives for industry applications, volume4, Notes in electrical ingineering 273, copyright springer Verlag Berlin Heidelberg (2014).

[33] Manish Bhardwaj et al. Performance Analysis of Dynamic source routing for Ad hoc Networks using active Packet, N. Chaki et al. (eds.),in : Computer Networks et Communications (NetCom), 221 ; Lecture Notes in Electrical Engineering copyrigt Springer Science+Business Media New York (2013).

[34] Dr.P.Nirmal Kumar et Deepadasarathan, Novel Method to avoid stale route cache problem of dynamic source routing protocol for mobile Ad hoc Network, in : International conference on current Trends in Engineering and technologie, ICCTET 13 (2013).

[35] Istikmal, Analysis And Evaluation Optimization Dynamic Source Routing (DSR) Protocol in Mobile Adhoc Network Based on Ant Algorithm, in : International Conference of information and communication technology (ICoICT) (2013).

[36] Song Guo, Oliver W.yang, Performance of backaup source routing in mobile ad hoc networks (2011).