Titre: Title:	Routage et réparation de routes dans les réseaux mobiles ad-hoc
Auteur: Author:	Benjamin Macabéo
Date:	2003
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Macabéo, B. (2003). Routage et réparation de routes dans les réseaux mobiles ad-hoc [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/6984/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/6984/
Directeurs de recherche: Advisors:	Samuel Pierre
Programme: Program:	Génie électrique

UNIVERSITÉ DE MONTRÉAL

ROUTAGE ET RÉPARATION DE ROUTES DANS LES RÉSEAUX MOBILES AD-HOC

BENJAMIN MACABÉO
DÉPARTEMENT DE GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE ÉLECTRIQUE)
AVRIL 2003

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

ROUTAGE ET RÉPARATION DE ROUTES
DANS LES RÉSEAUX MOBILES AD-HOC

présenté par : MACABÉO Benjamin
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury d'examen composé de :

M. GALINIER Philippe, Ph.D., président
M. PIERRE Samuel, Ph.D., membre et directeur de recherche
M. QUINTERO Alejandro, Ph.D., membre

REMERCIEMENTS

À M. Samuel Pierre, mon directeur de recherche, dont le soutien et les précieux conseils m'ont guidé tout au long de ma recherche.

À ma famille et mes amis pour leurs encouragements et leur soutien.

À tous les membres du LARIM (Laboratoire de recherche en Réseautique et Informatique Mobile) pour l'ambiance d'entraide qui y règne.

RÉSUMÉ

Contrairement aux réseaux cellulaires, les réseaux ad-hoc sont une forme de réseaux mobiles complètement dépourvus d'infrastructure fixe. Dans ces réseaux, les unités mobiles sont capables de s'auto-organiser pour acheminer les données d'une unité mobile à une autre. L'une des faiblesses des réseaux ad-hoc réside dans le fait que les routes utilisées entre une source et une destination sont susceptibles de se rompre subrepticement en cours de communication. Cette rupture est liée au fait que les nœuds constituant la route sont mobiles et risquent donc de se déplacer hors de portée des autres nœuds de la route.

Pour remédier à ces problèmes, une approche consiste à sélectionner des routes mettant en jeu des nœuds qui ont un comportement le plus stable possible (en particulier, une faible mobilité). Cependant, une autre approche consiste à améliorer la procédure de réparation des routes. Des algorithmes ont ainsi été développés pour réparer les routes de manière locale sans faire intervenir de re-routage initié par la source.

Dans ce mémoire, nous proposons une méthode qui vise à améliorer la probabilité de succès d'une réparation locale de route dans les réseaux mobiles ad-hoc. Cette méthode se base sur la densité des nœuds au voisinage d'une route ainsi que sur la disponibilité de ce voisinage. La prise en compte de ces paramètres lors de la phase de sélection de route (en fin de processus de routage) permet de sélectionner, parmi plusieurs routes, celle qui est potentiellement la plus facilement réparable. Par ailleurs, nous proposons une méthode de détection anticipée de l'échec d'une réparation locale de route. Cette méthode permet de déclencher directement une procédure de re-routage globale plus apte à rétablir la communication entre la source et la destination.

Lors de la mise en œuvre des méthodes, comme contribution de ce mémoire, nous avons implémenté nos améliorations dans le protocole AODV puis comparé les résultats obtenus lors de simulations avec ceux obtenus pour des versions n'incorporant pas nos améliorations.

Nous obtenons des résultats intéressants en particulier en ce qui concerne le nombre de paquets perdus et le temps moyen de réparation de route après la défection d'un nœud. Ainsi même si la durée de la phase de routage initiale est systématiquement rallongée par rapport à la version du protocole AODV n'incorporant pas notre amélioration, la sélection de la route la plus facilement réparable s'avère bénéfique en terme de gain de qualité de service.

ABSTRACT

Contrary to cellular networks, ad-hoc networks are a form of mobile networks without any fixed infrastructure. In these networks, the mobile units are able to self-organize in order to transmit the data between mobile units. One of the weaknesses of the ad-hoc networks is that a road used between a source and a destination is likely to break surreptitiously during the communication. This rupture is related to the mobility of each node constituting the road and thus any node is likely to move out of range of its neighbors.

To solve these problems, an approach consists of selecting roads in which nodes have the most stable possible behavior (in particular, a low mobility). However, another approach consists of improving the procedure of repairing the roads. Algorithms were thus developed to repair the roads in a local way without using route request initiated by the source.

In this thesis, we propose a method which improve the chances of success of a local route repair in a mobile ad-hoc networks. This method is based on the density of the nodes in the neighborhood of a route and on the availability of this neighborhood. Taking into account these parameters during the phase of selecting the best path (at the end of the process of path search) makes it possible to select, among several paths, that which is potentially most easily reparable.

For the implementation of our method which is the main contribution of this thesis, we implemented our improvement on AODV protocol. Then we compared the AODV protocol with the same version including our improvements. We have obtained interesting results, in particular for the number of packets lost and for the mean repair time of road after the defection of a node.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES SIGLES ET ABRÉVIATIONS

Sigle ou abréviation	Signification
IP	Internet Protocol
UDP	User Datagramm Protocol
TCP	Transmission Control Protocol
BTS	Base Transceiver Station
MSC	Mobile Switching Centre
IEEE	Institute of Electrical and Electronics Engineers
NIST	National Institute of Standards and Technology
IETF	Internet Engineering Task Force
QdS	Qualité de Service
MAC	Medium Access Control
ADSL	Asymmetric Digital Subscriber Line
MANET	Mobile Ad-hoc NETwork
OSI	Open System Interconnexion
RSVP	Resource Reservation Setup Protocol
CSMA	Carrier Sense Multiple Access
DSDV	Destination-Sequenced Distance-Vector
DSR	Dynamic Source Routing
AODV	Ad-Hoc On-demand Distance Vector.
RREQ	Route REQuest
RREP	Route REPly
RERR	Routing error
TDMA	Time Division Multiple Access
TTL	Time To Live
CPU	Central Processing Unit
ETE	End To End
ms	milliseconde

CHAPITRE I
INTRODUCTION

Depuis ses origines, l'Internet est basé sur le principe du *Best Effort* qui consiste à utiliser les liens de manière optimale, quitte à introduire des délais supplémentaires dans la transmission des données. Ce principe est aujourd'hui remis en cause par l'émergence d'applications qui nécessitent une prise en compte de la qualité de service (téléphonie IP, jeux en réseaux ou encore applications multimédia). Ainsi, l'Internet originel qui permettait d'envoyer des données sur le principe du *best-effort* ne suffit plus : il doit évoluer vers un Internet capable de répondre aux nouvelles exigences en matière de qualité de service (délai, bande-passante, etc.). L'usager ne se contente en effet plus d'envoyer des courriers électroniques, désormais il souhaite également pouvoir communiquer en direct avec l'autre bout de la planète, en visioconférence par exemple. Pour pouvoir être pris en charge, ces nouveaux besoins nécessitent une adaptation du réseau. De ce fait, la prise en compte de la qualité de service par Internet est devenue un champ d'investigation auquel se consacrent nombre de chercheurs depuis plusieurs années. Si des solutions sont proposées pour les réseaux filaires, il reste encore beaucoup à faire, en particulier dans les réseaux mobiles ad-hoc, qui fait l'objet de ce mémoire. Dans ce chapitre d'introduction, nous présenterons les définitions et concepts de base nécessaires à la bonne compréhension du mémoire, nous préciserons les éléments de la problématique et les objectifs de la recherche, pour terminer avec une esquisse du plan du mémoire.

1.1 Définitions et concepts de base

Dans les réseaux mobiles cellulaires, la zone de couverture est découpée en cellules contiguës, tel qu'illustré à la figure 1.1. Chaque cellule est desservie par une station de base (BTS) fixe servant d'intermédiaire entre les différentes unités mobiles du réseau. Les communications sont établies à l'aide de *commutateurs* de services mobiles

(MSC). Ce type d'architecture est aussi constitué par un réseau dorsal (backbone) utilisé pour acheminer la communication de la cellule où se situe l'émetteur vers celle où se situe l'unité mobile. Les communications entre unités mobiles mettent donc en jeu une infrastructure fixe complexe et coûteuse.

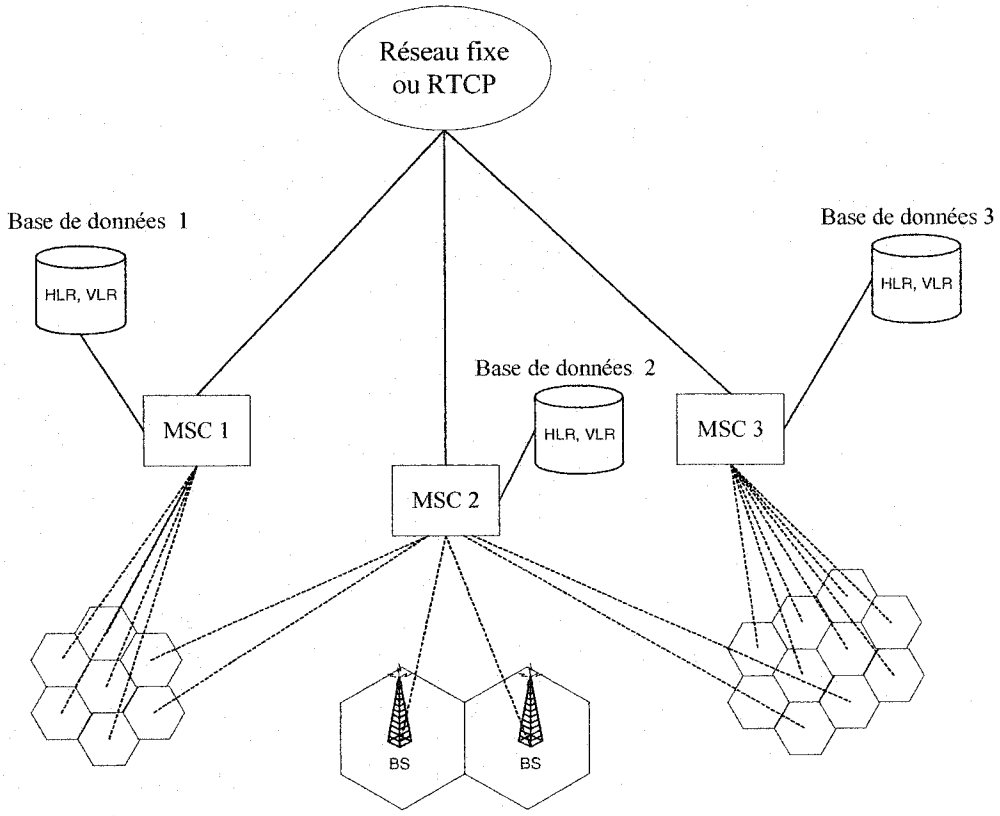


Figure 1.1 Architecture d'un réseau mobile cellulaire

Un *réseau ad-hoc*, pour sa part, est un réseau mobile sans fil constitué d'un ensemble de nœuds mobiles susceptibles d'établir des communications les uns avec les autres, mais dépourvu de toute infrastructure de commutation (BTS, MSC, etc.). Les nœuds de ces réseaux doivent donc être en mesure de s'auto-organiser, de coopérer pour permettre la bonne circulation de l'information entre les différents éléments qui le constituent. Si ces réseaux étaient initialement principalement conçus pour des

opérations militaires, on constate aujourd'hui leur émergence chez les particuliers sous la forme de réseaux domestiques, en particulier grâce à la norme IEEE 802.11. Cette forme de réseau connaît donc un succès croissant : il est même envisagé de déployer des réseaux ad-hoc à grande échelle. Leur principal atout réside dans le fait qu'ils ne sont pas tributaires d'installations fixes. Leur déploiement en est donc grandement simplifié par rapport aux autres formes de réseaux mobiles.

Dans un réseau ad-hoc, une *route* est définie par l'ensemble des unités mobiles qui contribuent à la bonne transmission des données de la source jusqu'à la destination. Le *routage* est l'opération de découverte et de sélection des routes entre la source et la destination.

La *qualité de service* ou QdS consiste en un panel de caractéristiques ou contraintes (bande passante, délai, gigue, garantie sur les taux de perte de paquets...) qu'une connexion entre une source et une destination se doit de vérifier au cours d'une communication pour répondre aux besoins d'une application donnée.

Le processus de routage tenant compte de la qualité de service consiste donc à déterminer les routes entre une source et sa destination qui vérifient les contraintes exigées par l'application source et à choisir parmi ces routes la route convenable qui optimise l'utilisation des ressources du réseau. Un algorithme de routage qui tient compte de la QdS doit aussi prendre en charge la maintenance des routes. En effet, la probabilité que la route utilisée se rompe au cours de la connexion est importante dans les réseaux ad-hoc du fait de la mobilité de tous les nœuds du réseau.

1.2 Éléments de la problématique

Face à la multiplication du nombre d'applications qui requièrent des exigences en terme de QdS, la prise en charge de la QdS dans les réseaux ad-hoc devient un enjeu incontournable pour assurer leur succès. Cependant, cette prise en charge n'est pas aisée. En effet, les contraintes imposées par les réseaux ad-hoc sont multiples : le médium radio utilisé est peu fiable, la topologie entièrement dynamique et la puissance de calcul des unités mobiles très limitée. De plus, la bande passante disponible est trop faible pour

autoriser de lourds messages de contrôle. Enfin, la prise en charge de la qualité de service est un problème réparti sur l'ensemble du chemin qui sépare la source de la destination; l'algorithme qui prend en charge la qualité de service est donc réparti. L'idée est d'assurer la qualité de service grâce à un processus de routage capable de détecter les chemins répondant aux exigences de QdS, couplé à un processus de réservation des routes détectées. Cependant, il faut intégrer à ce schéma la maintenance des routes rompues en cours de communication.

Une autre exigence dans la prise en charge de la QdS au niveau des réseaux ad-hoc est d'assurer la compatibilité de cette prise en charge avec les autres types de réseaux avec lesquels une cohabitation sera nécessaire. En effet, les réseaux de la future génération seront des réseaux hétérogènes intégrés et, comme le concept de QdS fait intervenir l'idée de connexion, les solutions envisagées doivent être supportées par l'ensemble des nœuds et donc des réseaux mis en jeu par la connexion (solutions de bout-en-bout). Au passage, il est important de noter que cette solution devra être basée sur IP, protocole qui semble incontournable dans le contexte actuel.

Jusqu'à présent, plusieurs pistes ont déjà été explorées, on peut les classer en quatre grandes catégories :

- *Modèles de QdS* : définissent les services fournis et les mécanismes à employer pour les fournir. Comme exemple de modèle de QdS, on peut citer FQMM (*a Flexible QdS Model for Mobile Ad-hoc Networks*) qui constitue une adaptation intéressante de *IntServ* et *DiffServ* aux réseaux ad-hoc. Cependant, cette solution relègue la plupart des problèmes à prendre en compte au protocole de routage sous-jacent ;
- *Couche MAC* : offre aux couches supérieures les mécanismes pour assurer la Qualité de Service. La faiblesse de ces mécanismes réside en particulier dans le fait que la modélisation du fonctionnement du médium radio est trop simpliste (les liens sont toujours considérés comme symétriques, alors que ce n'est pas rigoureusement le cas) ;

- *Routage QdS* : recherche et maintenance de routes selon un critère donné. Quelques solutions ont été proposées mais toutes engendrent un surcoût important en terme de trafic par rapport aux protocoles de routage de type *best-effort* ;
- *Protocoles de signalisation* : définissent les messages de contrôle. Leur objectif est de fournir un moyen de propager des informations de contrôle à travers un réseau.

L'idée qui ressort de ces différentes investigations est de récupérer les technologies conçues pour les réseaux classiques afin de conserver une certaine compatibilité avec les autres réseaux, mais en les simplifiant au maximum pour les adapter aux exigences des réseaux ad-hoc. Cependant, chacune de ces pistes présente des inconvénients notoires. De plus, toutes ces solutions ne gèrent que des problèmes particuliers. Aucune solution globale n'est apportée au problème de la qualité de service dans les réseaux ad-hoc.

1.3 Objectifs de recherche

L'objectif principal de ce mémoire est de proposer des mécanismes efficaces de gestion de la Qualité de Service dans les réseaux ad-hoc en nous focalisant sur l'aspect routage, tout en prenant soin de fournir des solutions adaptées à des systèmes évolutifs à grande échelle. De manière plus spécifique, nous visons les objectifs suivants :

- analyser les solutions déjà apportées en matière de qualité de service, en particulier au niveau du routage en vue d'en souligner leurs faiblesses ;
- concevoir et implémenter des algorithmes et protocoles qui permettent une meilleure gestion du problème de la qualité de service dans les réseaux ad-hoc ;
- évaluer la performance de ces algorithmes et protocoles en les comparant aux meilleurs modèles répertoriés dans la littérature.

1.4 Plan du mémoire

À la suite de ce chapitre d'introduction, ce mémoire s'articule autour de quatre chapitres.

Le deuxième chapitre de ce mémoire fait état des travaux relatifs à la prise en compte de la qualité de service dans les réseaux ad-hoc. Préalablement à tout travail de recherche, il apparaît en effet indispensable de faire un état de l'art et d'analyser en profondeur les mécanismes qui ont prévalu dans les recherches antérieures.

Dans le chapitre trois, nous introduisons nos solutions pour optimiser la prise en compte de la QdS lors du routage dans les réseaux ad-hoc dans la perspective de réseaux intégrés, ainsi que les algorithmes et protocoles qui en découlent.

Dans le chapitre quatre, nous présentons les détails d'implémentation de ces algorithmes et protocoles et utilisons des simulations pour analyser leur performance. Ces simulations nous aiderons donc à optimiser les protocoles proposés mais également de les confronter aux algorithmes disponibles dans la littérature.

Enfin, la conclusion permettra de faire une synthèse du travail réalisé dans le cadre de la maîtrise en mettant en exergue les principales contributions apportées par notre travail. Dans cette partie, nous suggérons également des directions de recherche pour des travaux futurs.

CHAPITRE II
ROUTAGE DANS LES RÉSEAUX MOBILES AD-HOC

Les environnements mobiles offrent aujourd'hui une grande flexibilité d'emploi. En particulier, ils permettent la mise en réseau de sites dont les câblages seraient trop onéreux à réaliser dans leur totalité, voire impossible. Il existe deux types de réseaux mobiles sans fil : les réseaux avec infrastructures et les réseaux sans aucune infrastructure.

Les réseaux sans fil comportant des infrastructures sont les plus répandus aujourd'hui. Dans ces réseaux, des stations de base, fixes, jouent le rôle d'intermédiaire et de routeur pour les unités mobiles. Une unité mobile ne peut être, à un instant donné, directement connectée qu'à une seule station de base. Elle communique avec les autres sites grâce à la station à laquelle elle est directement rattachée. L'acheminement de l'information entre différentes stations de bases est assuré par un réseau statique filaire.

Dans ce chapitre, nous commençons par définir les concepts propres aux réseaux ad-hoc puis nous exposons les différentes manières de prendre en compte la qualité de service dans de tels réseaux. Enfin, dans une dernière partie, nous nous focalisons sur l'une de ces manières : le routage avec qualité de service.

2.1 Définition et concepts de base

Les réseaux de télécommunication 1G, 2G et 3G, dont une représentation schématique est donnée à la Figure 2.1, mettent en jeu un réseau dorsal statique ainsi que des stations de base fixes. Ces infrastructures fixes sont coûteuses et complexes à déployer.

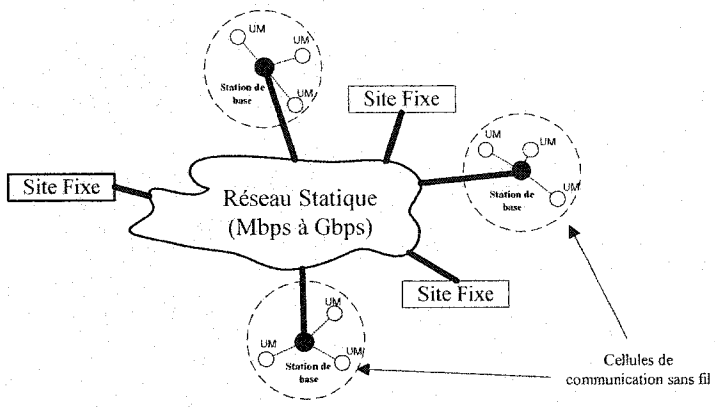


Figure 2.1 Modèle des réseaux mobiles avec infrastructure

Un nouveau type de réseaux sans fil, communément appelé réseau *ad-hoc,* est en plein essor aujourd'hui. Ces réseaux, qui constituent l'évolution naturelle des réseaux évoqués plus haut, sont dépourvus de toute infrastructure fixe. Les terminaux qui composent le réseau doivent donc être capables eux-même de s'interconnecter, de s'organiser et de communiquer entre eux. Ces réseaux présentent l'intérêt de se déployer très rapidement à n'importe quel endroit et à n'importe quel moment. Des exemples d'application des réseaux ad-hoc sont les opérations de sauvetage après une catastrophe naturelle, les campagnes militaires (déploiement de troupes en milieu hostile), etc. Plus communément, ils permettent d'établir spontanément un réseau à l'occasion d'un meeting ou d'un cours où ils facilitent l'échange d'information entre les participants. Par ailleurs, avec l'arrivée de la norme IEEE 802.11, il est important de signaler que les réseaux ad-hoc commencent à apparaître chez les particuliers sous la forme de réseaux domestiques de petite taille qui permettent d'interconnecter les différentes ressources informatiques du foyer et de les raccorder à l'Internet par l'intermédiaire d'un modem, du câble ou d'une ligne ADSL.

Le développement des réseaux *ad-hoc* fait face à de nombreux défis. Par exemple, les entités doivent pouvoir s'adapter rapidement à tout changement de topologie du réseau comme la rupture d'un lien lié au départ d'un terminal. Elles doivent également assumer le routage des paquets ainsi que leur transmission ou retransmission. Depuis quelques années, beaucoup de protocoles de routages ont été proposés dans la

littérature pour répondre aux exigences des réseaux *ad-hoc*. Cependant, les solutions proposées prennent rarement en compte la qualité de service (QdS). Or de plus en plus d'utilisations des réseaux mettent en jeu des applications en temps réel qui requièrent une certaine qualité de service comme le multimédia. Dès lors, il devient très important de concevoir des protocoles de routage efficaces prenant en compte la qualité de service.

Un réseau mobile ad-hoc, appelé aussi MANET (Mobile Ad-hoc NETwork), consiste en une grande population, relativement dense, d'unités mobiles qui se déplacent dans un territoire donné et dont le seul moyen de communication est l'utilisation d'interfaces sans fil, sans l'aide d'aucune infrastructure préexistante ou administration centralisée. La communication entre deux unités mobiles hors de portée radio l'une de l'autre s'effectue grâce à un processus *multihop*, les nœuds intermédiaires jouant le rôle de routeur.

Un réseau ad-hoc peut être modélisé par un graphe $G_t = (V_t,E_t)$ où V_t représente l'ensemble des nœuds (i.e. les unités ou les hôtes mobiles) du réseau et E_t modélise l'ensemble les connections qui existent entre ces nœuds. Si e = (u,v) appartient à E_t, cela signifie que les nœuds u et v sont en mesure de communiquer directement à l'instant t.

La Figure 2.2 représente un réseau *ad-hoc* de 12 unités mobiles sous forme d'un graphe. On remarque aisément que ce graphe est connexe ainsi, toutes les unités mobiles du réseau sont en mesure de communiquer entre elles soit directement comme les unités 2 et 4, soit par l'intermédiaire d'autres unités comme pour les unités 9 et 2. Dans ce cas, les unités 4, 6 et 7 jouent le rôle d'intermédiaires de rediffusion.

Les réseaux mobiles ad-hoc sont caractérisés par :

- *Une topologie dynamique* : Les unités mobiles du réseau, se déplacent d'une façon libre et arbitraire. Par conséquent la topologie du réseau peut changer, à des instants imprévisibles, d'une manière rapide et aléatoire. Les liens de la topologie peuvent être unidirectionnels ou bidirectionnels ;
- *Une bande passante limitée* : Une des caractéristiques primordiales des réseaux basés sur la communication sans fil est l'utilisation d'un médium de

communication partagé. Ce partage entraîne que la bande passante réservée à un hôte est modeste ;

- *Des contraintes d'énergie* : Les hôtes mobiles sont alimentés par des sources d'énergie autonomes comme des batteries ou d'autres sources consommables. La contrainte énergétique doit être prise en considération dans tout contrôle fait par le système ;
- *Une sécurité physique limitée* : Les réseaux mobiles ad-hoc sont plus vulnérables aux attaques que les réseaux filaires classiques. Cela se justifie par le fait que le contrôle des données transférées doit être minimisé pour limiter la surcharge du réseau. De plus, comme les messages envoyés transitent par plusieurs nœuds, il est difficile d'assurer qu'aucun nœud qui participe au routage n'accède à l'information véhiculée ou pire la modifie.
- *L'absence d'infrastructure* : Les réseaux ad-hoc se distinguent des autres réseaux mobiles par l'absence d'infrastructures préexistantes et de tout genre d'administration centralisée. Les hôtes mobiles sont responsables d'établir et de maintenir la connectivité du réseau d'une manière continue.

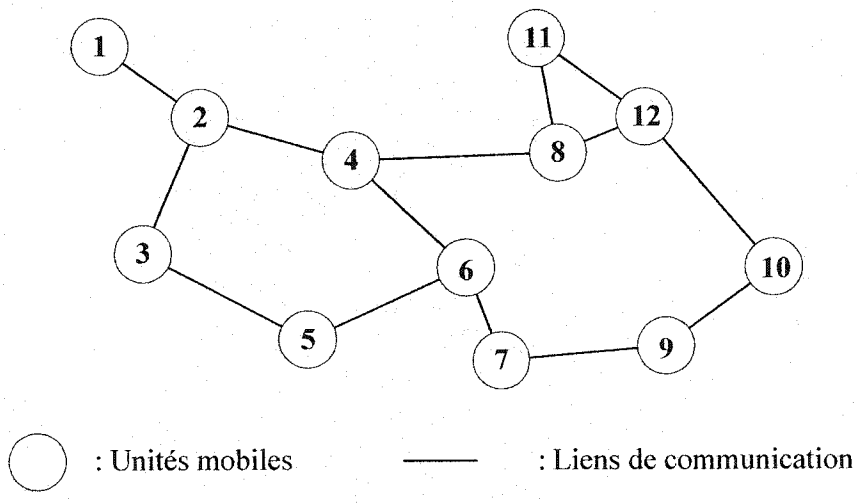


Figure 2.2 Modélisation d'un réseau ad-hoc

RFC 2386 définit la *qualité de service* comme un ensemble d'exigences que doit réunir un réseau lors de l'acheminement d'un flux de données de la source vers la

destination. La notion de qualité de service fait donc intervenir l'idée d'un contrat, d'une garantie : le réseau s'engage à fournir à l'utilisateur un ensemble d'attributs de service en terme de délai de transmission, variance de délai, bande passante disponible, probabilité de perte de paquet, etc.

La qualité de service n'a de sens que pour un flot de paquets entre une source et une destination, elle est donc liée à la notion de connexion logique. Le seul moyen de garantir une qualité de service pour une connexion donnée est d'avoir de bonnes techniques de réservation de ressources. Notons enfin que la qualité de service demandée est spécifique de l'application utilisée.

On dit qu'un réseau ad-hoc est *combinatoirement stable* lorsque la topologie du réseau change suffisamment lentement pour permettre une bonne propagation des mises à jours à travers le réseau. Avoir un réseau ad-hoc combinatoirement stable est nécessaire pour pouvoir parler de QdS. Un contre-exemple serait un réseau où les mises à jours sont trop lentes à être propagées à travers le réseau si bien que des chemins continuent à être utilisés pour une connexion entre une source et sa destination alors que ces derniers ne répondent plus aux exigences de QdS de la connexion ou pire qu'un lien de la connexion est rompu.

Un réseau ad-hoc est qualifié de *QdS-robust*, pour une QdS donnée si la QdS est maintenue et garantie à tout instant quels que soient les changements de topologies. De même, un réseau ad-hoc est QdS-*preserving* si la QdS est garantie à tout instant sauf lors des mises à jour de topologie. Un réseau ad-hoc *QdS-robust* est donc forcément *QdS-preserving*; l'inverse est faux.

2.2 Prise en compte de la qualité de service dans les réseaux ad-hoc

Dans [CHA 02], les différentes méthodes de prises en compte de la qualité de service sont classées en quatre grandes catégories : les modèles de qualité de services définissent des architectures globales dans lesquelles des garanties peuvent être fournies; les protocoles d'accès au médium cherchent à ajouter des fonctionnalités aux couches basses du modèles OSI afin de pouvoir offrir des garanties; les protocoles de

signalisation cherchent à offrir des mécanismes de réservation de ressources indépendants du protocole de routage sous jacent; les protocoles de routage avec qualité de service recherchent les routes ayant suffisamment de ressources disponibles pour satisfaire une requête. Nous évoquons dans cette partie les trois premières catégories. La quatrième catégorie fait l'objet d'une description approfondie au paragraphe suivant.

2.2.1 Modèles de qualité de service

Un modèle de qualité de service définit quels types de services sont offerts dans le réseau ainsi que les mécanismes mis en œuvre pour assurer ces services. En particulier, il définit quelles fonctionnalités doit fournir le protocole de routage, quelle est l'architecture des nœuds, etc. *IntServ / RSVP* [BRA 94] et *DiffServ* [BLA 98] sont les modèles de QdS proposés par l'IETF pour les réseaux filaires. Cependant, des études montrent que ces derniers sont peu adaptés aux contraintes des réseaux ad-hoc. *IntServ* requiert en effet un volume de traitement important, ce qui engendre des problèmes de consommation dans les mobiles. De plus, la signalisation de type RSVP n'est pas adaptée à ce type de réseaux car trop volumineuse par rapport à la bande passante limitée des réseaux sans fil. Enfin, le processus de maintenance des routes n'est pas adapté au caractère dynamique des réseaux ad-hoc. Le modèle *DiffServ* par contre, bien que conçu pour des cœurs de réseaux possédant une bande passante importante et dont la topologie est relativement statique, semble plus adapté aux caractéristiques des réseaux ad-hoc.

FQMM

FQMM [XIA 00] est un modèle de QdS situé entre les approches *IntServ / RSVP* [BRA 94] et *DiffServ* [BLA 98] pour gérer la QdS dans les réseaux filaires. Il a été conçu dans l'optique de facilité l'interfaçage entre les réseaux ad-hoc et le monde filaire. On considère ici des réseaux constitués au maximum d'une cinquantaine d'unités mobiles. FQMM définit plusieurs classes de service allant de la classe sans contraintes

jusqu'à celle permettant de spécifier pour chaque flux les contraintes spécifiques. Comme pour *DiffServ*, on définit trois types de nœuds :

- les nœuds d'entrée (émetteurs),
- les nœuds intermédiaires,
- les nœuds de sortie (récepteurs).

Du fait même de la nature des réseaux ad-hoc, chaque mobile est amené à assumer, parfois simultanément, différents rôles et ce selon le flux considéré. Les nœuds émetteurs sont responsables du conditionnement du trafic (lissage, marquage, etc). Par ailleurs *FQMM* présuppose l'utilisation d'un protocole de routage capable d'offrir une certaine qualité de service, c'est à dire capable de rechercher des routes satisfaisant certaines contraintes.

Par son approche hybride, *FQMM* entend résoudre certains problèmes liés aux modèles filaires et surtout faciliter l'interfaçage avec ces derniers. Cependant, si les problèmes d'extensibilité du modèle *IntServ* semblent pouvoir être résolus, la résolution de la plupart des problèmes liés au fonctionnement ad-hoc (volume de signalisation, consommation d'énergie, bande passante limitée et difficile à estimer) est laissée à la charge du protocole de routage sous-jacent. *FQMM* se contente donc de déplacer le problème de la QdS.

2.2.2 Couche MAC

Le protocole d'accès au médium radio est chargé d'éviter les collisions, d'assurer le partage de la bande passante et de résoudre certains problèmes spécifiques aux transmissions hertziennes comme le problème de stations cachées. Il a de ce fait un rôle primordial dans les réseaux ad-hoc. Cependant beaucoup de protocoles de routage avec qualité de service pourraient tirer parti de protocoles de niveau 2 capables de gérer certaines contraintes de QdS.

Différenciation de services pour IEEE 802.11

Dans [AAD 01], les auteurs proposent d'adapter certains paramètres de la fonction de coordination distribuée (DCF) du protocole IEEE 802.11 selon la priorité des paquets. Il devient alors possible de doter IEEE 802.11 d'un mécanisme de priorités entre les trames et donc de concevoir des mécanismes de différenciation de services efficaces. La fonction de coordination distribuée repose sur la détection de porteuse (CSMA). Avant d'émettre sur le médium, tout nœud doit s'assurer que le canal radio est libre depuis un certain temps (DIFS - DCF Inter Frame Spacing), afin de privilégier certains paquets de signalisation dont la transmission peut s'effectuer dès que le médium a été libre durant un temps SIFS (Short Inter Prame Spacing) plus court que le DIFS. On ajoute au DIFS, constant, un délai supplémentaire aléatoire permettant d'éviter que deux mobiles ne commencent à émettre au même moment. Dans ce cas, si une collision survient, le processus est réinitialisé et le délai aléatoire supplémentaire est allongé.

Un certain nombre de ces paramètres peuvent être adaptés dynamiquement afin d'offrir un mécanisme de priorités au protocole 802.11 :

- Lorsqu'une collision survient, les délais avant retransmission sont allongés aléatoirement. Il est possible d'incrémenter ces délais différemment selon le niveau de priorité.
- Il est possible d'utiliser différentes valeurs du délai de silence avant une transmission (DIFS) selon le niveau de priorité de la transmission.
- Enfin, il est possible de limiter la longueur des trames selon le niveau de priorité, les trames peu prioritaires occupant le canal moins longtemps.

Les trois principes ont été testés sur des flots UDP et TCP. De ces trois méthodes, la deuxième, consistant à jouer sur le délai DIFS, semble la plus stable et la plus performante.

MACA / PR

Le protocole MACA/PR (*Multiple Access Collision Avoidance with Piggyback Reservation*) introduit dans [GER 97] propose de différencier la politique d'accès au médium selon la nature des flux :

- Les paquets des flux sans aucunes contraintes de QdS sont traités de façon standard ;
- Pour les flux temps réel, une première demande d'autorisation à transmettre (échange RTS CTS) est effectuée en début de flux. Tous les paquets suivants sont ensuite transmis directement et doivent être acquittés par le récepteur. Dès qu'un paquet n'est pas acquitté, une nouvelle demande d'autorisation est émise.

Afin de traiter les réservations de bande passante, l'émetteur inclut des informations dans chaque paquet sur l'ordonnancement du paquet suivant. Tous les voisins du nœud récepteur, en écoutant l'acquittement d'un paquet de données possèdent des informations sur la date d'arrivée du prochain paquet et peuvent différer leurs transmissions. Ce mécanisme présente l'avantage de permettre de résoudre le problème des stations cachées sans avoir recours à des paquets de signalisation particuliers.

2.2.3 Protocoles de signalisation

Les protocoles de signalisation servent à fournir un moyen de propager des informations de contrôle à travers un réseau. Les informations transmises peuvent être de différentes natures : Il peut s'agir d'informations topologiques, de demandes de recherche de routes satisfaisant certaines contraintes ou encore de rapports sur l'état du réseau et la disponibilité des ressources. Concevoir un protocole de signalisation consiste à définir les données à échanger afin de réaliser une tâche particulière ainsi que la manière de les échanger.

INSIGNIA

INSIGNIA [LEE 2000] est un protocole de signalisation in-band (la signalisation est placée dans les entêtes des paquets de données) permettant d'effectuer des réservations de bande passante dans les réseaux ad-hoc. Il offre des garanties sur la base d'une granularité par flot aux applications adaptatives capables de modifier leur comportement en fonction de la quantité de bande passante qui leur est allouée. Chaque application spécifie deux niveaux de qualité de service. Le niveau de base permet de spécifier la bande passante minimale nécessaire au trafic et le niveau amélioré le débit optimal à atteindre lorsque les ressources sont disponibles. Ce protocole a été conçu pour réagir rapidement aux changements de topologie. *INSIGNIA* n'est pas lié à un protocole de routage particulier.

Les informations transmises par *INSIGNIA* sont incluses dans chaque paquet de données, sous la forme d'une option de l'entête IPv4. Ce champ de 26 bits indique si un paquet fait partie d'un flux privilégié ou non, le niveau de qualité de service requis par l'application adaptative émettrice ainsi que l'importance de chaque paquet dans le flux de données. Au départ, le champ est rempli par l'émetteur du flux de données. Il pourra être modifié tout au long du chemin afin d'acheminer des informations sur l'état actuel du réseau jusqu'au destinataire. Des rapports sont émis périodiquement par le récepteur d'un flux afin de permettre à l'émetteur d'adapter son transfert à l'état de la route. Les demandes de réservation de bande passante sont effectuée dans l'entête du premier paquet du flux. De plus, *INSIGNIA* comporte un mécanisme de reconstruction locale de routes.

INSIGNIA offre des performances encourageantes pour des réseaux dans lesquels la mobilité est moyenne. L'innovation de ce protocole est d'inclure une grande partie de la signalisation dans les paquets de données, ce qui réduit le nombre de contentions pour l'accès au médium.

2.3 Routage et routage avec qualité de service

Le routage est le processus d'acheminement des informations à la bonne destination à travers un réseau de connexions donné. Le problème du routage consiste à déterminer l'acheminement optimal des paquets à travers le réseau au sens d'un certain critère de performance. Il faut trouver l'investissement de moindre coût en capacités nominales et de réserves qui assure le bon écoulement du trafic nominal et garantit sa survivabilité quelle que soit la panne.

D'après les contraintes des réseaux ad-hoc évoquées plus haut, il découle que la construction des routes doit se faire avec un minimum de contrôle et de consommation de la bande passante. On peut subdiviser les protocoles de routage en deux catégories :

- Les protocoles de type *Best-effort* où le principe est de minimiser l'utilisation et surtout le gaspillage des ressources du réseau ;
- Les protocoles tenant compte de la qualité de service pour lesquels le critère est d'obtenir une connexion répondant à certains critères quitte à gaspiller un peu les ressources du réseau.

2.3.1 Les protocoles de routage de type *Best-effort*

Beaucoup de protocoles de routage de type *Best-effort* ont été publiés dans la littérature. Nous pouvons les classer en deux groupes :

- Les protocoles « proactifs » où la recherche de route se fait à l'avance en se basant sur l'échange périodique des tables de routage ;
- Les protocoles « réactifs » pour lesquels la recherche de routes se fait à la demande.

Plutôt que d'effectuer une énumération exhaustive des protocoles existants (indigeste et de toute façon impossible étant donné la masse de littérature), nous préférons aborder dans le détail les protocoles qui nous ont semblé les plus intéressants.

Un protocole proactif : DSDV

Cet algorithme de routage dont les initiales correspondent à "Vecteur de Distance à Destination Dynamique Séquencée" (en anglais : Dynamic Destination-Sequenced Distance- Vector) a été conçu spécialement par Perkins pour les réseaux mobiles. Il appartient à la classe des protocoles proactifs et utilise une version de l'algorithme distribué de Bellman-Ford adaptée au réseaux ad-hoc. Chaque station mobile maintient une table de routage qui regroupe l'ensemble des nœuds du réseau et pour chacun de ces nœuds :

- Le nombre de sauts nécessaires pour l'atteindre,
- Le numéro de séquence (NS) correspondant au nœud destination. Ce numéro est utilisé pour faire la distinction entre les anciennes routes et les routes plus récentes. Il permet donc d'éviter la formation de "boucles de routage".

Lorsqu'une station mobile souhaite initier une communication avec la destination, il lui suffit d'utiliser ses tables de routages pour avoir un chemin vers la destination. Afin d'assurer le rafraîchissement des tables de routage dans un contexte de topologie dynamique, chaque nœud du réseau émet régulièrement une mise à jour de sa table de routage destinée à l'ensemble de ses voisins directs. La mise à jour dépend de deux critères : le temps, c'est à dire la périodicité des mises à jours et les événements qui ont lieu comme l'apparition d'un nœud, la disparition d'un lien, etc. Son but est de permettre à une unité mobile de toujours être en mesure de localiser n'importe quelle autre unité du réseau. Elle peut se réaliser soit par *mise à jour complète*, s*oit par mise à jour incrémentale*.

Dans la mise à jour complète, la station transmet la totalité de la table de routage aux voisins. Elle nécessite donc l'envoi de plusieurs paquets de données. Au contraire, dans une mise à jour incrémentale, seules les entrées qui ont subit un changement par rapport à la dernière mise à jour sont envoyées. On limite ainsi la bande passante consommée par la mise à jour. Le choix du type de mise à jour est lié à la stabilité du réseau. Pour un réseau stable, la mise à jour incrémentale prédomine. La mise à jour complète n'est quant à elle utilisée que pour l'initialisation et occasionnellement pour

s'assurer de la consistance des tables. Dans le cas opposé d'un réseau à forte mobilité, la mise à jour complète est fréquente. En effet, les mises à jours incrémentales deviennent trop répétitives.

Un paquet de mise à jour contient le numéro de séquence incrémenté du nœud émetteur, l'adresse de chaque nœud destination "mis à jour" par le paquet, le nombre de nœuds séparant chaque destination et le numéro de séquence le plus récent connu de chaque destination. À la réception d'un paquet de données de mise à jour, chaque unité mobile compare les données de routage reçues à celles déjà disponibles. Les routes étiquetées par la plus grande valeur du numéro de séquence, c'est à dire les plus récentes, sont utilisées. Les autres route disponibles pour une même destination sont écrasées. Dans le cas où deux routes ont le même numéro de séquence, on choisit celle qui possède la meilleure métrique. La métrique utilisée dans le calcul des plus courts chemins est le nombre de nœuds existant dans le chemin. Les valeurs des métriques des routes, choisies après réception des données de routage, doivent être incrémentées. Les modifications faites sur les données de routage sont ensuite regroupées sous la forme d'un paquet de mise à jour destiné à être rediffusé à l'ensemble des nœuds voisins pour propager la mise à jour. Il faut cependant incrémenter les métriques des routes reçues avant l'envoi des mises à jours, car un nœud supplémentaire participe à l'acheminement des messages vers la destination. Un lien rompu est matérialisé par une valeur infinie de sa métrique, c'est à dire une valeur plus grande que la valeur maximale permise par la métrique.

La principale faiblesse de ce protocole, imputable à sa pro-activité, est de générer un fort trafic de routage peu compatible avec les limitations des réseaux ad-hoc. Par ailleurs, le protocole DSDV est lent. En effet, une unité mobile doit attendre d'avoir reçu la prochaine mise à jour initiée par la destination pour mettre à jour l'entrée correspondante dans la table de distance.

Le protocole de routage DSR

Le protocole "Routage à Source Dynamique" (en anglais : Dynamic Source Routing), utilise la technique de routage source qui consiste à déterminer, à la réception d'une demande de connexion pour une destination donnée, la séquence complète des nœuds à travers lesquels les paquets de données seront acheminés jusqu'à la destination. Il s'agit donc d'un protocole réactif.

Pour envoyer des données à un nœud destination, l'émetteur détermine d'abord une route vers la destination et l'inclut dans l'entête de chaque paquet de données. La spécification de cette route est effectuée en insérant l'adresse de tous les nœuds à travers lesquels le paquet doit transiter avant d'atteindre la destination. Par la suite, l'émetteur transmet le paquet au premier nœud spécifié dans la route source, à l'aide de son interface radio. Chaque nœud qui reçoit le paquet, et qui diffère de la destination, supprime son adresse de l'entête du paquet reçu et le retransmet au nœud suivant identifié dans la route source. Ce mécanisme se renouvelle jusqu'à ce que le paquet atteigne la destination finale où il est envoyé à la couche supérieure.

Le protocole DSR s'axe autour de deux opérations distinctes : la découverte de routes et la maintenance de routes. L'opération de découverte de routes permet à un nœud du réseau ad-hoc de découvrir dynamiquement un chemin menant à un nœud donné. Le nœud initiateur de cette opération, diffuse un paquet de requête de route qui identifie la destination et qui se propage dans le réseau. Si l'opération de découverte est réussie, l'hôte initiateur reçoit un paquet réponse de route qui liste la séquence de nœuds permettant d'atteindre la destination. Le paquet requête de route contient :

- L'adresse de l'initiateur de la requête,
- Un champ route, qui accumule la séquence des nœuds visités durant la propagation de la requête de route dans le réseau comme illustré dans la Figure 2.3,
- Un identificateur unique de la requête utilisé pour détecter les duplications de réceptions de la requête de route. Chaque nœud du réseau ad-hoc maintient une

liste de couples <adresse de l'initiateur, identificateur de requête> des requêtes récemment reçues.

À la réception d'un paquet requête de route par un nœud *p* du réseau, le traitement suivant est effectué :

- Si le couple <adresse de l'initiateur, identificateur de requête> du paquet reçu, existe déjà dans la liste des requêtes récemment reçues ou encore si l'adresse de *p* existe dans le champ enregistrement de route du paquet de la requête, le paquet est ignoré.
- Dans le cas contraire, si l'adresse de *p* et celle de la destination sont identiques, alors l'enregistrement de route contient le chemin à travers lequel le paquet de la requête est passé avant d'atteindre le nœud *p*. Une copie de ce chemin est envoyée dans un paquet réponse de route à l'initiateur de la requête.
- Sinon, l'adresse de *p* est rajoutée dans l'enregistrement de route du paquet et ce dernier est rediffusé.

Le paquet de requête de route se propage à travers le réseau en suivant ce mécanisme jusqu'à atteindre le nœud destination seul apte à répondre à la source. Précisons au passage qu'en ignorant la requête, dans le cas où l'adresse du récepteur existe dans l'enregistrement de route, on empêche la formation de boucles de nœuds dans le chemin découvert.

La Figure 2.3 (a) donne une illustration de la phase de construction de l'enregistrement de route. Le nœud source propage un paquet de requête de route vers la destination : le nœud 7. Ce paquet est d'abord reçu par les nœuds 10, 3 et 2. Ces nœuds retransmettent le paquet après avoir pris soin d'ajouter leur adresse dans le paquet d'enregistrement de route. On obtient alors respectivement les paquets suivants : [1,10], [1,3] et [1,2]. Les nœuds 4, 5 et 9 reçoivent les paquets et le processus ce renouvelle jusqu'à atteindre la destination.

À la fin de la phase de construction de route, l'hôte destination doit disposer d'un chemin vers l'initiateur pour lui retourner le paquet réponse de route et lui signifier ainsi qu'une route est à présent disponible. Dans le cas où la destination ne dispose pas dans

ses tables d'une telle route, le chemin spécifié dans l'enregistrement de route contenu dans le paquet requête de route peut être inversé et utilisé. Cependant, cela exige que les liens entre les nœuds dans le chemin soient bidirectionnels. Ainsi, dans la Figure 2.3 (b), la destination (le nœud 7) propage un paquet réponse de route vers la source (le nœud 1) le long de la route enregistrée. Il transite par les nœuds 6, 5 puis 3.

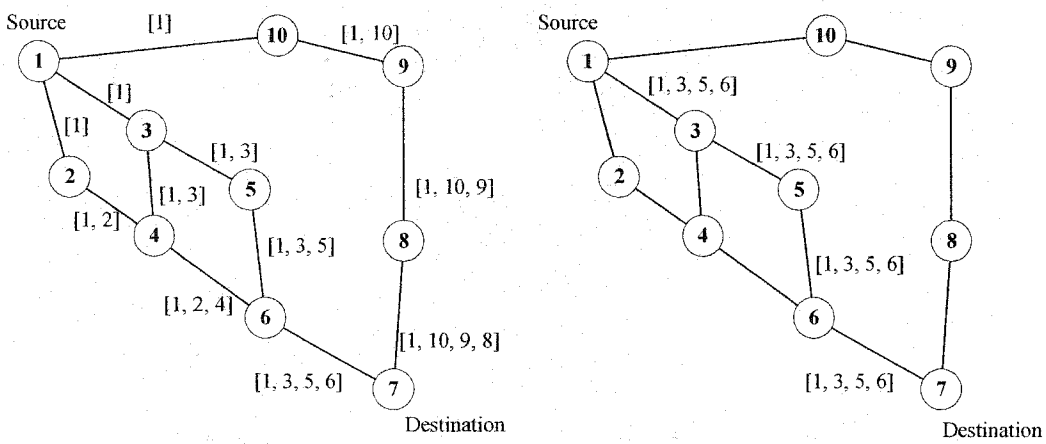


Figure 2.3 Découverte de chemins dans DSR

Dans le but de réduire le coût en temps et en bande passante ainsi que la fréquence des opérations de découverte de routes, chaque nœud doit garder en mémoire les chemins contenus dans les message de routage qu'il reçoit. Ces données seront utilisables jusqu'à ce qu'elles soient invalides.

Contrairement à DSDV, dans DSR l'opération de découverte de routes n'est pas intégrée avec celle de maintenance. Il faut donc exécuter une procédure de maintenance de routes pour assurer la validité des chemins utilisés. Ainsi, quand un nœud détecte un problème de transmission, à l'aide de sa couche de liaison, un message erreur de route est envoyé à la source. Le message d'erreur stipule l'adresse du nœud qui a détecté l'erreur et celle du nœud qui le suit dans le chemin. À la réception du paquet erreur de route par l'hôte source, le nœud concerné par l'erreur est supprimé du chemin sauvegardé, et tous les chemins qui contiennent ce nœud sont tronqué à son niveau. Pour

continuer la communication, une nouvelle opération de découverte de routes vers la destination doit alors être initiée par l'émetteur.

Le protocole de routage AODV

L'algorithme de routage AODV (Ad-hoc On Demand Vector) représente une amélioration de DSDV [PER 99], [PER 2000]. En fait, pour synthétiser, il reprend les avantages de DSDV mais limite la consommation de bande passante du fait de sa réactivité. En effet, il fonctionne *à la demande* c'est-à-dire qu'il ne construit des routes que sur demande des nœuds sources. Il maintient ces routes aussi longtemps que les sources le nécessitent. Il est capable à la fois de routage *unicast* et *multicast*. De plus, AODV constitue des arborescences connectant les membres des groupes *multicast*. Les arbres sont composés des membres des groupes et des nœuds nécessaires pour connecter les membres. AODV utilise une numéro de séquence pour assurer la "fraîcheur" des routes. Il ne fait pas de boucle, est auto-démarrant et s'accommode d'un grand nombre de nœuds mobiles.

Comme DSR, AODV construit les routes par l'emploi d'un cycle de requêtes "Route *Request* /Route *Reply*". Lorsqu'un nœud source désire établir une route vers une destination pour laquelle il ne possède pas encore de route, il diffuse un paquet *Route Request* (RREQ) à travers le réseau. Les nœuds recevant le paquet mettent à jour leur information relative à la source et établissent des pointeurs de retour vers la source dans les tables de routage. Outre l'IP de la source, le numéro de séquence courant et l'*ID de broadcast*, le RREQ contient également le numéro de séquence de la destination le plus récent connu de la source. Un nœud recevant un RREQ émet un paquet *Route Reply* (RREP) soit s'il est la destination, soit s'il possède une route vers la destination avec un numéro de séquence supérieur ou égal à celui repris dans le RREQ. Si tel est le cas, il envoie un paquet RREP vers la source. Sinon, il diffuse le RREQ. Les nœuds conservent chacun trace des IP sources et des *ID de broadcast* des RREQ. S'ils reçoivent un RREQ qu'ils ont déjà traité, ils l'écartent et ne le retransmettent pas. Les nœuds établissent des pointeurs de propagation vers la destination alors que les RREP reviennent vers la

source. Une fois que la source a reçu les RREP, elle peut commencer à émettre des paquets de données vers la destination. Si, ultérieurement, la source reçoit un RREP contenant un numéro de séquence supérieur ou le même mais avec un compte de *hop* plus petit, elle mettra à jour son information de routage vers cette destination et commencera a utiliser la meilleure route.

Une route est maintenue aussi longtemps qu'elle continue à être active. C'est à dire tant que des paquets de données transitent périodiquement de la source à la destination selon ce chemin. Lorsque la source arrête d'émettre des paquets de données, le lien expire et est effacé des tables de routages des nœuds intermédiaires. Si un lien est rompu alors qu'une route est active, le nœud extrémité du lien rompu émet un paquet r*oute error* (RERR) vers le nœud source pour le notifier que la destination est désormais inaccessible par cette route. Après réception du paquet RERR et si la source désire toujours communiquer avec la destination, elle initie un nouveau processus de découverte de route.

Les *routes multicast* sont établies de la même manière. Un nœud désireux de rejoindre un *groupe multicast* diffuse un RREQ avec l'IP de destination positionnée à celle du *groupe multicast* et le *flag* J (*join*) positionné pour indiquer la volonté de rejoindre le groupe. Tout nœud membre de l'*arborescence multicast* recevant cet RREQ et ayant un numéro de séquence assez frais est susceptible d'émettre un RREP. Les nœuds établissent des pointeurs de routage alors que les RREP voyagent vers la source.

À la réception des RREP, la source retient la route avec le numéro de séquence le plus récent ou à défaut avec le plus petit compte de nœuds. Après une période donnée de découverte, le nœud source émet un message d' *activation multicast* (MACT) au hop suivant sélectionné. Ce message active la route. Un nœud ayant préparé un *routage multicast* qui ne reçoit pas de MACT dans la période spécifiée expire et efface le pointeur. Si un nœud recevant le MACT ne fait pas partie de l'arborescence, il conserve la trace de la meilleure route pour les RREP reçus. Dès lors il émet également un MACT vers son prochain nœud. Ce message se propage jusqu'à atteindre un nœud de l'arborescence.

AODV maintient les routes aussi longtemps que celles-ci sont actives. Ceci inclut la maintenance d'une *arborescence multicast* aussi longtemps que le groupe est actif. Du fait de l'intermittence des nœuds, il est plus que probable que de nombreuses pertes de liens auront lieu le long de la route durant sa durée de vie.

2.3.2 Les protocoles de routage garantissant une qualité de service

Les protocoles de type *Best-effort* ne tiennent pas compte, dans la recherche de routes, de la qualité du service offert. Un protocole de routage tenant compte de la qualité de service permet d'établir le chemin d'une connexion entre une source et une destination en fonction de critères de qualité de service donnés. L'établissement du chemin entre la source et la destination se fait alors en deux étapes successives toutes deux prises en charge par le protocole de routage :

- **Découverte** du chemin convenable répondant aux exigences en terme de qualité de service ;
- **Réservation** des ressources le long du chemin découvert.

Pour finir, comme la réservation de ressources change la topologie du réseau (en particulier en terme de bande-passante disponible le long d'un lien), la nouvelle topologie doit être propagée aux autres nœuds.

Les deux premiers protocoles proposés dans cette partie s'appuient seulement sur l'état local maintenu dans chaque nœud. Aucune information sur l'état global ou sur la topologie globale du réseau n'est requise. Le troisième protocole proposé se base sur l'état global pour déterminer le routage des paquets. Suit pour finir la description de deux autres algorithmes de routage tenant compte de la qualité de service et basés sur des approches originales : prédiction de localisation et réservation de bande passante grâce au TDMA.

2.3.2.1 Algorithme de routage initié par la source

Il s'agit d'un protocole réactif. Lorsqu'une source veut joindre un nœud destination avec des contraintes de qualité de service données, elle initie le processus de

recherche de route en diffusant à ses voisins une demande de connexion. Chaque demande de connexion est identifiée par un numéro d'identification unique *id* (par exemple l'adresse IP de la source et un numéro de séquence). En effet, des demandes de connexion peuvent être soumises simultanément dans le réseau mobile. Leur numéro d'identification permet donc de les distinguer et d'éviter toute interférence parmi les requêtes. On s'intéressera dans la suite au processus de routage pour une requête unique.

Algorithme

À l'arrivée d'une requête de connexion, le nœud source initie le processus de routage en propageant des messages de routage à travers le réseau à tous ses voisins répondant aux exigences de QdS (par exemple une certaine bande passante). Une requête de connexion, notée *probe[id],* comporte au moins 3 champs : la destination, la bande passante requise, et le numéro d'identification de la connexion *id.*

Lorsqu'un nœud *i* reçoit sa première requête pour une connexion donnée, il la propage à tous ses voisins qui satisfont à l'exigence de qualité de service escomptée pour la connexion. Le nœud *i* ne tiendra alors plus compte des autres requêtes reçues pour la même connexion. On aura donc au plus une unique requête émise sur un lien donné. Comme chaque sonde se propage le long de liens répondant aux exigences de qualité de service, une requête qui parvient à la destination détecte un chemin répondant aux exigences de QdS recherchées par la source.

Le *prédécesseur* du nœud *i* est défini comme le nœud à partir duquel *i* a reçu la requête *probe[id]*. Lorsque *i* reçoit la première requête d'un nœud *k*, *i* enregistre *k* comme son prédécesseur. C'est un *soft state* qui sera effacé après un certain intervalle de temps (TTL).

Lorsque la destination reçoit la requête de connexion, elle retourne à la source un message de confirmation. Ce message emprunte en sens inverse le chemin parcouru par la requête parvenue à la destination et est chargé de la réservation des ressources appropriées correspondant à la QdS escomptée. Pour se maintenir sur le chemin déterminé par la sonde, le message de confirmation exploite l'identification du

prédécesseur du nœud courant enregistrée lors de la propagation de la sonde. Une fois le message de confirmation parvenue à la source, la connexion est établie.

Comme la topologie du réseau peut changer au cours de ce processus de routage, il se peut que le chemin découvert par la requête de connexion soit invalide avant la fin de l'étape de réservation par le message de confirmation. Dans ce cas, le message de confirmation n'arrivera jamais à la source. Il est alors redirigé vers la destination pour libérer les ressources de bande-passante déjà allouées à la connexion le sur les liens du côté en direction de la destination.

Propriétés

Le chemin sélectionné par le routage ne comporte pas de boucle. Dans le pire des cas, le temps nécessaire au routage est $0(2v)$ et le nombre de messages envoyés $0(e)$ où v est le nombre de nœuds et e le nombre de liens du réseau. Si la topologie du réseau et la bande passante de chaque lien restent inchangés au cours du processus de routage, l'algorithme trouve un chemin si un tel chemin existe.

Faiblesses et améliorations

Cet algorithme génère un trafic très important qui est particulièrement gênant dans un réseau ad-hoc où la bande-passante est très limitée. Dans [CHEN 99], S. Chen propose de limiter le trafic causé par l'algorithme en évitant la propagation de requêtes inutiles. En particulier, une fois qu'un nœud j a reçu sa première requête du nœud i, il n'est pas nécessaire qu'elle renvoie à i une requête pour la même connexion. En effet, cette dernière ne serait de toute façon pas prise en compte par le nœud i. Cette première solution permet de limiter le trafic inhérent au protocole de routage. Par ailleurs, et toujours pour limiter la surcharge du réseau, S. Chen propose d'utiliser un nouveau concept, le *routage multicast local*. Il s'agit d'utiliser la propriété de diffusion radio des réseaux ad-hoc. Au lieu d'envoyer une requête à chaque voisin vérifiant les exigences de QdS, le nœud i exécute un *multicast local* à tous ses voisins. Seuls les voisins recevant cette requête qui vérifient les exigences de QdS tiennent comptent du message ; les

autres voisins se contentent de l'effacer. Ce procédé permet de mieux rentabiliser le temps CPU du nœud *i* et donc d'améliorer la bande-passante. Par ailleurs, il limite aussi le nombre de requêtes envoyées au cours du processus de routage.

Une autre astuce proposée par S. Chen est utilisée pour encore optimiser le processus de routage. Il s'agit lors de l'émission de sondes de collecter de l'information sur la topologie au niveau de chaque nœud, en particulier la distance du nœud courant à un autre nœud. Cette donnée est stockée dans un cache pour une durée donnée et peut-être utilisée comme distance approximative en terme de nœuds pour une nouvelle opération de routage. Au bout d'un certain intervalle de temps, cette donnée est écrasée. A la réception d'une requête pour la destination *t*, le nœud *i* consulte alors son cache. Dans le cas, où il a une donnée de distance pour la destination *t*, un nouveau champ dans la sonde est initialisé : un compteur de distance pour la destination. Ce compteur est initialisé avec la valeur lue en cache (légèrement majorée pour faire face à une éventuelle modification de topologie du réseau depuis le routage précédent) et est décrémenté à chaque réception de la sonde par un nouveau nœud répondant à l'exigence de performance. Ce processus permet encore de limiter l'envoi de requêtes inutiles. En effet, lorsque le compteur de distance est nul, la sonde n'est plus retransmise même si le nœud courant n'est pas la destination.

Enfin, S. Chen propose une variante de son protocole basée sur la collecte d'information sur le voisinage des nœuds. En effet, si chaque nœud maintient une information complète sur son voisinage, le routage vers n'importe quel nœud du voisinage peut s'exécuter localement. Pour assurer la mise à jour dans chaque nœud de l'information sur l'état de son voisinage, chaque nœud propage régulièrement un message d'état qui comprend l'identification de l'ensemble de ses voisins, la bande passante disponible pour chaque lien, un *timestamp* et un champ TTL. Lorsqu'un nœud reçoit ce message d'état, il vérifie le *timestamp* du message pour vérifier s'il ne l'a pas déjà reçu. Le cas échéant, il l'écrase. Sinon, le nœud enregistre les informations du message, décrémente le champ TTL puis, dans le cas ou le champ TTL est non nul, le rediffuse à l'ensemble de ses voisins. Dans le cas contraire, le message est écrasé.

2.3.2.2 Algorithme de routage initié par la destination

À la différence de l'algorithme présenté plus haut, les requêtes chargées de découvrir un chemin répondant aux exigences de la connexion en terme de QdS sont cette fois émises à partir de la destination vers la source après une phase d'initialisation.

Initialisation

La phase d'initialisation a pour but de dégager une première estimation du nombre de nœuds qui séparent la source de la destination. Lorsqu'une demande de connexion parvient au nœud source *s*, un message de contrôle est acheminé à la destination à l'aide d'un protocole de routage de type best-effort (par exemple DSR).

Dès que la destination reçoit le message de contrôle, elle commence la recherche du chemin répondant aux exigences de qualité de service. Le message de contrôle envoyé par la source à la destination comporte un champ compteur initialisé à 0 et incrémenté à chaque passage par un nouveau nœud. Ce champ permet à l'arrivée à la destination d'avoir une estimation de la distance entre la source et la destination. Même si cette estimation s'avère imprécise, on peut l'utiliser pour limiter la portée des *broadcasts* utilisés ultérieurement. Lors de sa traversée du réseau entre la source et la destination, le message de contrôle peut aussi vérifier la bande-passante disponible sur chacun des liens traversés. Si le chemin traversé par le message de contrôle répond aux exigences de QdS de la connexion, il peut être utilisé directement sans étape ultérieure de *broadcast*. La destination renvoie alors un message de confirmation qui emprunte, en sens inverse, le même chemin que le message de contrôle. Ce message de confirmation se charge aussi de réserver la bande passante exigée le long des liens parcourus. Cela permettra en particulier de limiter l'engorgement du réseau dû au protocole de routage et surtout d'écourter le temps d'exécution de l'algorithme de routage.

Si l'algorithme de routage best-effort supporte un routage multi-chemins, la source peut propager plusieurs messages vers la destination. Chaque message utilise un chemin différent de telle sorte qu'on multiplie les chances d'avoir un chemin répondant

directement aux exigences de QdS tout en augmentant la précision de l'estimation de la distance en terme de nœuds entre la source et la destination.

Algorithme

Lorsque la destination reçoit le message de contrôle envoyé par la source, et que ce dernier n'a pas directement trouvé de chemin répondant aux exigences de QdS, la destination initie la recherche du chemin répondant à ces exigences en utilisant un *local multicast*. Une requête est construite. Elle comporte quatre champs : la source *s*, la bande passante requise B, l'identificateur de connexion et un champ TTL initialisé grâce à l'estimation de la distance source-destination fournie par le message de contrôle à sa réception à la destination légèrement majoré. La requête est ensuite propagée par *broadcast* à tous les nœuds voisins. Les nœuds qui ont la bande passante suffisante conservent le message et le re-propagent. Si ce n'est pas le cas, le message est effacé.
Considérons le cas général où un nœud *i* reçoit une requête par *broadcast local* envoyé par un nœud *j* où *j* peut être la destination *t* ou un nœud quelconque intermédiaire :

1. Si la bande passante disponible au niveau du lien (*i,j*) n'est pas suffisante, la requête n'est pas prise en compte par le nœud *i* et la propagation est stoppée à son niveau. Sinon, on continue la propagation vers les nœuds suivant.
2. Si *i* a déjà reçu un message sonde pour cette connexion, on n'en tient pas compte et le message est effacé. Sinon, deux cas peuvent se présenter : Si *i* est la source *s*, un chemin répondant aux exigences de la connexion en terme de bande passante est découvert. Un message de confirmation est renvoyé à la destination pour confirmer la connexion et surtout pour réserver la bande passante tout au long du chemin. Sinon, le champ TTL du message sonde est décrémenté. Dans le cas où le champ TTL est nul, le message sonde n'est pas pris en compte. Sinon, le nœud *i* le propage à ses voisins.

Propriétés

Comme pour le routage source, le chemin détecté en fin d'exécution de l'algorithme de routage est toujours sans boucle. Par ailleurs, il y a au plus un unique envoi de requête par nœud. Il est également à noter qu'une requête n'a pas à comporter de champ contenant l'ensemble des destinataires lors des *local broadcasts*. En effet, chaque récepteur d'une requête est lui-même apte à déterminer si le message lui est destiné (il utilise les informations dont il dispose sur son état local). Enfin, la portée de la recherche du chemin par les *broadcasts* successifs est limitée (en terme de nœuds) par l'estimation de la distance source-destination obtenue lors de la phase d'initialisation de l'algorithme.

2.3.2.3 Algorithme de routage basé sur les états globaux

Le protocole de routage basé sur les états globaux utilise la notion de *ticket-based probing* pour identifier une route entre la source et la destination.

Algorithme

Chaque nœud i maintient l'état local de tous les liens vers chaque nœud. Les informations sur l'état d'un lien (i,j) sont constituées par le délai entre i et j, la bande passante du lien, la bande passante résiduelle et le coût.

En plus de l'état local, chaque nœud maintient un état global. Les informations suivantes doivent être maintenues au nœud i pour chacune des destinations t.

1) Connectivité : Ri(t) est une table de routage comportant un ensemble de nœuds adjacents qui peuvent être utilisés pour relier t.
2) Délai : Di(t) représente le délai minimum de i à t.
3) Bande-passante : Bi(t) est la valeur de la bande passante maximale disponible, c'est à dire celle disponible sur le chemin de plus grande bande-passante.
4) Coût : Ci(t) représente la valeur du coût minimum entre i et t, c'est à dire le coût du chemin le moins onéreux entre i et t.

L'information est complétée par la donnée de deux variables pour mesurer l'imprécision sur Di(t) et sur Bi(t) : la *variation de délai (resp. variation de bande passante)* est une estimation de la variation de délai (*resp.* variation de délai) avant le rafraîchissement suivant. Ces informations peuvent être rafraîchies à l'aide d'un protocole *vecteur de distance.*

Principe du routage

Des messages de routage sont envoyés de la source *s* vers la destination *t* pour trouver le chemin qui répondra aux exigences de la communication en termes de bande passante ou de délai pour le coût le plus faible.

Des *tickets* sont générés au niveau de la source à l'initialisation du routage. Ces tickets sont affectés au message de routage. Le nombre de tickets est fonction des ressources du réseau. Chaque message se voit attribué au moins un ticket. De plus, à un instant donné, le nombre maximum de messages de routage dans le réseau pour une demande de connexion donnée est fixé par le nombre total de tickets. Comme chaque message de routage cherche un chemin, le nombre maximum de chemins cherchés correspond aussi au nombre de tickets. La Figure 2.4 donne une illustration de la répartition des tickets.

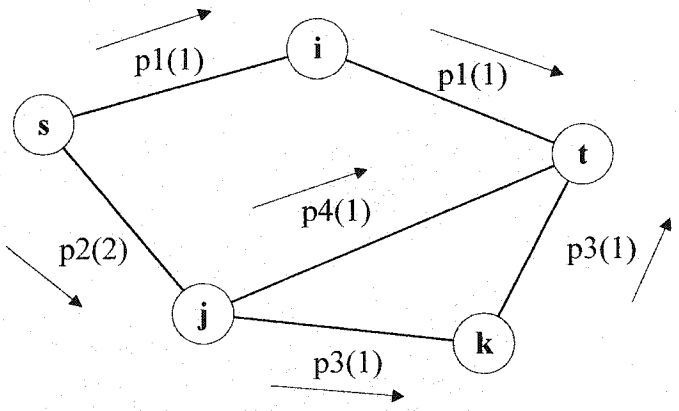


Figure 2.4 Répartition des tickets dans l'algorithme basé sur les états globaux

Deux messages de routage *p1* et *p2* sont envoyés à partir de la source *s*. Le nombre entre parenthèses à droite du message de routage indique le nombre de tickets

contenus dans chaque message. Au nœud *j*, *p2* est scindé en *p3* et *p4* contenant chacun un ticket. Il y a au plus 3 messages de routage à un instant donné. De plus, trois chemins sont cherchés $s \rightarrow i \rightarrow t$, $s \rightarrow j \rightarrow t$ et $s \rightarrow j \rightarrow k \rightarrow t$.

Le protocole de routage utilise l'état d'information aux différents nœuds intermédiaires pour guider les tickets et les messages qui les englobent le long des meilleurs chemins vers la destination de sorte de maximiser la probabilité de trouver un chemin répondant aux exigences attendues tout en garantissant le meilleur coût.

Ce concept de tickets présente un intérêt multiple :

- Le nombre de messages de routage envoyés au cours du routage pour une connexion donnée est limité par le nombre de tickets affectés au premier message de routage ce qui permet de faire un compromis entre la performance du routage et le gaspillage de bande passante par le processus de routage. Ainsi, lorsque les exigences en matière de QdS sont faciles à remplir d'après les informations disponibles, peu de tickets seront attribués au message de routage. Dans d'autres conditions plus hostiles on affectera plus de tickets au message de routage.
- Le schéma fonctionne avec des informations d'état imprécises. Le niveau d'incertitude a un impact direct sur le nombre de tickets affectés au début du routage.
- Ce protocole de routage considère non seulement les performances en termes de QdS mais aussi les exigences de coût. En effet, on donne une préférence à la découverte de chemins à faible coût lors de l'exécution de protocole de routage.

Sélection de route

Le processus de sélection de route peut se faire, au choix, selon deux critères : le délai entre la source et la destination ou la bande passante disponible. Dans cet exposé, nous nous bornerons au routage par contrainte de délai mais S. Chen a prévu également un algorithme similaire pour répondre à la contrainte sur la bande passante. Pour assurer le routage avec contrainte de délai, on introduit deux types de tickets. Les tickets sont

ainsi soit vers jaunes soit verts. Selon leur couleur, les tickets ont des fonctions différentes :

1) **Les tickets jaunes** ont pour rôle de maximiser la probabilité de trouver un chemin faisable. Ainsi, les tickets jaunes (ou plutôt les messages de routage les transportant) privilégient les chemins avec des faibles délais pour améliorer les chances de satisfaire les exigences pour un délai donné. Le nombre de tickets jaunes (Y_0) est déterminé par l'exigence de délai D pour la connexion. Si D est très grand et peut être très vraisemblablement vérifié, un unique ticket jaune sera suffisant pour trouver un chemin convenable. Si D est trop petit pour être obtenu, aucun ticket jaune n'est affecté, la demande de connexion est rejetée. Dans les autres cas, on affecte plus d'un unique jeton jaune pour trouver un chemin adéquat.
2) **Les tickets verts** servent à maximiser la probabilité de trouver un chemin peu onéreux. Les tickets verts (ou plutôt les messages de routage les transportant) préfèrent les chemins avec des faibles coûts qui peuvent cependant ne pas répondre aux exigences de QdS (en particulier ne pas vérifier l'exigence de délai D). Le nombre de tickets verts (G_0) est aussi fonction du délai requis pour la connexion.

L'idée sous-jacente est d'utiliser les tickets verts, plus agressifs, pour trouver des chemins peu onéreux avec une faible probabilité de répondre aux exigences de délai et d'utiliser les jetons jaunes comme une garantie pour trouver les chemins répondants aux exigences de délai. Le nombre de tickets jaunes et verts est fonction de la topologie du réseau.

Processus de diffusion des tickets

Si $Y_0 + G_0 = 0$, la requête de connexion est rejetée. Sinon, les messages de routage transportant les tickets sont envoyés de *s* vers la destination *t*. Un message de routage n'est propagé que lorsque le chemin en question a un délai inférieur à D. Ainsi,

en arrivant en t, un message de routage détecte un chemin qui répond aux exigences de délai de la demande de connexion.

Chaque message de routage accumule le délai total correspondant au chemin. Plus précisément, il comporte un champ délai noté $delay(p)$ pour un message de routage p. Ce champ est initialisé à 0. Chaque fois que le message de routage traverse un lien (i,j), le nœud qui reçoit le message met à jour le champ délai suivant (2.1).

$$delay(p) := delay(p) + delay(i,j) \qquad (2.1)$$

La distribution des tickets s'exécute comme suit : supposons qu'un nœud i reçoive un message de routage p avec $Y(p)$ tickets jaunes et $G(p)$ tickets verts en provenance du nœud k. On envoie le message de routage seulement aux nœuds qui appartiennent à l'ensemble $R_i(t)$ définit par (2.2).

$$R_i(t) = \{j \,/\, delay(p) + delay(i,j) + Dj(t) - \Delta Dj(t) < D\} \qquad (2.2)$$

En effet, les nœuds qui n'appartiennent pas à cet ensemble ne vérifient pas la contrainte de délai exigée par la connexion. Si cet ensemble est vide, aucun message de routage ne sera envoyé. Sinon, pour chaque élément j de l'ensemble, i fait une copie de p que l'on nomme p_j. Chaque p_j reçoit $Y(p_j)$ tickets jaunes et $G(p_j)$ tickets verts en respectant les contraintes (2.3) et (2.4).

$$\sum_{j \in R_i(t)} Y(p_j) = Y(p) \qquad (2.3)$$

$$\sum_{j \in R_i(t)} G(p_j) = G(p) \qquad (2.4)$$

Nous n'évoquerons pas dans cette présentation la stratégie utilisée pour la distribution des tickets jaunes et verts aux nœuds voisins. Nous nous bornerons à préciser que cette stratégie cherche à maximiser la probabilité de découvrir un chemin répondant à l'exigence de délai pour la connexion.

Détection de la terminaison et sélection du chemin

Le processus de routage est terminé lorsque tous les messages de routage ont soit atteint le nœud destination soit été stoppés par des nœuds intermédiaires. Pour détecter

la terminaison, les nœuds intermédiaires envoient des messages d'invalidation à la destination *t* au lieu de se contenter d'effacer les messages de routage. De ce fait, tous les tickets arriveront à la destination. Le processus de routage est terminé lorsque la destination *t* a reçu tous les tickets. On utilise un *timeout* pour résoudre les problèmes causés par la perte de tickets. Si la destination ne réceptionne que des messages d'invalidation, *t* envoie un message à la source *s* pour informer que la demande de connexion a été rejetée.

Lorsqu'un message de routage parvient à la destination avec un ticket valide, un chemin répondant à l'exigence de délai pour la connexion a été trouvé. Ce chemin est l'un de ceux traversés par le message. Il existe deux approches pour enregistrer le chemin : la première consiste à enregistrer le chemin dans le message de routage lui-même; la deuxième consiste à enregistrer le chemin dans les nœuds intermédiaires sur une base *nœud par nœud.* La première approche requiert des messages de routage de plus grande taille, ils consomment donc plus de bande-passante et plus de mémoire quand ils sont en attente dans les queues.

Notons également qu'un message de routage comptabilise le coût du chemin qu'il traverse. Si plusieurs messages de routage parviennent à la destination *t*, on pourra donc sélectionner celui de moindre coût.

La sélection du chemin peut se faire avant que le processus de routage ne soit terminé afin de réduire le temps de routage. Lorsque *t* reçoit un message de routage, il vérifie le coût de la route figurant dans le message. Si la qualité du chemin est bonne, le chemin est sélectionné immédiatement et un message de confirmation est envoyé à la source. Sinon, *t* attend jusqu'à ce qu'un message de routage avec un meilleur coût lui parvienne ou alors jusqu'à la fin du processus de routage.

2.3.2.4 Routage basé sur une prédiction de localisation

Du fait de la mobilité de tous les nœuds dans les réseaux ad-hoc, les informations disponibles sur l'état aussi bien local que global deviennent très vite obsolètes. Il faut donc les rafraîchir régulièrement ce qui est très gourmand en bande passante.

L'approche de Samarth H. Shah et Klara Nahrstedt [SHA 02] se base sur ce constat pour limiter le gaspillage de bande passante causé par le rafraîchissement des informations d'état dans les réseaux ad-hoc. L'idée sous-jacente est d'essayer de prévoir l'évolution future de la position des nœuds à l'aide d'informations sur leur position jusqu'à un instant donné. Cet algorithme permet donc de commencer une opération de maintenance de route (recherche de nouvelles routes) avant même qu'une route soient effectivement brisée. Surtout, il prend en compte la QdS pour réduire les délais excessifs dus à un re-routage mis en œuvre trop tardivement.

2.3.2.5 Protocole de routage basé sur TDMA

Ce protocole exposé dans [LIA 02] est un protocole de routage sur demande. Il est basé sur un routage source et fonctionne de manière similaire au protocole DSR décrit plus haut en ce qui concerne la recherche de routes mais il prend aussi en compte la bande passante disponible.

Un nœud source *s*, qui veut établir une connexion avec le nœud destination *d*, envoie un paquet requête QREQ(..., b, PATH, NH) à ses voisins. Le champ PATH, contient l'information concernant la route partielle vers la destination et les *time slots* disponibles le long de ce chemin. NH correspond à la liste des nœuds qui peuvent être utilisés comme nœuds suivants pour étendre le chemin d'un nœud vers la destination. Un nœud *x* appartenant à NH qui reçoit QREQ pour la première fois transmet le paquet à ses voisins s'il a suffisamment de *time slots* disponibles. Il est à noter que dans le paquet envoyé par *x*, les informations sont mises à jour au préalable en ajoutant les information du nœud *x* dans les divers champs de QREQ. Lorsque le message QREQ parvient à la destination *d*, elle renvoie à la source *s* un paquet QREP(PATH). Ce paquet est routé à travers le réseau par le champ PATH et se charge de réserver les *time slots* le long du chemin.

On choisit les time slots dans les nœuds intermédiaires grâce au lemme suivant :

1. *Un time slot t peut être utilisé par un hôte x pour envoyer vers un nœud y sans causer de collision si les conditions suivantes sont satisfaites.*
2. *le slot t n'est pas encore réservé ni pour envoyer ni pour recevoir ni dans x ni dans y.*
3. *pour n'importe quel voisin direct z de x, le slot t n'est pas réservé pour recevoir en z.*
4. *pour n'importe quel voisin direct z de x, le slot t n'est pas réservé pour envoyer en z.*

Par exemple, sur la Figure 2.5, le nœud 4 cherche un *time slot* disponible pour transmettre vers le nœud 2. Les slots #1 et #2 ne peuvent pas être considérés parce que le slot #1 est utilisé par 4 pour envoyer et slot #2 est utilisé par 2 pour recevoir. Ensuite, les slots #3 et #4 ne peuvent pas non plus être utilisés parce qu'ils feraient des collisions avec 3 et 5. Pour finir, les slots #5 et #6 ne peuvent pas être utilisés car 1 et 3 envoient déjà sur ces slots. On conclut que seul le slot #7 peut être utilisé.

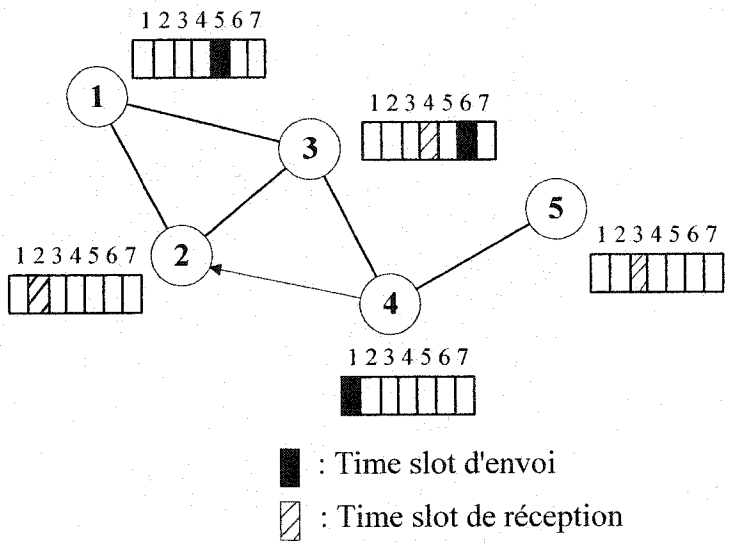


Figure 2.5 Affectation des slots dans l'algorithme de routage basé sur TDMA

Conclusion

Dans ce chapitre, nous avons introduit les réseaux ad hoc en mettant en valeur leurs avantages par rapport aux autres types de réseaux. Puis nous avons défini la notion de qualité de service et son importance croissante dans nombre d'applications. Enfin, nous nous sommes penchés sur le problème du routage. Nous avons ainsi dans un premier temps considéré les protocoles de routages de type Best-effort sur lesquels beaucoup de travaux ont été entrepris. Dans un deuxième temps, notre intérêt s'est focalisé sur les protocoles de routage tenant compte de la qualité de service dans les réseaux ad-hoc. Force est de constater que si certaines approches tenant compte de la qualité de service ont été publiées dans la littérature, aucune ne résout complètement le problème de la qualité de service dans les réseaux ad-hoc. Car ce problème est compliqué par les caractéristiques intrinsèques des réseaux ad-hoc : comme le fait remarquer Mishra dans [CHA 01], si les unités sont trop mobiles, il est impossible de garantir une quelconque qualité de service. De plus, le problème du routage est rendu ardu par le manque d'informations précises disponibles sur l'état du réseau tant au niveau local que global. Enfin, la taille du réseau devient vite un problème insurmontable pour les délais de propagation des mises à jour.

Tous ces problèmes sans solution mettent en exergue que le routage avec exigence de qualité de service dans les réseaux ad-hoc demeure un champ libre aux investigations scientifiques. Il représente un nouveau challenge à surmonter pour permettre aux réseaux ad-hoc de supporter les applications multimédia offertes par les réseaux fixes tout en autorisant la meilleure flexibilité, adaptabilité et surtout mobilité propre à ce type de réseau.

CHAPITRE III
PROTOCOLE DE DÉCOUVERTE DE ROUTES PROPOSÉ

Dans ce chapitre, nous exposons le principe ainsi que les mécanismes que nous avons imaginés en vue d'améliorer les communications avec des exigences en terme de qualité de service dans les réseaux mobiles ad-hoc. L'amélioration que nous apportons se situe au niveau du protocole de routage et consiste à tenir compte du caractère plus ou moins facilement réparable d'une route pour choisir, parmi les routes disponibles à la fin de la phase de découverte de route, celle qui sera utilisée pour la communication. La route sélectionnée est alors celle dont la réparation est le plus rapide. Nous rappellerons d'abord les bases du protocole de routage sur lequel nous comptons nous appuyer, puis nous exposons en détail l'amélioration que nous souhaitons lui apporter.

3.1 Le protocole initial

Il s'agit du protocole de routage source avec prise en compte de la QdS par réservation de bande passante et/ou assurance sur un délai donné. Dans un premier temps, nous ne nous intéresserons qu'à la réservation de la bande passante dans la description du protocole. Nous reviendrons ultérieurement sur le critère de délai.

Nous avons choisi ce protocole de routage car c'est le protocole qui se prête le mieux à l'incorporation de l'amélioration que nous proposons. Par ailleurs, ce choix a été renforcé par le fait que ce protocole a été largement étudié dans la littérature. Ce protocole est subdivisé en deux parties indépendantes :

- découverte et réservation de route ;
- maintenance des routes.

3.1.1 Découverte et réservation de routes

À la réception d'une demande de connexion pour une destination donnée avec une exigence de qualité de service exprimée en terme de bande passante, la source initie le processus de routage. Elle envoie à tous ses voisins une demande de connexion vers la destination. Les nœuds qui reçoivent le message pour la première fois et qui répondent aux exigences de bande passante propagent la demande à leur voisinage. Ce message de demande de connexion se propage ainsi dans le réseau jusqu'à la destination. Lorsque la destination reçoit le message de demande de connexion de la source, elle envoie un message de réservation et de confirmation le long de la route trouvée. Dès que la source reçoit ce message de réservation, la connexion est établie et la communication peut commencer.

3.1.2 Maintenance des routes

Du fait de la mobilité de tous les nœuds responsables de la transmission des données entre une source et une destination, les risques que la route se coupe avant la fin de la communication sont très importants. En cas de problème de rupture de lien ou de défaillance de nœud en cours de communication, il existe alors deux scénarii pour le re-routage :

- un re-routage complet à partir de la source de la communication. Ce re-routage est mis en œuvre dans la plupart des protocoles de routage, bien qu'il prenne un temps important et consomme beaucoup de bande passante ;
- un re-routage partiel à partir du nœud où la défaillance a eu lieu. Ce re-routage partiel présente l'intérêt d'être rapide et de consommer peu de bande passante.

Détection des défaillances

Lorsqu'une source souhaite envoyer un paquet de données à une destination et qu'elle dispose de la route pour y accéder avec les bonnes contraintes de QdS, elle commence à envoyer les paquets par le biais de son interface réseau au prochain nœud identifié dans la table de routage récupérée de la phase de détection de route. À la

réception d'un paquet, chaque nœud intermédiaire de la route envoie au nœud précédent un accusé de bonne réception puis le retransmet au nœud suivant de la route. Dans le cas où le lien vers le nœud suivant est rompu, le nœud initie la phase de réparation de route. Cette phase s'appuie sur un re-routage global ou partiel.

Une rupture de lien est détectée par le nœud en amont du lien rompu. Des informations sur la rupture du lien peuvent être fournies par des protocoles de couche inférieure. Par exemple, le protocole IEEE 802.11 de la couche liaison peut prendre en charge cette opération.

Re-routage global

Si, lors de l'acheminement d'un paquet de la source vers la destination, un problème de rupture de lien apparaît, le nœud qui le détecte propage un message d'erreur jusqu'à la source. La source initie alors une nouvelle phase de découverte de route vers la destination. Cette phase de re-routage présente l'inconvénient d'être longue. Cet inconvénient est d'autant plus gênant pour une connexion avec des exigences de QdS exprimées sous forme de délai maximum. Le risque est alors de dépasser le délai maximum et donc de ne plus vérifier le critère défini au moment de l'établissement de la connexion. Un autre inconvénient du re-routage global est qu'il surcharge le réseau avec beaucoup de messages de routage. Il en découle un gaspillage de bande passante préjudiciable à la performance du réseau.

Re-routage partiel

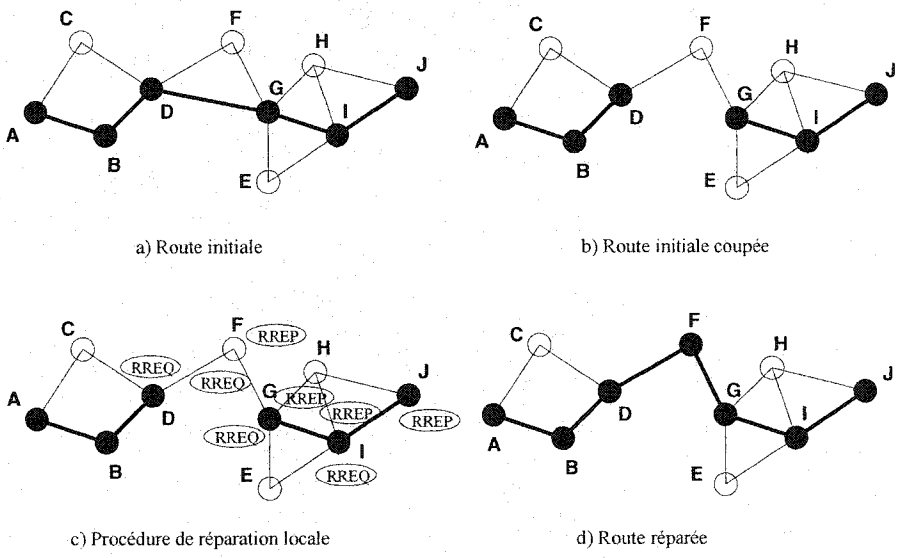
Le re-routage partiel constitue une alternative au re-routage global. Lorsqu'un nœud détecte une anomalie dans un lien, il n'envoie plus systématiquement un message d'erreur au nœud source. Il garde en mémoire le paquet qu'il n'a pas pu envoyer ainsi que les paquets suivants que la source de la communication continue d'envoyer; en parallèle, il se charge lui-même de la réparation de la route puis de la retransmission des paquets vers les nouveaux nœuds intégrés à la route.

Le principe est de trouver un chemin jusqu'à la destination. On s'inspire pour cela de la procédure de re-routage global. Le nœud émet donc des paquets de recherche de route. La différence réside dans le fait que, dans ce cas, le re-routage s'exécute plus près de la destination. On peut donc limiter l'aire de prospection et donc le nombre de ré-émissions des paquets de recherche de route. Par ailleurs cette étape est plus rapide que dans le cas d'un re-routage global car le nœud est plus proche de la destination.

En cas de rupture de route, un paquet de re-routage spécifique (Route Request : RReq) est donc diffusé. La diffusion de ce paquet est limitée par un compteur décrémenté à chaque arrivée du paquet de re-routage dans un nouveau nœud. Quand le compteur passe à zéro, le paquet n'est plus rediffusé. Le chemin trouvé doit répondre aux exigences de QdS. Une fois que le nœud destination reçoit le paquet RReq et qu'il accepte de rétablir la connexion, il retourne un paquet de validation (Route Reply : RRep).

Considérons le cas de la Figure 3.1. Le cas a) représente la route telle qu'elle est avant la détection de la défaillance. Au cas b), le nœud D détecte une défaillance du lien entre D et G. Selon la procédure de re-routage locale du cas c), D diffuse un paquet RReq. À la réception de ce paquet, J retourne à D un paquet RRep. Ce paquet transite par I, G et F. Lorsque D le reçoit, la route est réparée et la communication peut reprendre.

Pour diminuer les risques de défaillance en cours de communication, on peut choisir d'utiliser la route la plus fiable possible, c'est à dire celle pour laquelle les risques de rupture sont minima. Une autre solution consiste pourtant à sélectionner une route facilement réparable, c'est à dire une route dont le risque de rupture n'est pas forcément minimisé mais dont la réparation est facile. Une route est facilement réparable lorsqu'elle peut être réparée par un re-routage local. La mise en œuvre de réparations locales aux probabilités de succès rehaussées est assurée par la prise en compte d'un nouveau paramètre : la *densité* des nœuds le long d'une route donnée.



a) Route initiale

b) Route initiale coupée

c) Procédure de réparation locale

d) Route réparée

Figure 3.1 Restauration de route par re-routage partiel

3.2 Protocole « amélioré » de découverte de routes

L'objectif des modifications apportées est d'assurer la sélection de la route la plus facilement réparable parmi celles dégagées lors de la phase de découverte de routes. Pour atteindre cet objectif, nous préconisons de prendre en compte la nature du voisinage des nœuds du réseau, et en particulier la *densité* de nœuds, ainsi que leur *disponibilité*. La réparation de la route en cas de défaillance d'un nœud s'effectue par la mise en œuvre d'une procédure locale de re-routage, et donc en évitant des re-routages complets coûteux en bande passante et en temps d'exécution, améliorant ainsi les délais de communication.

3.2.1 La *disponibilité*

Le problème est de déterminer dans quelle mesure un nœud faisant partie d'une route entre une source et une destination données et immédiatement voisin d'un autre nœud (c'est à dire dans sa portée radio) peut être remplacé par ce dernier dans le cas où il subirait une défaillance. Il faut en particulier que le nœud de remplacement dispose de canaux de communication de bande passante suffisante pour assurer cette nouvelle

connexion. On utilise alors le paramètre *disponibilité* pour établir si tel nœud est capable de remplacer tel autre nœud. La *disponibilité* d'un nœud dépend de sa nature (ordinateur portable, PDA, etc.), du nombre de paquets qui transitent par ses différents canaux de communications ainsi que de leur capacité. Pour résumer, on peut dire que la *disponibilité* correspond à la bande passante libre au niveau d'un nœud sur ses canaux de communication. Pour que chaque nœud dispose des informations sur la *disponibilité* de ses nœuds voisins, il faut prévoir l'échange régulier entre voisins des données concernant leur disponibilité.

Chaque nœud transmet donc par *broadcast* des informations sur sa disponibilité tous les INTERVALLE_HELLO. Les nœuds qui reçoivent ces messages mettent à jour leur base de données locale. Le Tableau 3.1 illustre la forme des données de disponibilité au niveau d'un nœud avec 5 voisins.

Tableau 3.1 Exemple de la forme des données de disponibilité dans un nœud

	Voisin 1	Voisin 2	Voisin 3	Voisin 4	Voisin 5
Disponibilité (en Koctets/s)	16	132	0	15	45

3.2.2 La *densité*

Nous définissons la *densité* au nœud λ par le nombre de voisins directs de λ (c'est à dire le nombre de nœuds dans la zone de portée radio de λ) dont la bande passante disponible est supérieure à celle exigée par la connexion. Le paramètre *densité* est propre à un nœud et à une connexion donnée (une bande passante donnée). Ainsi, à la réception d'une requête de connexion, chaque nœud évalue sa densité en décomptant le nombre de voisins dont la disponibilité est supérieure à celle exigée par la connexion.

Plaçons nous dans le cas où un nœud reçoit une requête de connexion avec une exigence stipulée en terme de bande passant. Supposons que σ nœuds de son voisinage aient la disponibilité requise par la connexion. La densité est alors fixée à σ pour ce

nœud et cette connexion. Étudions les différents cas envisageables selon la valeur prise par la densité.

Si la densité est égale à un, aucun voisin autre que l'émetteur de la requête ne répond aux exigences de QdS de la connexion. Continuer à diffuser le message de requête de connexion est inutile puisqu'aucun nœud du voisinage n'est de toute façon en mesure de prolonger la route vers la destination. On aboutit alors à une détection anticipée de l'échec imminent de la découverte de route passant par ce nœud, tout en limitant l'engorgement du réseau.

Si la densité est égale à deux, un seul voisin autre que l'émetteur de la requête répond aux exigences de QdS de la connexion. On peut ainsi continuer à diffuser le message de requête de connexion. Cependant, aucun nœud ne pourra prendre le relais du nœud courant défaillant dans le cas où la route serait utilisée pour la communication. Cette densité est donc trop faible pour assurer un re-routage local au niveau du nœud courant. On parle alors d'étranglement.

Pour une densité supérieure à deux, on peut définir le paramètre *redondance* à partir de la *densité* :

$$redondance = densité - 2.$$

Ce paramètre rend compte plus intuitivement de la facilité de réparation de la route en cas de défaillance d'un nœud. Ainsi, pour une densité six, on obtient une redondance égale à quatre. Autrement dit, quatre nœuds seront éventuellement en mesure de remplacer le nœud défaillant au terme d'une phase de re-routage local, limitant ainsi le délai de re-routage. Afin de mieux expliciter ce concept, nous exposons l'exemple qui suit.

Prenons un nœud avec cinq voisins. La mise à jour des données de disponibilité se fait selon les quatre étapes suivantes :

- Étape 1 : Chaque nœud évalue sa disponibilité. Un aperçu est donné au Tableau 3.2.

- Étape 2 : La disponibilité est diffusée par chaque nœud à son voisinage à intervalle de temps régulier.
- Étape 3 : Chaque nœud réceptionne les données de disponibilité envoyées par ses voisins.
- Étape 4 : Le nœud central met à jour les données de disponibilité de ses voisins dans sa base de données locale. Ainsi, la base de données locale de disponibilité au nœud central à l'instant *N*-1, représentée par le Tableau 3.3 est rafraîchie pour prendre en compte les modifications survenues dans le réseau. On obtient alors une nouvelle base de données à l'instant *N* représentée par le Tableau 3.4.

Tableau 3.2 Les données de disponibilité dans chaque nœud

Voisin	Disponibilité (en Koctets/s)
1	16
2	132
3	0
4	15
5	45

Tableau 3.3 Disponibilité des nœuds voisins du nœud central à l'instant *N*-1

Instant *N*-1	Voisin 1	Voisin 2	Voisin 3	Voisin 4	Voisin 5
Disponibilité (en Koctets/s)	16	43	0	54	45

Tableau 3.4 Disponibilité des nœuds voisins du nœud central à l'instant *N*

Instant *N*	Voisin 1	Voisin 2	Voisin 3	Voisin 4	Voisin 5
Disponibilité (en Koctets/s)	16	132	0	15	45

Le processus de mise à jour des données de disponibilité des nœuds voisins peut être résumé par l'enchaînement illustré à la Figure 3.2.

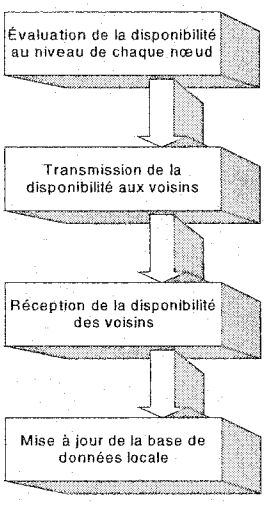


Figure 3.2 Diagramme de mise à jour de la disponibilité des nœuds voisins

3.2.3 Déroulement de la phase de découverte de routes

Nous reprenons la phase de découverte de route décrite à la section 3.1 à laquelle nous ajoutons la prise en compte des paramètres de densité et de disponibilité. Ainsi, dans notre protocole amélioré, la source initie le processus de routage à la réception d'une demande de connexion. Elle envoie à tous ses voisins une demande de connexion vers la destination. Les nœuds qui reçoivent le message pour la première fois et qui répondent aux exigences de QdS propagent la demande à leur voisinage après avoir :

- **incrémenté** le nombre ***n*** de nœuds de la route d'une unité dans le message de requête de connexion ;
- **dénombré** la ***densité*** des nœuds voisins aptes, en terme de bande passante disponible, à prendre le relais de la connexion en cas de défaillance du nœud central ;
- **mis à jour** les champs *densité moyenne* et *nombre d'étranglements* de la route en construction dans le paquet RReq en propagation vers la destination. La mise à jour de la densité moyenne se réalise à l'aide de la formule (3.1).

$$D_m = \frac{D_m * (n-1) + densité}{n} \qquad (3.1)$$

La requête se propage de proche en proche vers la destination en suivant le même schéma de diffusion. Finalement, le message de demande de connexion émis au niveau de la source parvient à la destination. Ce message comporte les données suivantes :

- nombre de nœuds de la route ;
- densité moyenne le long de la route en construction ;
- nombre d'étranglements le long de la route en construction.

La destination déclenche alors un compte à rebours et réceptionne d'autres messages de demande de connexion correspondant à autant de routes détectées entre le point de rupture et la destination. À échéance du compte à rebours et grâce aux données contenues dans les messages de demande de connexion, la destination sélectionne la route dont la réparation est la plus facile, puis propage sur cette route un paquet de confirmation RREP pour réserver les ressources nécessaires.

3.2.4 Déroulement de la phase de sélection de route

Il s'agit de définir une fonction qui maximise les chances de sélectionner la route la plus facilement réparable et donc dont les délais de reconstruction seront les plus faibles puisqu'ils mettent en jeu un re-routage local. Dans cette phase décisionnelle, nous cherchons à exploiter au maximum les paramètres contenus dans les messages de demande de connexions parvenus à la destination. Nous rappelons que ces paramètres sont, pour chaque route, le nombre de nœuds impliqués, la densité moyenne et le nombre d'étranglements. Mais exploiter ces données n'est pas évident.

Dans ce contexte, il faut choisir la route ayant les caractéristiques suivantes :

- le nombre de nœuds le plus faible ;
- la densité moyenne la plus forte possible ;
- le nombre d'étranglements le plus faible.

Une route est d'autant plus difficile à réparer par un re-routage partiel qu'elle fait intervenir de longs enchaînements de nœuds autour desquels la densité est faible. Une

topologie qui contient de tels enchaînements est considérée comme peu intéressante car sa maintenance est compliquée. Le re-routage local en cas de défaillance d'un nœud sur ces routes ne sera en effet pas efficace. Il faudra donc plus souvent procéder à un re-routage global plus gourmand en temps et en bande passante qu'un re-routage local.

Pour mieux comprendre, considérons la configuration de la Figure 3.3. Le Tableau 3.5 indique la densité aux différents nœuds.

Tableau 3.5 Densité le long de la route correspondant à la Figure 3.3

Nœud	1	2	3	4	5	6
Densité	9	3	2	5	8	7

Cette route présente une densité égale à deux au niveau du nœud trois. Supposons que le nœud trois ait une défaillance. Une reconstruction de la route, par re-routage partiel, est alors initiée par le nœud deux qui présente trois voisins dont les nœuds un et trois qui font déjà partie de la route, soit une *redondance* égale à un. Le re-routage partiel est rendu difficile et échouera sans doute. Il faudra alors procéder à un re-routage global initié par la source de la communication. Le problème est qu'un re-routage global est gourmand en temps. Ici, la réparation de la route sera d'autant plus longue que le re-routage global s'effectue après une tentative avortée de re-routage partiel. Dans ce cas, il semble plus judicieux de ne pas essayer de reconstruire la route par un re-routage partiel. Nous préconisons au contraire, en cas de défaillance, de procéder directement à un re-routage global plutôt que d'exécuter d'abord un re-routage partiel puis, comme ce dernier n'a pas réussi, un re-routage global.

Mais une autre solution consiste à éviter d'utiliser de telles routes difficilement réparable en cas de défaillance. En fait, il faudrait sélectionner, dès la construction de la route, une route réparable par re-routage partiel. Nous reviendrons plus loin sur la manière de sélectionner une route répondant à ces critères.

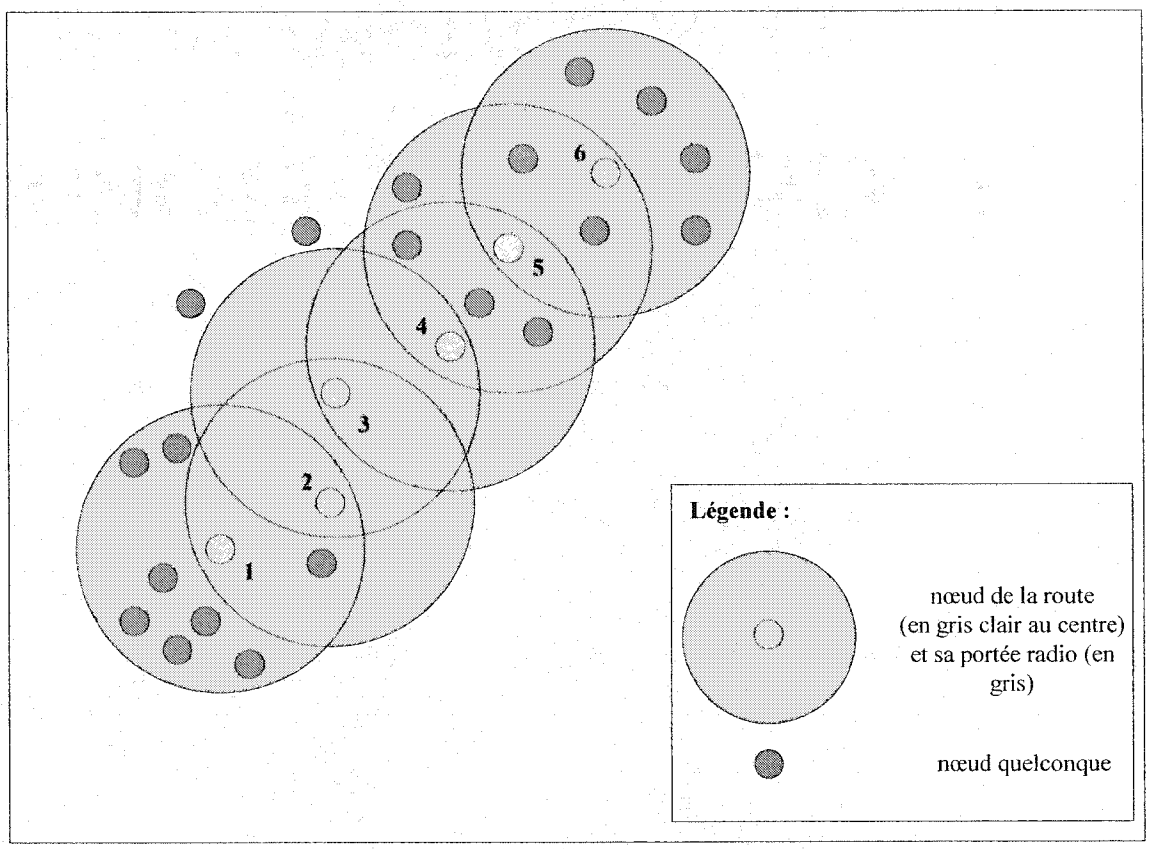


Figure 3.3 Exemple de route difficilement réparable

Réciproquement, une route est d'autant plus facile à réparer par un re-routage partiel qu'elle fait intervenir de longs enchaînements de nœuds autour desquels la densité est forte. Une telle topologie est très intéressante car sa maintenance est facile en cas de défaillance d'un nœud. Il faudra donc plus rarement procéder à un re-routage global à la suite d'un re-routage partiel qui a échoué.

Pour mieux comprendre, considérons la configuration de la Figure 3.4. Le Tableau 3.5 précise la densité des différents nœuds.

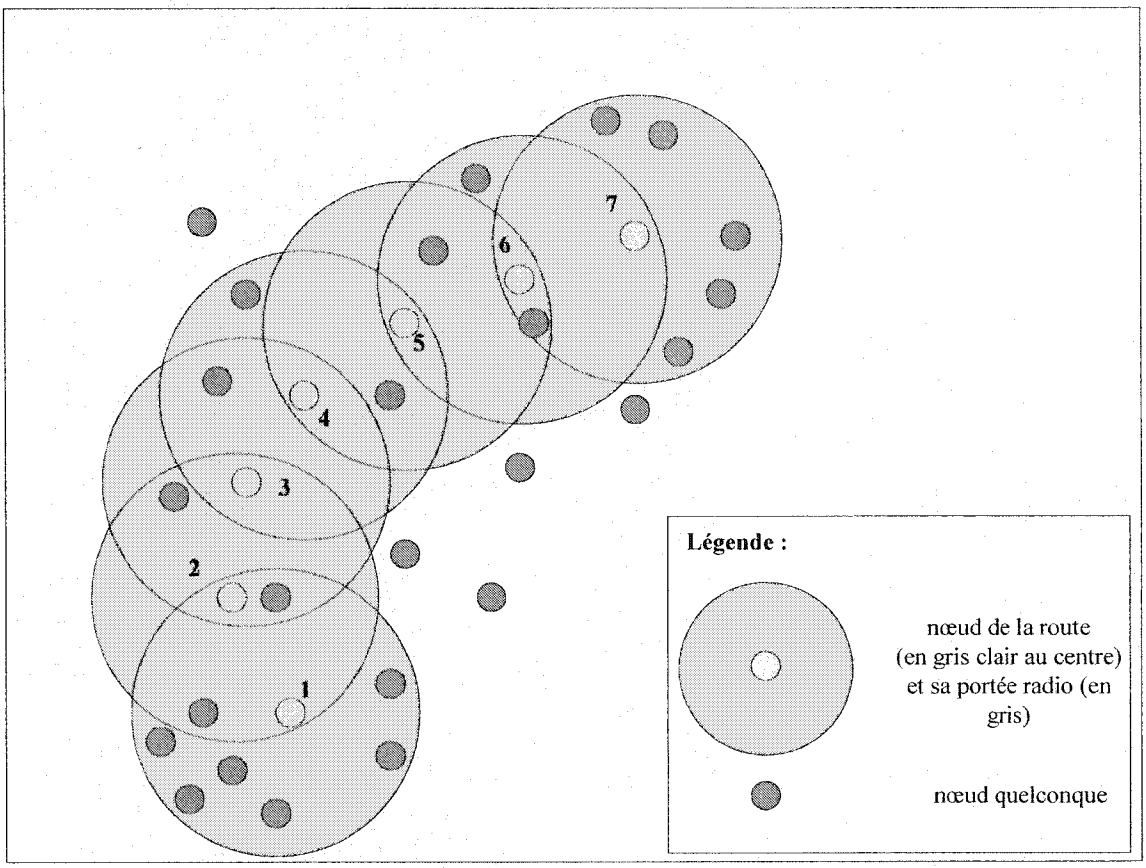


Figure 3.4 Exemple de route facilement réparable

Tableau 3.6 Densité le long de la route correspondant à la Figure 3.4

Nœud	1	2	3	4	5	6	7
Densité	9	5	5	5	5	5	7

Cet exemple illustre bien l'intérêt d'avoir non seulement des nœuds à forte densité mais surtout que chaque nœud ait une densité minimale assurant de grandes chances qu'un re-routage partiel réussisse en cas de défaillance d'un nœud le long de la route. En effet, quel que soit le nœud ayant une défaillance, le nœud qui le précède est en mesure de mener un re-routage local dont le succès est assuré par les fortes densités (la densité est ici égale à cinq).

Notation de route

Notre schéma de notation de route, prélude à la sélection de la route la plus facilement réparable, doit s'appuyer sur les différents critères évoqués plus haut. La note tient compte du nombre de nœuds n de la route : plus il est élevé, plus la note sera faible. Elle tient aussi compte de la densité moyenne D_m le long de la route exprimée par (3.2).

$$D_m = \frac{\sum densité}{n} \qquad (3.2)$$

Pour finir, la note de la route tient compte du nombre d'*étranglements* qu'elle comporte. Rappelons qu'un *étranglement* correspond à un nœud de la route où la densité est égale à deux (c'est à dire un nœud pour lequel la redondance est nulle). Plus il y a d'étranglements, plus la route est difficile à réparer par un re-routage partiel et donc plus la note attribuée est faible. Soit e le nombre d'étranglements. Nous pouvons donc calculer la note d'une route comme suit :

$$Note = \frac{D_m}{n + 1 + e(5/2 - 1/2n)} \qquad (3.3)$$

On retiendra que la route la plus facilement réparable est celle dont la notation est la plus élevée.

Choix et réservation de la route

Une fois qu'une note a été affectée à chaque route, le nœud destination détermine celle dont la note est la plus grande. Ce sera cette route qui sera utilisée entre la source et la destination pour la connexion. Si deux routes ont la même note, la route sélectionnée est celle dont le message de requête de connexion est arrivé le plus tard à la destination. En effet, les données contenues dans une telle requête reflètent le mieux l'état actuel du réseau.

Dès que la route a été choisie, la destination envoie un message de réservation et de confirmation qui se propage le long de la route. La connexion est établie à la réception de ce message par la source. La communication peut alors commencer.

3.2.5 Analyse mathématique

Nous proposons une justification théorique de l'amélioration du protocole décrite plus haut. Nous évaluons le temps de re-routage moyen pour une route quelconque et pour une défaillance ponctuelle quelconque. Soit *n* le nombre de nœuds de cette route et *e* le nombre d'étranglements le long de cette route.

Durée d'un re-routage local

Soit τ le temps de propagation d'un message de notification d'un nœud à un autre. Désignons par $T_{rl,i}$ la durée du re-routage local au nœud *i,* où *i* désigne la distance entre la source de la communication et le nœud qui initie le re-routage. À l'aide la Figure 3.5, nous déterminons facilement :

$$T_{rl,i} = 2\tau(n-i) \qquad (3.4)$$

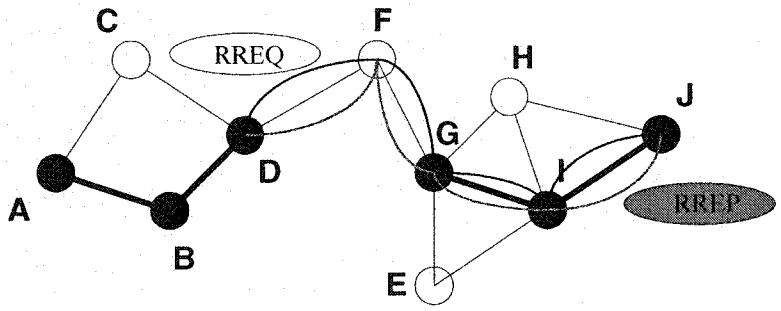


Figure 3.5 Re-routage local

Durée d'un re-routage global

Désignons par $T_{rg,i}$ la durée du re-routage global au nœud *i.* À l'aide la Figure 3.6, nous déterminons :

$$T_{rg,i} = \tau[i+2n] \qquad (3.5)$$

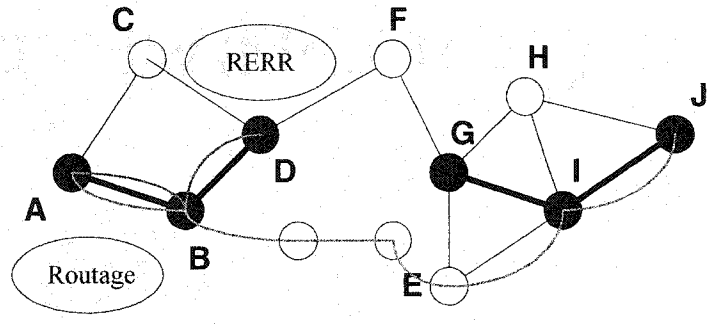


Figure 3.6 Re-routage global

Durée d'un re-routage pour un nœud i donné

Soit T_i la durée d'un re-routage au nœud *i* et soit p_i la probabilité de succès d'un re-routage local au niveau du nœud *i*. Nous avons :

$$p_i = 1 - \frac{e}{n} \qquad (3.6)$$

Ainsi, la probabilité de succès d'un re-routage local décroît avec le nombre d'étranglements de la route mais est indépendante de la position *i* à partir de laquelle le re-routage local est mis en œuvre.

Comme nous procédons d'abord systématiquement à un re-routage local puis, si ce dernier échoue, nous enchaînons avec un re-routage global, nous en déduisons :

$$T_i = p_i * T_{rl,i} + (1 - p_i) * \left[T_{rl,i} + T_{rg,i}\right] \qquad (3.7)$$

Durée moyenne d'un re-routage

Nous considérons que la probabilité de défaillance d'un nœud est indépendante de sa position sur la route. Soit *T* la durée moyenne d'un re-routage. Nous avons :

$$T = \frac{\sum_{i=0}^{n-1} T_i}{n} \qquad (3.8)$$

donc

$$T = \frac{\tau}{n}\sum_{i=0}^{n-1}\left[2(n-i)+\frac{e}{n}(i+2n)\right] \qquad (3.9)$$

soit après simplification :

$$T = \tau\left[(n+1)+e(5/2-1/2n)\right] \qquad (3.10)$$

Comme nous pouvions nous y attendre, nous remarquons que T croit avec le nombre e d'étranglements et la longueur n de la route. Cette observation est donc conforme à la définition de la note donnée par l'équation (3.3). En effet, la notation est inversement proportionnelle à ces paramètres.

3.3 Protocole « amélioré » de maintenance de route

Si un nœud détecte que le lien avec son successeur pour une route donnée est rompu, il peut s'avérer inutile, voire coûteux en terme de temps d'exécution du re-routage, de procéder à un re-routage partiel. Supposons par exemple que le nœud en question dispose d'une densité égale à deux, soit une redondance nulle (c'est à dire que ses seuls voisins sont son prédécesseur et son successeur sur la route). Lorsqu'il entreprend un re-routage partiel, la probabilité que ce dernier réussisse est presque nulle. En effet, nous savons qu'aucun nœud voisin n'est en mesure de relayer la communication vers la destination. Dans ce cas, nous décidons de confier directement la tâche de re-routage à la source, ce qui revient à procéder à un re-routage complet avec les inconvénients que cela implique mais évite la perte de temps supplémentaire occasionnée par l'échec de la tentative de re-routage partiel.

Une autre alternative consiste à choisir un nœud de la route dont la densité est suffisamment importante pour déboucher sur un succès de re-routage partiel et donc éviter le re-routage global. Il existe d'autres cas où le choix d'un point d'ancrage judicieux peut s'avérer intéressant, en particulier si l'unité mobile qui devrait procéder au re-routage a une trop faible capacité de stockage ou une trop faible capacité de calcul comme par exemple un assistant numérique personnel ou PDA.

Re-routage partiel avec sélection optimale du point d'ancrage

Ce re-routage n'est mis en œuvre que dans le cas où la densité de nœuds autour de l'unité mobile qui détecte la rupture de lien est égale à deux, ou si elle n'a pas les capacités pour assurer une phase de re-routage. Après avoir détecté la rupture du lien, l'unité mobile envoie un message de notification de rupture du lien en question aux nœuds en amont. Un nœud de la route qui reçoit ce message vérifie sa densité d'unités mobiles pour la route en question. Si cette densité est supérieure à quatre et qu'il en a la capacité, le nœud prend en charge le re-routage. Sinon, il se contente de le transmettre au nœud précédent jusqu'à atteindre la source à partir de laquelle il faudra alors exécuter un re-routage global.

Une solution séduisante serait d'exécuter un re-routage partiel à chaque nœud en aval du lien rompu. Procéder de la sorte permettrait d'exécuter une réparation rapide de la route. Cependant, nous ne retenons pas cette solution car elle induit un flot de requêtes différentes concernant la même connexion qui engorgera le réseau.

3.4 Prise en compte des exigences de délai

L'objectif est toujours de sélectionner la route qui sera la plus apte à être réparée facilement, mais cette fois avec le délai comme critère de QdS. Cela signifie que lorsqu'une route est défaillante, la nouvelle route issue du re-routage doit continuer à vérifier les exigences de délai global de la connexion.

Le délai d'un nœud est le temps mis par un paquet pour traverser ce nœud. Ce délai dépend de l'engorgement du nœud et de sa nature (ordinateur portable, assistant personnel, etc.). La donnée du délai au niveau d'un nœud correspond à un délai garanti pour les paquets qui le traversent. En pratique, le délai effectif pour un nœud donné doit donc toujours être inférieur au délai indiqué pour ce nœud.

Puisque le délai dépend fortement du temps, il est important de le réévaluer régulièrement. Chaque nœud doit disposer des informations sur le délai de ses nœuds voisins. Il faut donc prévoir que chaque nœud transmette régulièrement les données concernant son délai. Ainsi, chaque nœud transmet par diffusion locale des informations

sur son délai tous les INTERVALLE_HELLO à ses voisins. Les nœuds qui reçoivent ces messages mettent à jour leur base de données locale. Un exemple de la forme des données de délai au niveau d'un nœud avec cinq voisins est donné au Tableau 3.7.

Tableau 3.7 Données de délai des nœuds voisins du nœud central

	Voisin 1	Voisin 2	Voisin 3	Voisin 4	Voisin 5
Délai (en ms)	218	49	65	142	92

La *densité* au nœud *n* réfère au nombre de voisins directs de *n* (c'est à dire du nombre de voisins dans la zone de portée radio de *n*). Cependant, le paramètre *densité* ne tient compte que des nœuds du voisinage dont le délai intrinsèque est inférieur ou égal au délai du nœud courant.

Le paramètre densité est propre à un nœud mais ne dépend plus de la connexion, contrairement à la densité relative aux exigences en terme de bande passante. Pour l'évaluer, on décompte le nombre de nœuds voisins dont le délai est inférieur à celui du nœud central.

Interprétation de la densité

On se place dans le cas où un nœud *n* reçoit une requête de connexion avec une exigence sur un délai donné. Supposons que quatre nœuds de son voisinage aient un délai inférieur au délai du nœud *n*. Deux de ces quatre nœuds correspondent respectivement au successeur et au prédécesseur du nœud le long de la future route, on ne peut donc pas les considérer comme remplaçants envisageables en cas de défaillance du nœud. Par contre, les deux autres nœuds seront aptes, éventuellement, à assurer ce rôle (si on suppose qu'entre temps, la configuration du réseau n'a pas trop changé).

Pour résumer, si la densité est égale à un, aucun voisin autre que l'émetteur de la requête, ne répond aux exigences de QdS de la connexion. Continuer à diffuser le message de requête de connexion est en pure perte puisqu'aucun nœud du voisinage

n'est en mesure de prolonger la route vers la destination. On aboutit à un échec de la requête de connexion pour ce nœud.

Si la densité est égale à deux, un seul voisin autre que l'émetteur de la requête répond aux exigences de QdS de la connexion. On continue à diffuser le message de requête de connexion. Cependant, aucun nœud ne pourra prendre le relais du nœud courant défaillant dans le cas où la route serait sélectionnée. Cette densité est donc trop faible pour assurer un re-routage partiel au niveau du nœud courant.

Pour une densité supérieure à deux, on peut définir le paramètre redondance de la même manière que précédemment :

$$Redondance = Densité - 2.$$

Ce paramètre rend compte du nombre de nœuds voisins de la route et normalement aptes, en cas de défaillance d'un nœud, à assurer le re-routage partiel. Ainsi, pour une densité égale à six, on peut estimer que quatre nœuds seront peut-être en mesure de remplacer le nœud défaillant au terme de la phase de re-routage partiel.

Exèmple

Prenons un nœud avec cinq voisins. La mise à jour des données de disponibilité se fait en cinq étapes.

- Étape 1 : Chaque nœud évalue son délai.
- Étape 2 : Le délai est diffusé par chaque nœud à son voisinage à intervalle de temps régulier.
- Étape 3 : Chaque nœud réceptionne les données envoyées par les voisins.
- Étape 4 : Le nœud central met à jour les données de délai de ses voisins dans sa base de données locale. Ainsi, la base de données locale de délai au nœud central à l'instant N-1 qui est représentée par le Tableau 3.7 est mise à jour pour prendre en compte les modifications. On obtient alors une nouvelle base de données à l'instant N représentée par le Tableau 3.8.
- Étape 5 : Le nœud central évalue sa densité en dénombrant le nombre de ses voisins direct dont le délai est inférieur au sien.

Tableau 3.8 Données de délai dans chaque nœud

No. du voisin	Délai (en ms)
1	229
2	72
3	65
4	185
5	87

Tableau 3.9 Délai des nœuds voisins du nœud central à l'instant *N*-1

Instant *N*-1	Voisin 1	Voisin 2	Voisin 3	Voisin 4	Voisin 5
Délai (ms)	218	49	65	142	92

Tableau 3.10 Délai des nœuds voisins du nœud central à l'instant *N*

Instant *N*	Voisin 1	Voisin 2	Voisin 3	Voisin 4	Voisin 5
Délai (ms)	229	72	65	185	87

La Figure 3.7 résume le processus de mise à jour des données de délai des nœuds voisins.

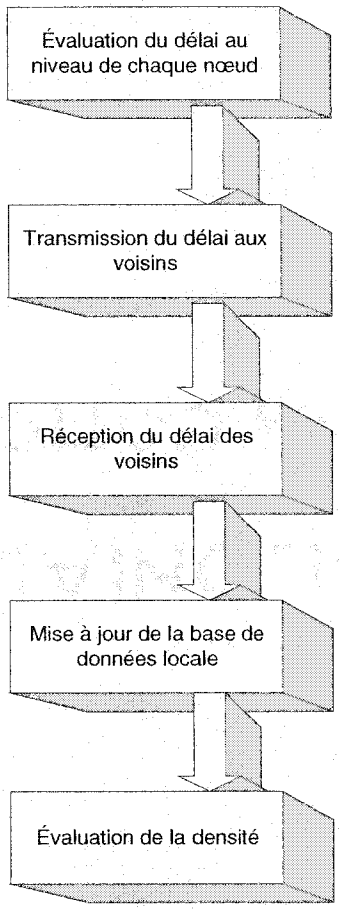


Figure 3.7 Procédure de mise à jour du délai des nœuds voisins

Découverte de routes

À la réception d'une demande de connexion en provenance d'une couche supérieure, la source initie le processus de routage. Elle envoie à tous ses voisins une demande de connexion vers la destination avec une exigence sur un délai donné. Cette demande de connexion comporte en particulier une variable contenant la valeur du délai maximal autorisé pour la connexion ainsi qu'une autre variable correspondant au délai de la source jusqu'au nœud courant. Chaque nœud propage la requête au voisinage après avoir :

- **incrémenté** le nombre de nœuds de la route d'une unité dans le message de requête de connexion ;

- **ajouté** la densité du nœud courant au tableau des densités contenu dans le message de requête de connexion. Il n'est pas nécessaire de réévaluer la densité au nœud courant car cette densité est déjà évaluée à chaque rafraîchissement des données en provenance du voisinage;
- **déterminé** le délai global de la route en construction. Si le délai global, stocké dans le message de demande de connexion, dépasse le délai maximum autorisé par la connexion, le message est effacé car la route en cours de construction, même si elle parvient à la destination, ne vérifiera pas les critères de qualité de service.

La requête se propage de proche en proche vers la destination en suivant le mécanisme décrit précédemment. Lorsque la destination reçoit le message de demande de connexion de la source, ce dernier comporte les données suivantes :

- nombre de nœuds de la route ;
- tableau de densité tout au long de la route ;
- délai de la connexion de la source à la destination.

La destination reçoit éventuellement d'autres messages de demande de connexion correspondant à la même connexion. Elle sélectionne alors la route la plus facilement réparable qui vérifie les contraintes de délai.

Une fois que la route a été choisie, la destination envoie un message de réservation et de confirmation le long de cette dernière. Quand ce message est parvenu à la source, la connexion est établie et la communication peut commencer. C'est la fin de l'étape de routage.

Mécanisme de sélection de route

Il s'agit encore de définir une heuristique qui maximise les chances de sélectionner la route la plus facilement réparable et donc dont les délais de reconstruction seront les plus faibles.

Les critères de choix de la route ont été énoncés précédemment : on cherche la route qui minimise le nombre de nœuds et d'étranglements, et qui maximise la densité.

La relation (3.3) peut être utilisée pour noter les routes. Une fois qu'une note a été affectée à chaque route, le nœud destination détermine celle dont la note est la plus grande. Ce sera cette route qui sera utilisée entre la source et la destination pour la connexion. Si deux routes ont la même note, la route sélectionnée est celle dont le message de requête de connexion est arrivé le plus tard à la destination. En effet, les données contenues dans la requête arrivée le plus tard à la destination reflètent le mieux l'état actuel du réseau.

Dès que la route a été choisie, la destination envoie un message de réservation de ressources le long du chemin sélectionné en utilisant la liste des nœuds précédant la destination et stockés dans le champ correspondant de la requête parvenue à la destination. Ce message se propage le long de la route jusqu'à la source. À la réception de ce message par la source et après réservation au niveau de la source des ressources adéquates, la communication est initiée entre source et destination.

Les exigences de QdS peuvent aussi porter à la fois sur le délai et la bande passante. Dans ce cas, les routes recherchées devront vérifier ces deux contraintes. On fusionne alors les deux processus décrits plus haut. Ainsi, chaque nœud transmet régulièrement à ses voisins son délai et la bande passante disponible sur ses différents liens. La densité devient le nombre de voisins d'un nœud qui respectent à la fois le délai et la bande passante nécessaires pour la connexion. Pour le reste, le routage est le même que pour le routage avec contrainte sur un seul critère.

3.5 Prise en compte cumulée de la fiabilité et de la facilité de réparation

Une route est *fiable* lorsque la probabilité de la voir se briser soit par départ soit par défaillance d'un nœud est faible. Il est clair que dans les réseaux ad-hoc où on ne contrôle pas la mobilité des nœuds, une route complètement fiable demeure difficile à obtenir. Cependant, des heuristiques ont été établies pour assurer une optimisation de la fiabilité des liens retenus au moment de la phase de sélection de route. Ces heuristiques permettent ainsi de dégager, parmi un ensemble de routes possibles, laquelle sera la plus fiable. Elles s'appuient sur les informations disponibles dans le réseau à propos de

l'historique des unités mobiles. Ainsi, on peut exploiter la durée depuis laquelle une unité mobile est immobile pour prédire si elle risque de rester immobile (dans un avenir proche) ou au contraire de se déplacer mettant ainsi en péril une communication.

Une idée intéressante consiste à prendre en compte simultanément la facilité de réparation de la route et sa fiabilité. On peut alors améliorer le choix de la route pour sélectionner la route la plus fiable et en même temps la plus réparable. Cette route cumulerait alors les avantages suivants : elle est suffisamment fiable pour se rompre rarement et, lorsqu'elle se rompt, un re-routage partiel est réalisable du fait de sa facilité de réparation.

CHAPITRE IV

IMPLÉMENTATION ET RÉSULTATS

Après avoir conçu notre algorithme de routage, nous procédons à présent à son implémentation. Comme le cycle de développement complet d'un protocole reprenant toutes les spécifications proposées peut s'avérer fort long et nous éloigner de nos principaux objectifs, nous préférons nous focaliser sur une partie définie du travail qui permet, via une phase de tests, de valider l'ensemble du travail présenté. Dans ce chapitre, nous commencerons par présenter *Opnet Modeler*, l'outil que nous avons choisi pour réaliser les simulations de réseau ad-hoc. Puis, nous décrirons l'implémentation du protocole AODV initial ainsi que les améliorations encodées. Ensuite, nous suivrons le déroulement de notre implémentation dans une configuration donnée. Pour finir, et après avoir défini notre plan d'expérience, nous présenterons les résultats des simulations.

4.1 Présentation de *Opnet Modeler*

De grande notoriété dans le monde professionnel, *Opnet Modeler* est un logiciel de simulation spécialisé dans les réseaux. Commercialisé par la firme *Opnet*, il offre une interface graphique et une modélisation orientée objet qui facilitent le développement de protocoles et la configuration de systèmes complexes en cours de tests. L'un de ses principaux atouts est de permettre la détection de failles dans un système au cours de sa conception et non plus tard, lors de son déploiement. L'organisation du logiciel s'articule autour d'une hiérarchie à quatre niveaux : le niveau projet, le niveau réseau, le niveau modèle de nœud et le niveau processus et machine à états finis.

Le niveau projet

Il constitue le niveau le plus élevé de la hiérarchie. Il permet de gérer la liste des scénarii du projet (par exemple dupliquer un scénario). Nous l'utilisons surtout pour accéder aux différents scénarii.

Le niveau réseau

Il regroupe l'ensemble des unités mobiles et offre une vision d'ensemble du scénario envisagé. C'est à ce niveau que l'utilisateur peut ajouter des nœuds ou ajuster leurs attributs comme les paramètres position, mobilité (fréquence et pas de chaque déplacement), les modes de communication entre nœuds (mode silencieux ou communication avec interlocuteur). La Figure 4.1 représente l'apparence du niveau réseau. Chaque octogone correspond à un nœud mobile du réseau ad-hoc.

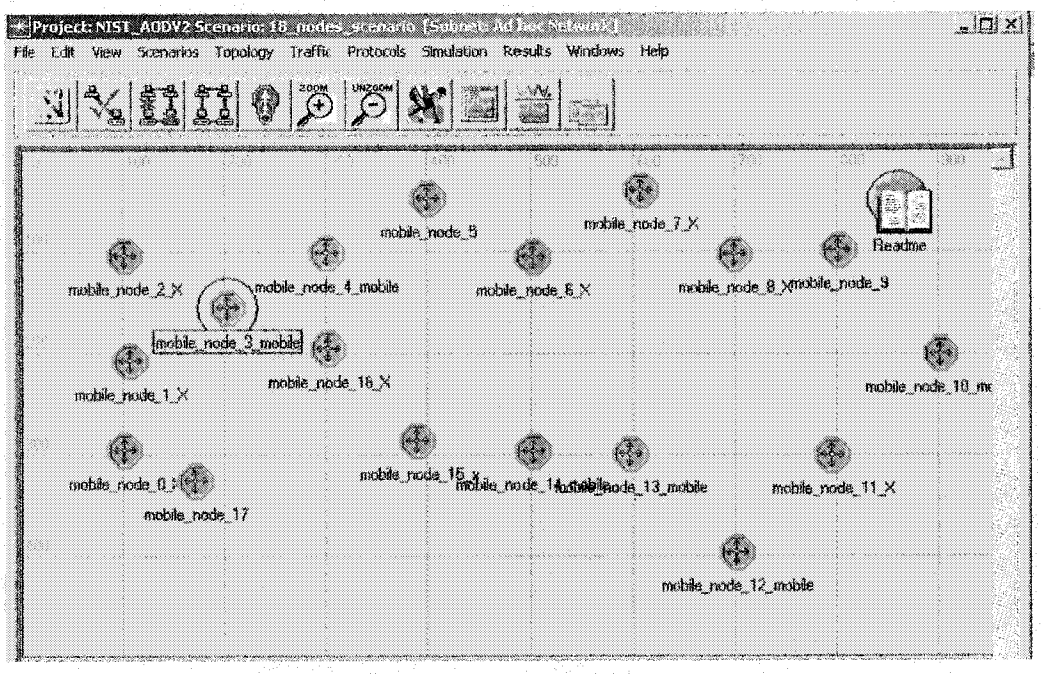


Figure 4.1 Exemple de scénario

Le modèle de nœud

Ce niveau correspond à une vue de détail d'un nœud. Il regroupe l'ensemble des bus et des machines à états finis. Le modèle de nœud utilisé dans nos simulations est donné à la Figure 4.2. Nous constatons qu'il incorpore une couche application, une couche routage, une couche liaison de données, une couche physique ainsi qu'un élément responsable de la gestion de la mobilité du nœud. Notre travail dans ce mémoire est réalisé au niveau de la couche routage.

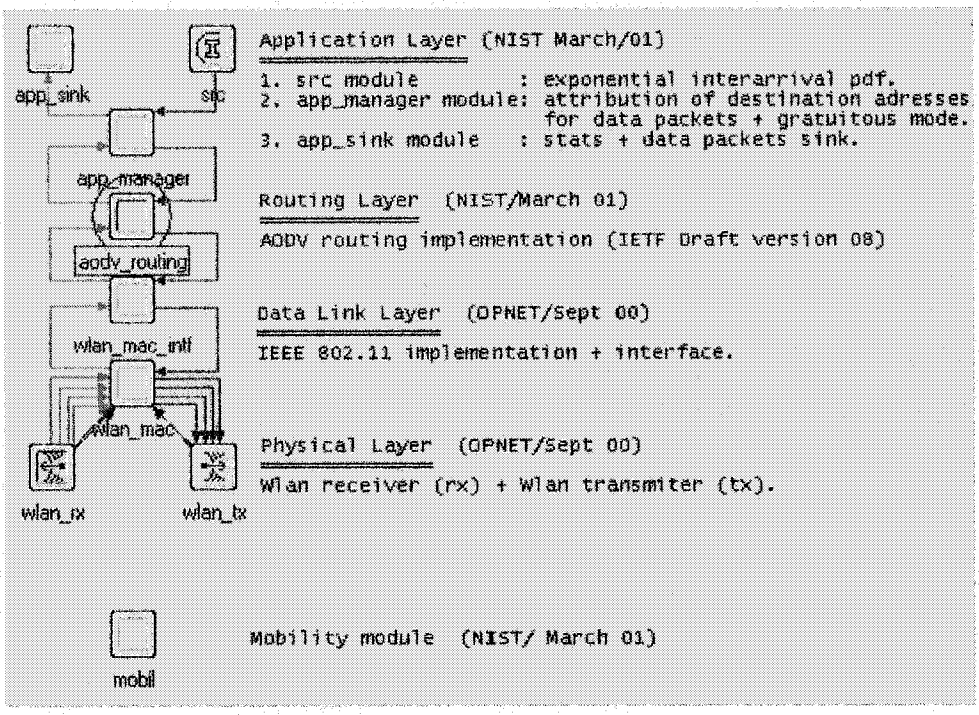


Figure 4.2 Modèle d'un nœud AODV

Le niveau processus et machine à états finis

Il est constitué par l'ensemble des états et des transitions entre ces états. Il existe deux types d'états : les états bloquants et les états non-bloquants. Pour les états non-bloquants, l'entrée dans l'état est suivie automatiquement par la sortie de cet état. En revanche, dans les états bloquants, la sortie de l'état est sujette à un évènement précisément stipulé par l'utilisateur.

Ce niveau est entièrement configurable par l'utilisateur soit graphiquement (ajout d'un nouvel état, d'une nouvelle transition), soit directement en code C pour des modifications plus subtiles. *Opnet Modeler* ne comporte pas de compilateur mais utilise celui de Microsoft Visual Studio.

Collecte des résultats

Toute l'efficacité de la simulation réside dans la capacité de l'outil de simulation à fournir des mesures précises de données pertinentes du point de vue de l'utilisateur. Dans *Opnet Modeler*, les données statistiques sélectionnées sont stockées dans des fichiers particuliers appelés *scalar files*. Opnet modeler permet la génération de courbes présentant l'ensemble des résultats d'un scénario. Typiquement les données collectées sont des délais ou encore le débit.

4.2 Implémentation du protocole AODV

L'implémentation de AODV sous Opnet Modeler se présente de la manière suivante. Il s'agit d'un package développé en mars 2001 par le National Institute of Standards and Technology (NIST) d'après le Draft version 08 de l'Internet Engineering Task Force (IETF) et disponible en téléchargement directement à partir du site Internet www.opnet.com. Dans ce package, le protocole de routage est inclus dans un nœud dont le modèle est représenté à la Figure 4.2. Nous remarquons qu'AODV est situé entre la couche application et la couche état de lien, basée sur la technologie 802.11. Il est important à ce stade de préciser que cette version de AODV supporte les réparations locales de route.

4.2.1 Format des paquets

Quatre types de paquets sont échangés entre les nœuds dans le cadre du protocole de routage AODV : les paquets RERR, REEQ, RREP et DATA.

RERR

Il s'agit des paquets d'erreurs. Ces paquets sont envoyés à la source de la communication pour signaler que la route vers une destination est rompue et que la tentative de re-routage local a été infructueuse. Lorsque la source reçoit ce paquet, elle invalide le statut correspondant à la destination, interrompt l'envoie les paquets vers cette destination et les place dans une file d'attente. Parallèlement, la source déclenche une procédure de re-routage global. Le format d'un RERR est donné à la Figure 4.3.

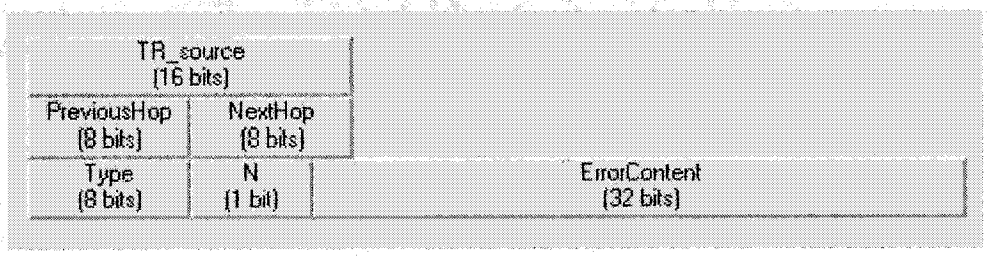


Figure 4.3 Format d'un paquet RERR

RREQ

Ces paquets sont générés lorsqu'une source ne dispose pas d'une route active pour une destination alors qu'elle a des données à lui transmettre. La source les diffuse à ses voisins qui les rediffusent autant de fois qu'indiqué par le champ TTL. Le format d'un RREQ est indiqué à la Figure 4.4

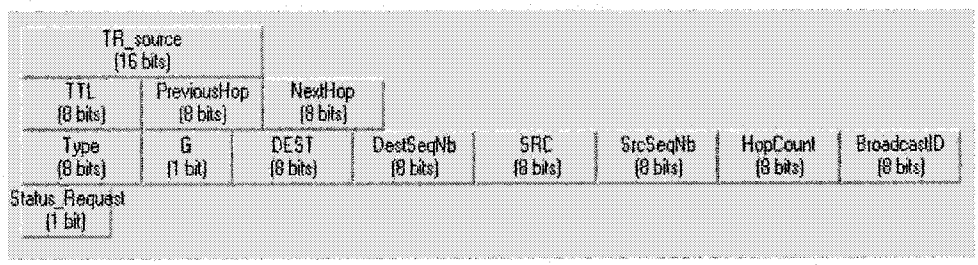


Figure 4.4 Format d'un paquet RREQ

RREP

Ces paquets sont générés par la destination à la réception du RREQ correspondant. Le RREP est propagé le long de la route jusqu'à la source qui a initié le

processus de découverte de route. À sa réception, la source initie la communication. Le format d'un RREP est donné à la Figure 4.5

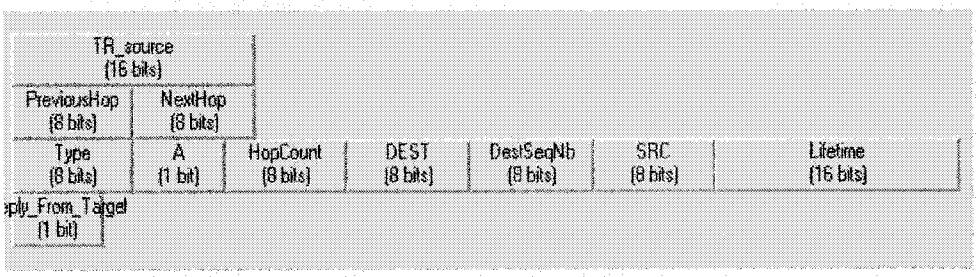


Figure 4.5 Format d'un paquet RREP

DATA

Ces paquets dont l'entête est représentée à la Figure 4.6 transportent l'information que la source souhaite envoyer à la destination. La taille de ces paquets est fixée à 1024 bits. *Opnet Modeler* tient compte de la taille de ces paquets dans les simulations grâce au paramètre *bulk*.

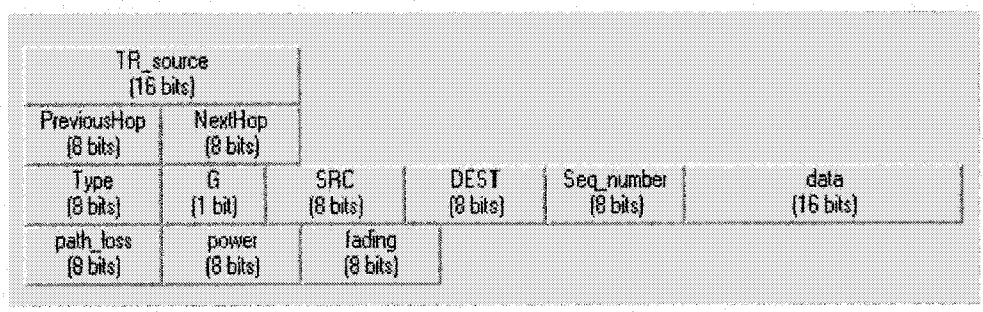


Figure 4.6 Format de l'entête d'un paquet DATA

4.2.2 Les variables d'état

De par sa complexité, le protocole AODV fait intervenir de nombreuses variables d'états. Nous n'évoquons donc que les principales nécessaires à la bonne compréhension du fonctionnement :

- *node_id :* elle correspond à l'identité du nœud. Chaque nœud a une identité unique. Elle sont distribuées lors de l'initialisation de la simulation.

- *node_addr* : cette variable d'état est promue au niveau nœud, c'est à dire que nous devons spécifier manuellement l'attribut adresse pour chaque nœud avant de lancer la simulation.
- *MySeqNb :* cette variable est le numéro de séquence de la source.
- *RequestSeen[N][N]* : *RequestSeen[source][destination]* contient des informations à propos du RREQ issu du nœud "source" et destiné au nœud "destination". Cette donnée aide le nœud qui reçoit un RREQ à savoir s'il doit le propager ou non.
- *RequestSent*[N] : *RequestSent*[i] contient des informations concernant le RREQ qui a été généré pour la destination "i". Ces informations sont le numéro de séquence de la destination, le nombre de tentatives de routage, le temps d'expiration.
- *Routingtable :* il s'agit d'une liste chaînée regroupant toutes les informations de routage disponibles au niveau du nœud pour chaque destination.
- *ReverseListOfPrecursors* : il s'agit d'une autre liste chaînée qui regroupe pour chaque entrée la liste des successeurs d'un nœud donné. Par exemple, si A est dans la liste de précurseurs de B, alors B doit être dans la liste de successeurs de A.

4.2.3 La machine à états finis

Comme nous pouvons le remarquer à la Figure 4.7, la machine à états finis s'articule autour de 9 états parmi lesquels un seul est bloquant. L'état *Init*, pointé par la flèche noire vers la droite, permet d'initialiser les variables d'état de chaque unité mobile et de prévoir les premières interruptions pour déclencher la diffusion des messages HELLO. L'état bloquant central correspond au cas où le système est au repos. Chaque état non bloquant autre que l'état *Init* est atteint par une transition à partir de l'état *Idle*.

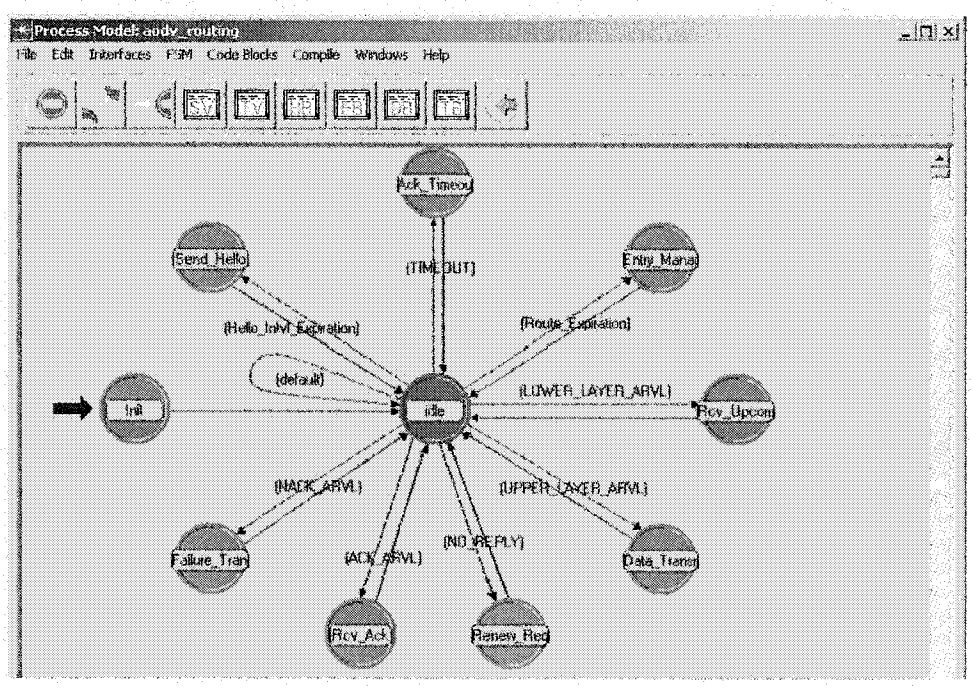


Figure 4.7 Représentation de la machine à états finis initiale

4.2.4 Les principales fonctions et les statistiques

La version initiale de AODV implémente pas moins d'une soixantaine de fonctions chargées de la génération des différents paquets ainsi que du déroulement de leur réception. D'autres fonctions sont encore chargées de la gestion des bases de données qui servent au routage des paquets ou encore de l'affichage en mode déboguage.

Dans la version initiale de AODV, peu de statistiques étaient relevées. Nous ajoutons la mesure du délai de propagation pour les paquets de données. Nous mesurons également le nombre de paquets envoyés et reçus, le nombre de paquets détruits au cours de la simulation, ainsi que les temps de recherche de route initiaux et de réparations locales. En fin de simulation, nous collectons ces résultats et nous éditons un fichier texte les regroupant.

La durée d'une opération de recherche de route globale correspond au laps de temps écoulé entre le moment où un paquet de données pour lequel nous ne disposons pas de route arrive de la couche supérieure et est mis en file et l'instant où la phase de

recherche de route est terminée et le paquet est sorti de la file pour être envoyé à la couche MAC.

La durée d'une opération de réparation locale de route correspond au laps de temps écoulé entre le moment où la fonction aodv_link_repair_attempt() génère un RREQ pour la destination rendue inaccessible par le bris de lien et le moment où le premier paquet mis en file est envoyé à la couche inférieure.

4.3 Détail d'implémentation

Nous avons commencé par implémenter un mécanisme périodique d'échange de la disponibilité entre voisins et de calcul, également périodique, de la densité résultante au niveau de chaque nœud. Puis, nous avons apporté différentes améliorations basées sur la prise en compte de ces critères.

4.3.1 Implémentation du calcul périodique de la densité

Les variables d'état

Il s'agit de l'ensemble des variables propres à chaque nœud :

- *disponibilité* : comme notre protocole ne permet pas de réserver la bande passante, la mise à jour de cette variable ne dépend pas de l'état d'engorgement du réseau. Nous décidons de promouvoir cette variable pour la paramétrer individuellement pour chaque nœud.
- *disponibilite_voisins* : il regroupe les dernières disponibilités reçues pour chaque nœud du voisinage depuis la dernière mise à jour de la densité.
- *densité* : elle est mise à jour à intervalle de temps régulier (toutes les 20 secondes). L'intervalle de temps choisi est le fruit d'un compromis entre une bonne information sur l'état du réseau et un engorgement limité.

Le format des paquets RREPs

Comme indiqué à la Figure 4.8, nous ajoutons aux RREPs un champ de 8 bits qui stocke la disponibilité du nœud émetteur lors de la diffusion au voisinage d'un message

hello. Nous faisons en effet le choix d'inclure la communication de la disponibilité dans les paquets HELLO (de type RREP) diffusés aux voisins. Ce choix permet d'offrir la prise en compte de la *densité* sans générer un trafic supplémentaire qui engorgerait davantage le réseau. Le réseau ne diffuse donc pas plus de paquets pour prendre en charge la densité.

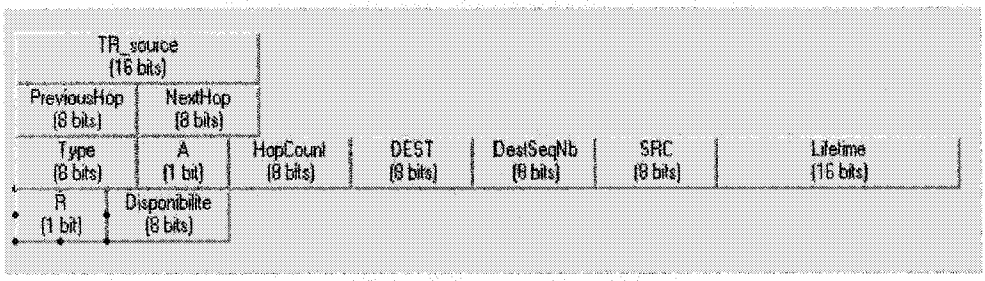


Figure 4.8 Représentation d'un paquet RREP

Modification apportées au niveau de la machine à états finis

Comme nous le montrons à la Figure 4.9, nous avons ajouté un état non bloquant (*Densite_Timeout*) responsable du calcul à intervalle de temps réguliers de la variable d'état densité. Cette variable est évaluée en dénombrant dans le tableau des voisins (variable d'état) les nœuds dont la disponibilité est supérieure à 100. Une fois ce calcul terminé, le tableau des disponibilités est réinitialisé pour ne prendre en compte que les nœuds qui ont diffusé leur disponibilité dans l'intervalle de temps écoulé. Avant de sortir de cet état, nous planifions la prochaine interruption pour le calcul de la densité.

4.3.2 Implémentation des améliorations basées sur le critère de densité

Nous présentons d'abord l'implémentation de notre méthode de choix de route, puis nous expliquons l'implémentation de la technique permettant d'empêcher le re-routage local à partir d'un nœud où la densité est faible.

1) Choix de la route

Les variables d'état

Implémenter une technique de choix de route demande d'ajouter plusieurs variables d'état dans chaque nœud :

- *tableau_notes[N][N]* : il regroupe l'ensemble des notes correspondant à chaque route pour une source donnée.
- *RREQreçus[N][N]* : il regroupe un pointeur sur chaque paquet RREQ reçu et en attente de la sélection de la meilleure route.
- *Nb_RREQreçus[N]* : il comptabilise le nombre de RREQs pour une destination donnée.
- *Finattenterreq[N]* : il contient l'instant de la fin de l'attente pour la route en cours de construction pour une source donnée.
- *routechoisie[N]* : drapeau qui permet de retenir si la route a déjà été choisie pour une destination donnée à la réception d'un RREQ.

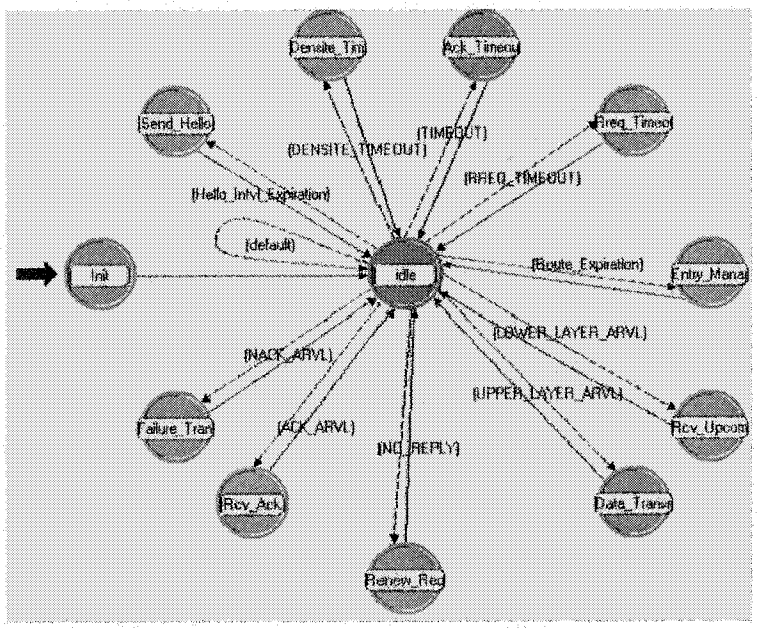


Figure 4.9 Représentation de la nouvelle machine à états finis

Le format des paquets RREQ

Comme indiqué à la Figure 4.10, nous ajoutons trois champs aux paquets RREQs échangés :

- un champ de 32 bits de type *float* pour la densité moyenne le long de la route en construction. Ce champ est mis à jour tout au long de la propagation du RREQ dans le réseau ;
- un champ de 8 bits de type *integer*. Ce champ stocke le nombre d'étranglements de la route en cours de construction ;
- un champ de 1 bit de type *integer*. Ce champ indique si le paquet est envoyé en mode densité.

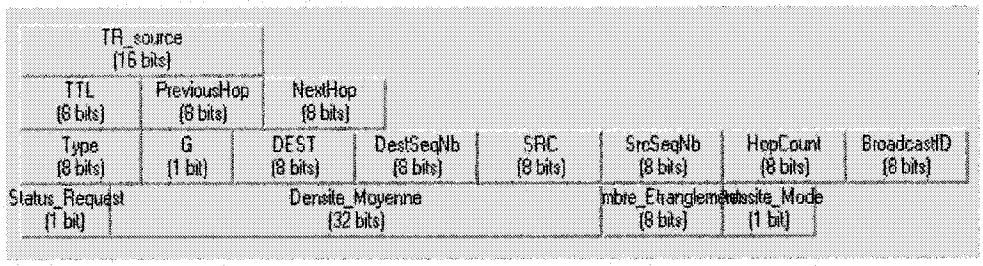


Figure 4.10 Représentation d'un paquet RREQ

Modification au niveau de la machine à états finis

Nous ajoutons un état non bloquant responsable du choix de la route en fin de phase de recherche de route. Cet état s'exécute automatiquement une fois écoulé le délai d'attente déclenché après la réception du premier paquet RREQ à la destination. Le temps d'attente est proportionnel à la durée de la propagation du premier RREQ entre la source et la destination avec un coefficient d'attente x. Ce temps d'attente additionnel rallonge la durée de la recherche de route mais améliore les chances de succès de réparation locale de la route suite à une rupture de lien. Si la durée de l'initialisation du routage est T_r dans une configuration sans sélection de route, nous pouvons majorer la durée du routage par la relation (4.1) dans le cas de la sélection de route :

$$T_{rx} = T_r * (1+x) \qquad (4.1)$$

Il découle de ces observations qu'il est primordial de minimiser le coefficient x de façon à ne pas prendre un temps de routage excessif tout en maintenant une attente suffisante pour que d'autres RREQs parviennent à la destination. Après plusieurs expériences de calibrage, il apparaît qu'un coefficient x pris égal à 2.2 semble le plus adapté. Une fois le temps d'attente écoulé, nous appliquons la fonction élaborée au chapitre III sur chacun des paquets reçus et générons un paquet RREP le long de la route sélectionnée. Nous modifions la gestion de la réception des paquets de RREQ par les nœuds intermédiaires de manière à ajouter/mettre à jour les champs de densité moyenne et de nombre d'étranglements pour la route en construction dans les RREQ en cours de propagation vers la destination.

2) Empêcher le re-routage local à partir d'un nœud où la densité est faible

Nous ajoutons une nouvelle condition : la réparation locale est tentée seulement à condition que la densité soit strictement supérieure à deux. Dans le cas contraire, un RERR est directement envoyé en direction de la source de la communication pour entreprendre une réparation globale de la route. L'ajout de condition est réalisé dans la fonction chargée de la réparation de liens : aodv_link_repair_attempt. Nous apportons cette modification pour éviter la perte de temps de routage qu'occasionnerait une tentative de réparation locale de route de toute façon vouée à l'échec (si la densité est inférieure à trois) et accélérer de ce fait la phase de re-routage dans des conditions pourtant hostiles.

4.4 Exemple de mise en œuvre

Nous étudions à travers un exemple détaillé en quoi les modifications apportées au code initial affectent le comportement du réseau. Prenons le cas de figure représenté à la Figure 4.11.

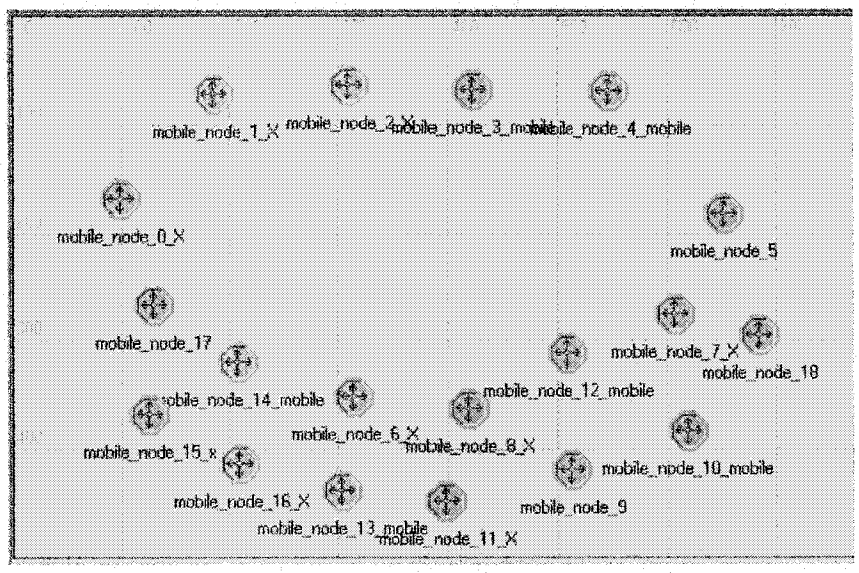


Figure 4.11 Répartition spatiale des nœuds

Mise à jour de la densité

Chaque intervalle de temps HELLO_INTERVAL, les 19 nœuds du réseau diffusent à leurs voisins des paquets RREP d'un genre particulier où la destination n'est autre que la source. Nous précisons ici que les diffusions de paquets RREP pour les 18 nœuds ne sont pas simultanées car l'initialisation s'effectue à des instants aléatoires pour chaque nœud. Nous évitons ainsi une situation de saturation du réseau provoquée par la diffusion au même instant et par tous les nœuds du réseau de paquets RREP.

Les paquets RREP permettent de signaler leur présence aux éventuels voisins. Au moment de la création d'un paquet, le nœud affecte son champ *disponibilité* par la valeur prise par sa variable d'état *disponibilité* à cet instant.

Lorsqu'un nœud mobile réceptionne un paquet RREP ayant ces caractéristiques, il met à jour dans son tableau *disponibilite_voisins[N]* l'entrée correspondant à ce nœud. Tous les intervalles de temps DENSITE_INTERVAL, une nouvelle interruption est programmée. Un nœud qui entre dans cet état dénombre les nœuds dont la disponibilité stockée dans son tableau de disponibilité est supérieure à 100 et affecte la densité avec la valeur trouvée. Avant de sortir de cet état, on programme la prochaine interruption et on réinitialise chaque entrée du tableau *disponibilite_voisins[N]*.

En suivant l'exemple de la Figure 4.11, le nœud mobile 6 reçoit des paquets RREP provenant de ses 5 voisins 14, 16, 13, 11 et 8 (la portée de communication est fixée à 160 m pour chaque nœud). Il extrait la disponibilité de ces nœuds et affecte le tableau *disponibilite_voisins[N]* avec ces données. Lors de l'interruption pour le calcul de la densité, il entre dans l'état *Densite_Timeout* et décompte cinq nœuds dans le tableau *disponibilite_voisins[N]* pour lesquelles la disponibilité est supérieure à 100. La densité au nœud 6 est donc égale à 5.

Recherche initiale de route

Le nœud source 0 génère des paquets de données au niveau application pour une application du nœud destination 5. Au niveau de la couche réseau de la source, le premier paquet de données reçu est mis en file car aucune route pour la destination n'est disponible dans la table de routage *Routingtable*. La source déclenche donc la phase de recherche de route pour la destination en diffusant un paquet RREQ avec le champ source (resp. destination) affecté à 0 (5) et les champs densité moyenne et nombre d'étranglements affectés en fonction de sa densité locale. Les nœuds qui reçoivent ce paquet pour la première fois mettent à jour leur base de données locale *Routingtable* pour la destination d'adresse 0 (en enregistrant en particulier l'identité du prochain nœud) et le rediffusent après avoir mis à jour la densité moyenne et incrémenté le nombre d'étranglements en fonction de leur densité locale. Les paquets RREQ se propagent de cette manière le long du réseau.

Ainsi, le nœud 6 reçoit un paquet RREQ pour la destination 5 ayant transité par les nœuds 17 et 14. Dans ce paquet RREQ, le champ densité moyenne est 3 et le nombre d'étranglements est 1. Le nœud 6 ajoute dans sa base de données une entrée pour la destination 0 en indiquant que le prochain nœud pour cette destination est 14. Puis, le nœud 6 met à jour le champ densité moyenne en lui affectant la valeur 3.5 et le champ nombre d'étranglements à 1. Le paquet RREQ peut alors être rediffusé.

Le premier paquet qui arrive à la destination est le paquet ayant transité par les nœuds 1, 2, 3 et 4. Un pointeur sur ce paquet est stocké dans la première case du

tableau *RREQreçus[0].* Nous évaluons également la note pour la route correspondante à l'aide de la formule de notation établie au chapitre III. Nous obtenons 0.095. Cette note est stockée dans la première case du tableau *tableau_notes[0].* On incrémente alors la variable *Nb_RREQreçus[0]* qui permet de pointer la case suivante des tableaux en cas d'arrivée d'un nouveau RREQ. Un délai de garde (*time-out*) dont la valeur dépend du temps mis par le paquet pour se propager jusqu'à la destination est aussitôt lancé. Dans cet intervalle, la destination stocke chaque nouveau paquet RREQ qui lui parvient dans les cases suivantes successives du tableau en suivant le même processus. Ainsi, un autre paquet RREQ ayant transité par les nœuds 17, 14, 6, 8, 12 et 7 est traité par la destination. La longueur de la route détectée est ici égale à 7. La note attribuée à cette route est : 0.28 (densité moyenne : 3.625 et 2 étranglements)

À échéance du *time-out*, une interruption est générée. On détermine alors, parmi les deux notes stockées dans le tableau *tableau_notes[0],* laquelle est la plus élevée. On identifie le numéro d'arrivée du paquet RREQ correspondant à cette route. Il s'agit du numéro 2. Le pointeur stocké dans *RREQreçus[0][1]* permet alors la génération du paquet RREP qui sera retourné vers la source 0. Chaque nœud intermédiaire qui reçoit le paquet RREP met à jour sa base de données locale vers la destination 5 en enregistrant en particulier le prochain nœud vers 5. Lorsque le paquet RREP parvient à source 0, l'entrée *RequestSent[5]* correspondant à la destination est effacée. Les paquets de données mis en file peuvent être envoyés à la destination le long de cette nouvelle route.

Réparation locale

Étudions comment s'effectue la réparation en cas de disparition du nœud 12. Le nœud 8 qui le précède sur la route détecte sa disparition au niveau couche MAC lorsqu'il ne reçoit pas d'accusé de réception émanant de 12, suite à un envoi de paquets de données. Un paquet NACK est alors retourné à la couche réseau. Le nœud 8 met en file le paquet DATA correspondant et procède à une tentative de réparation locale. En effet, sa densité est supérieure à deux. Il commence donc par passer le statut de la route vers 5 à UNDER_REPAIR dans la *routingtable* pour que les paquets qui arriveront

ultérieurement soient mis en file. Le nœud 8 prévoit aussi une interruption pour prendre en charge le cas où le re-routage local n'aboutirait pas. Puis, il envoie des paquets RREQ vers 5 suivant le même principe que lors du routage initial, mais avec la différence notable que cette fois le champ *Densite_Mode* est désactivé dans les paquets RREQ envoyés. Les paquets se propagent donc dans le réseau sans prendre en compte la densité moyenne et le nombre d'étranglements. Le premier RREQ parvenu à la destination 5 déclenche immédiatement la génération d'un RREP adressé au nœud 8. Ce paquet se propage jusqu'au nœud 8. Une fois parvenu à 8, le statut de la route est repassé à ACTIF et les paquets stockés peuvent être envoyés à la destination le long de la nouvelle route.

4.5 Plan d'expérience

Dans notre approche, nous cherchons à exploiter l'hétérogénéité de la répartition des nœuds mobiles dans le réseau par la prise en compte du paramètre densité. Il serait donc inutile de baser les tests de notre amélioration sur une configuration aléatoire statistiquement homogène. Nous avons néanmoins mené plusieurs simulations dans de telles configurations pour arriver à la conclusion que, pour ces configurations, notre contribution apporte peu de gains de performance sensibles par rapport au protocole AODV classique. Pour réaliser nos tests, nous avons préféré nous baser sur une configuration hétérogène donnée, puis la faire évoluer par le départ de nœuds.

Dans la configuration présentées à la Figure 4.11, le nœud 0 cherche à établir une communication avec le nœud 5. Tous les autres nœuds se comportent ici comme des routeurs. Plusieurs types de routes sont envisageables. Parmi ces routes, la route passant par les nœuds 1, 2, 3 et 4 est la plus courte. La durée des simulations est fixée à 4 minutes. Pour tester nos hypothèses, nous planifions la "disparition" d'un nœud à un instant précis. La disparition du nœud est orchestrée par son déplacement hors de la portée radio de tous les autres nœuds mobiles du réseau. Nous choisissons de transférer le nœud choisi au point d'abscisse 900 et d'ordonnée 400 au bout de 100 secondes de simulation. Nous nous intéressons alors à l'évolution du système et en particulier comment s'opère la réparation de la route.

Nous décidons d'envisager des déplacements des nœuds mobiles 1, 4, 12 et 17. Ce choix s'appuie sur le fait que l'étude des déplacements des autres nœuds est soit impossible (comme le déplacement des nœuds 0 et/ou 5 qui entraverait définitivement la communication) soit déductible des déplacements envisagés (par exemple, le déplacement du nœud mobile 6 aurait des effets comparables à celui du déplacement de 8. Bien sûr, des combinaisons de déplacements peuvent aussi être envisagées, comme le départ de 2 ou 3 nœuds simultanément, mais nous ne les traitons pas car elles sont peu probables.

Nous considérons dans chaque simulation les trois cas suivants :

Cas 1 : Protocole AODV sans prise en compte du paramètre densité et sans re-routage local. Il s'agit en fait du protocole initial dépourvu de sa capacité de prise en charge des réparations locales de route.

Cas 2 : Protocole AODV initial .

Cas 3 : Protocole AODV avec prise en compte du paramètre densité et re-routage local en fonction de la densité (en particulier, le re-routage local est interdit lorsque la densité est inférieure à 3. Il s'agit du protocole amélioré que nous avons proposé.

Les différents résultats sont présentés à la section suivante et correspondent à des moyennes sur des séries de simulations. L'intérêt de ces batteries de tests est de gommer les effets imputables au hasard dans les simulations pour ne conserver que les résultats interprétables.

4.6 Tests et analyse des résultats

Disparition du nœud 4

La courbe donnant l'évolution du délai de transmission (ETE Delay) est représentée à la Figure 4.12 pour les cas 1 et 2, et à la Figure 4.13 pour le cas 3. Le Tableau 4.1 récapitule les résultats obtenus au cours des simulations. Dans les deux premiers cas, nous observons que les courbes du délai ainsi que les données recueillies sont fortement influencées par le départ du nœud 4. Le délai moyen qui s'établit en

moyenne à 0.0095 seconde avant la disparition du nœud 4 passe ainsi à 0.014 seconde et la route utilisée se rallonge en terme de nombre de nœuds. Ces observations s'expliquent par le fait que ce nœud fait partie de la route utilisée pour le transport des données vers la destination 5 lorsqu'il disparaît.

Dans le cas 1, le nœud 3 détecte que le nœud 4 a disparu et retourne immédiatement un message RERR au nœud 0. Celui-ci déclenche alors une opération de re-routage global. Comme il apparaît dans le Tableau 4.1, cette phase de re-routage dure plus longtemps que la phase initiale (0.018 s contre 0.0092 s).

Le cas 2 est semblable en tout point au cas 1. La seule différence réside dans le fait que le nœud 3 tente d'abord une réparation locale de route avant de générer un RERR vers la destination. Cette tentative infructueuse s'avère non seulement très coûteuse en temps (il faut 0.548 s pour que le nœud se résigne à générer le RERR) mais aussi en terme de paquets détruits car, durant cette tentative de réparation locale, les paquets en direction de la destination continuent d'affluer vers la destination. Ce re-routage local a donc des conséquences désastreuses pour le système.

Dans le cas 3, les paquets empruntent un chemin dont le nœud 4 ne fait pas partie. Nous n'observons donc aucun changement. Le délai reste inchangé avant et après le départ du nœud 4. Il s'établit à 0.014 seconde. Nous notons par contre que le temps de recherche de route initiale est nettement plus long que dans les deux cas précédents. Nous expliquons cette observation par le fait que nous attendons d'autres RREQs au niveau de la destination avant de retourner un RREP à la destination sur la route choisie.

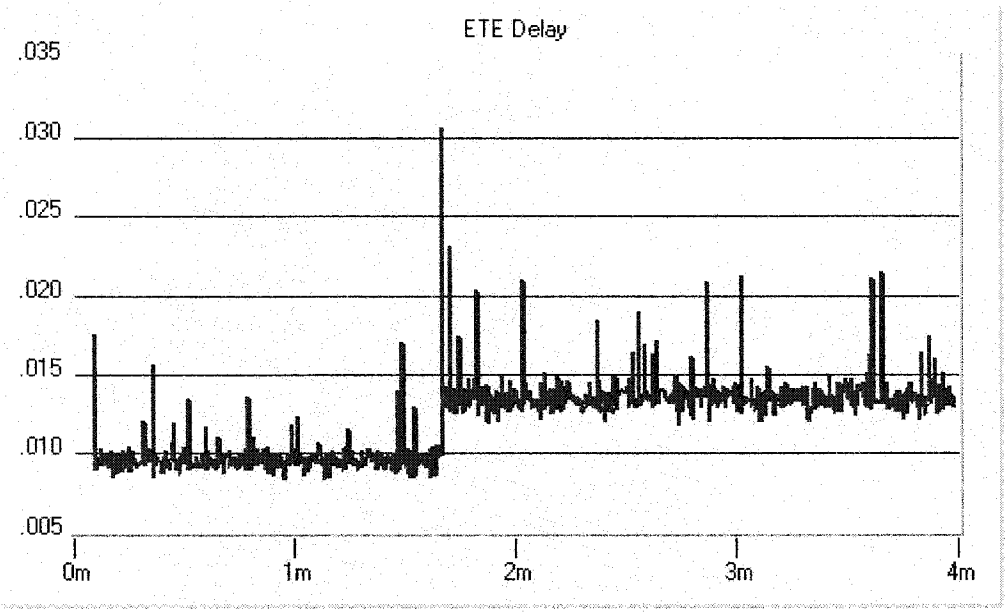


Figure 4.12 Délai ETE – départ du nœud 4 – cas 1 & 2

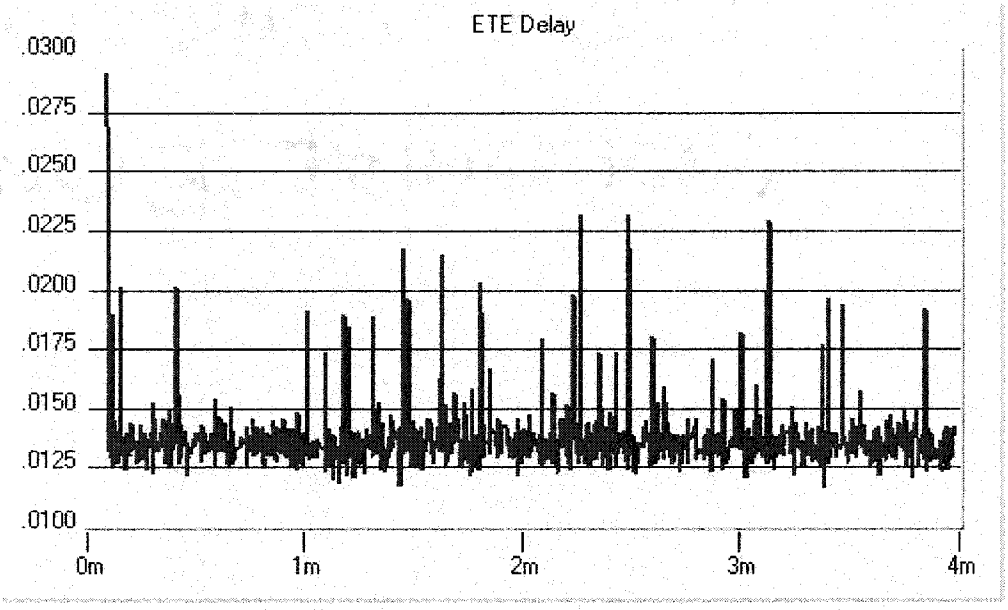


Figure 4.13 Délai ETE – départ du nœud 4 – cas 3

Tableau 4.1 Tableau récapitulatif des résultats pour le départ du nœud 4

	Cas 1	Cas 2	Cas 3
Nombre de paquets perdus	**2**	**4.6666667**	**0**
Tps routage initial (en seconde)	**0.0092**	**0.0084833**	**0.019729**
Tps réparation locale (en seconde)	**0**	**0.5485933**	**0**
Temps total re-routage (en seconde)	**0.0181667**	**0.5627**	**0**
Délai moyen avant re-routage (en seconde)	**0.0095**	**0.0095**	**0.014**
Délai moyen après re-routage (en seconde)	**0.016**	**0.014**	**0.014**
Longueur moyenne des routes (en nœuds)	**6.56307**	**6.1969993**	**7**

Disparition du nœud 1

Pour la disparition du nœud 1, nous obtenons dans les trois cas des courbes de délai ETE semblables à celles obtenues pour la disparition du nœud 4. Un récapitulatif des données recueillies est fourni au Tableau 4.2. Nous notons que ce tableau ressemble beaucoup au tableau précédent. Quelques différences méritent cependant d'être soulignées. En particulier, le temps de re-routage dans le cas 1 a beaucoup augmenté et

est passé à 0.3358827 s. Ce temps de re-routage excessivement long s'explique par le fait que la recherche de route subséquente à la disparition du nœud 1 n'intervient qu'à l'arrivée d'un nouveau paquet en provenance de la couche supérieure. Ce temps de re-routage est donc fortement lié au temps d'inter-arrivées des paquets. Par ailleurs, nous remarquons qu'aucun paquet n'est perdu dans le re-routage. Dans le cas 2, 4 paquets sont encore perdus dans le re-routage et le temps de re-routage est très long. La raison est identique que pour la disparition du nœud 4.

Disparition du nœud 12

Les données recueillies pour les cas 1 et 2, et en particulier la Figure 4.14, révèlent qu'aucun re-routage n'est entrepris suite au départ du nœud 12. Nous pouvions le prévoir car le nœud 12 ne fait pas partie de la route utilisée dans les cas 1 et 2. Dans le cas 3 par contre, il s'avère que les résultats sont fortement affectés. Ainsi, comme en atteste la Figure 4.15, le délai ETE passe de 0.014 seconde avant la disparition du nœud 12 à 0.016 seconde. Nous observons aussi dans le Tableau 4.3 qu'un re-routage local a été entrepris avec succès. Nous notons que ce re-routage local dure 0.00782 seconde. ce temps de re-routage est à comparer à 0.018 seconde du re-routage global obtenu dans le cas 1 pour la disparition du nœud 4. Nous en déduisons que le délai de re-routage est nettement amélioré par rapport au cas 1. Par ailleurs, nous constatons qu'aucun paquet n'est perdu dans les procédures de gestion de la disparition du nœud 8. Nous expliquons ce bon résultat, à opposer aux paquets perdus dans les cas 1 et 2 lors de la disparition du nœud 4, par le fait que le re-routage local est un succès. Les paquets stockés pendant la phase de re-routage local peuvent donc tous être envoyés à la destination.

Tableau 4.2 Tableau récapitulatif des résultats pour le départ du nœud 1

	Cas 1	Cas 2	Cas 3
Nombre de paquets perdus	**0**	**4**	**0**
Tps routage initial (en seconde)	**0.0089333**	**0.0095833**	**0.019729**
Tps réparation locale (en seconde)	**0**	**0.5545713**	**0**
Temps total re-routage (en seconde)	**0.3358827**	**0.5677333**	**0**
Délai moyen avant re-routage (en seconde)	**0.095**	**0.095**	**0.014**
Délai moyen après re-routage (en seconde)	**0.014**	**0.014**	**0.014**
Longueur moyenne des routes (en nœuds)	**6.621067**	**6.1970783**	**7**

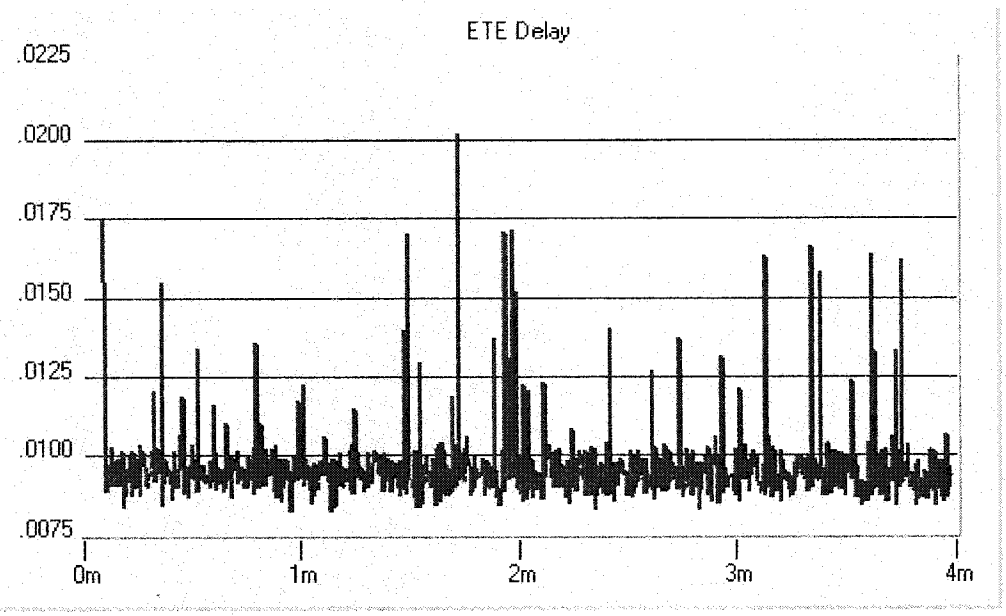


Figure 4.14 Délai ETE – départ du nœud 12 – cas 1 & 2

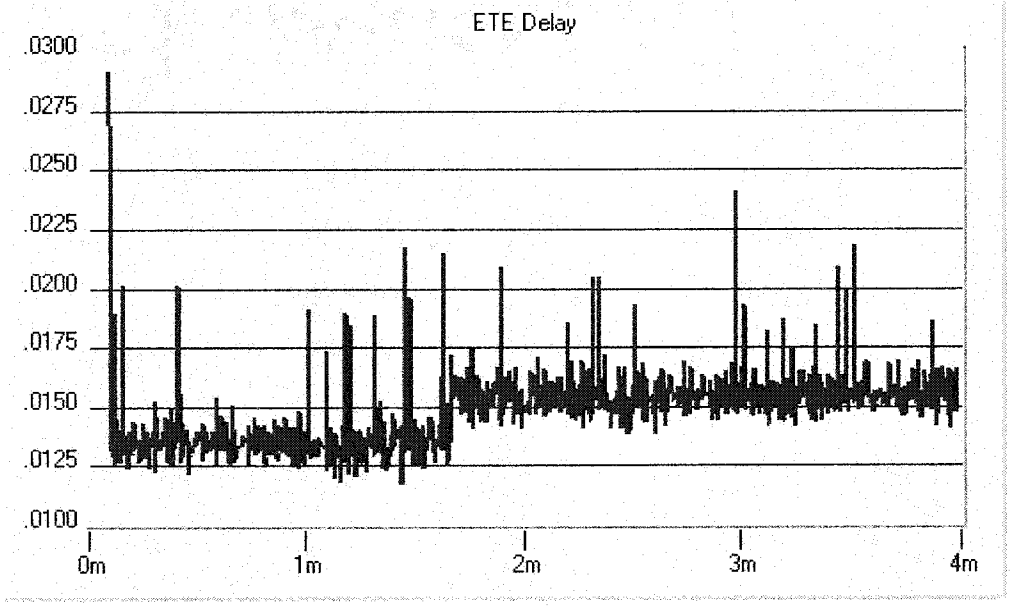


Figure 4.15 Délai ETE – départ du nœud 12 – cas 3

Tableau 4.3 Tableau récapitulatif des résultats pour le départ du nœud 12

	Cas 1	Cas 2	Cas 3
Nombre de paquets perdus	**0**	**0**	**0**
Tps routage initial (en seconde)	**0.00855**	**0.00855**	**0.022013**
Tps réparation locale (en seconde)	**0**	**0**	**0.00782**
Temps total re-routage (en seconde)	**0**	**0**	**0.00782**
Délai moyen avant re-routage (en seconde)	**0.0095**	**0.0095**	**0.014**
Délai moyen après re-routage (en seconde)	**0.0095**	**0.0095**	**0.016**
Longueur moyenne des routes (en nœuds)	**5**	**5**	**8.261543**

Disparition du nœud 17

Comme pour la disparition du nœud 12, seul le cas 3 est affecté par la disparition du nœud 17. Nous obtenons l'évolution du délai ETE représentée à la Figure 4.16. Nous observons ainsi que le délai diminue de 0.014 s avant la disparition du nœud à 0.0095 s après sa disparition. Ces résultats obtenus pour le cas 3 sont à mettre en parallèle avec ceux obtenus pour les cas 1 et 2 lors de la disparition du nœud 1. Nous remarquons que

la durée de la recherche de route qui suit la disparition du nœud 17 est 0.039 seconde. Cette durée est largement inférieure aux durées de re-routage mesurées dans les cas 1 et 2 pour le départ du nœud 1.

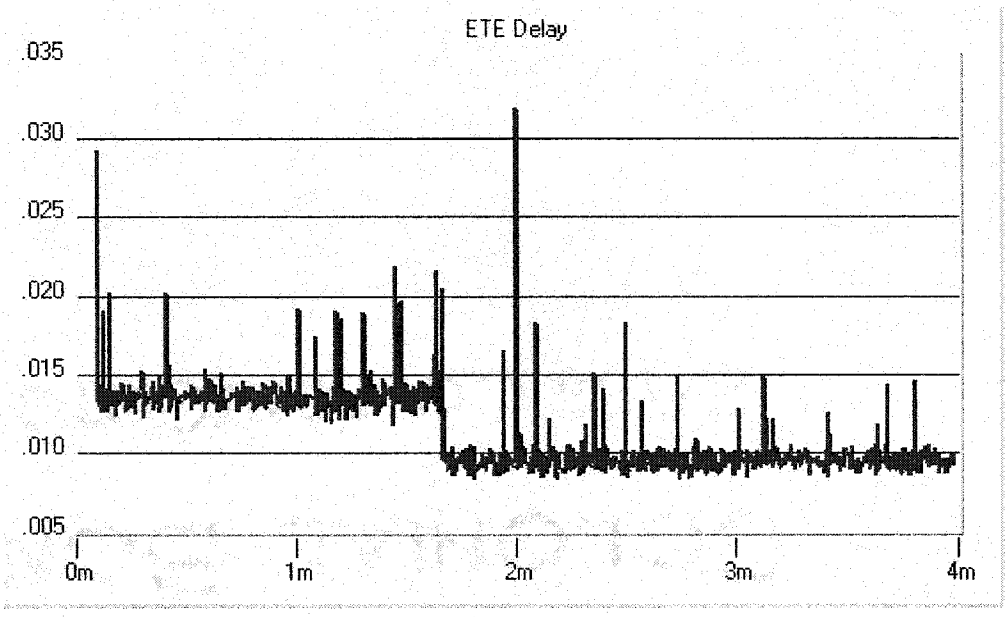


Figure 4.16 Délai ETE – départ du nœud 17 – cas 3

Tableau 4.4 Tableau récapitulatif des résultats pour le départ du nœud 17

	Cas 1	Cas 2	Cas 3
Nombre de paquets perdus	**0**	**0**	**0**
Tps routage initial (en seconde)	**0.00855**	**0.00855**	**0.019756**
Tps réparation locale (en seconde)	**0**	**0**	**0.039012**
Temps total re-routage (en seconde)	**0**	**0**	**0.039012**
Délai moyen avant re-routage (en seconde)	**0.0095**	**0.095**	**0.014**
Délai moyen après re-routage (en seconde)	**0.0095**	**0.095**	**0.0095**
Longueur moyenne des routes (en nœuds)	**5**	**5**	**5.9371097**

Bilan de la disparition d'un nœud

Nous cherchons à étendre nos résultats obtenus pour la défaillance des nœuds 1, 4, 12 et 17 à l'ensemble des nœuds du réseau. Ainsi, nous envisageons la défaillance de tous les nœuds mobiles faisant partie d'une route initiale. Nous n'envisageons pas par contre les cas de défaillance des nœuds 0 ou 5 (leur défaillance serait irrécupérable pour la communication) ainsi que les nœuds 9, 10, 11, 13, 15, 16 ou 18. La défaillance de ces

derniers a en effet une incidence limitée sur le système. Nous supposons que la défaillance de chaque nœud du système est équiprobable. Nous classons les nœuds en plusieurs groupes suivant le cas auquel nous faisons référence.

Nous obtenons les répartitions suivantes :

Cas 1 et 2 :

La disparition du nœud 12 est représentative de la disparition des nœuds suivants : 6, 7, 8, 12, 14, 17. La disparition du nœud 4 est représentative de la disparition des nœuds 2, 3, et 4.

Cas 3 :

La disparition du nœud 4 est représentative de la disparition des nœuds suivants : 1, 2, 3 et 4. La disparition du nœud 12 est représentative de la disparition des nœuds 6, 7, 8, 12 et 14.

À l'aide de ces répartitions, nous pondérons les différents scénarii de disparition pour obtenir les résultats synthétiques présentés au Tableau 4.5.

Tableau 4.5 Résultat de comparaison de 3 variantes de AODV

	Cas 1	**Cas 2**	**Cas 3**
Mode	**AODV sans re-routage local**	**AODV avec re-routage local**	**AODV avec choix de route**
Nombre de paquets perdus	0.6	1.8	**0**
Durée du routage initial (en s)	0.00878	0.008632	**0.0205**
Durée totale du re-routage (en s)	0.039	0.22561	**0.0058**
Délai moyen avant (en s)	0.0095	0.0095	**0.014**
Délai moyen après (en s)	0.119	0.0113	**0.0146**
Longueur de route (en nœuds)	5.63	5.47	**7.52**

Nous remarquons dans ce tableau que le cas 3, c'est à dire le cas faisant intervenir la version de AODV munie de nos améliorations, permet d'éviter complètement la perte de paquets lors de l'occurrence du départ d'un nœud quelconque du réseau. Ce résultat met en exergue une amélioration sensible de la fiabilité du protocole de routage par rapport aux cas où la réparation locale n'est pas implémentée ou alors réalisée sans choix de route (cas 1 & 2). Dans ces deux cas, le départ d'un nœud quelconque du réseau engendre la perte en moyenne de 0.6 paquet de données pour le cas 1 et 1.8 paquets de données pour le cas 2.

Nous constatons cependant que le temps de routage initial est systématiquement plus long pour le cas 3. En moyenne, le temps de routage est ainsi augmenté de 137% par rapport aux cas 1 et 2. Cela constitue le principal inconvénient inhérent à notre amélioration. Mais cette attente plus longue au niveau de la phase d'établissement de route est nécessaire pour choisir une route mieux réparable. Nous notons ainsi que le délai moyen de re-routage occasionné par le départ d'un nœud du réseau dans le cas 3 est sensiblement meilleur que les cas 1 et 2. Ainsi, le temps d'attente est en moyenne 6.7 fois plus court que dans le cas 1 et 38 fois plus que dans le cas 2. Cette observation atteste donc de l'amélioration sensible de la prise en charge de la qualité du service par le protocole amélioré proposé par rapport à la version initiale.

Une faiblesse de notre amélioration s'observe au niveau de la longueur moyenne de la route. Nous constatons que la route est en moyenne égale à 7.52 nœuds dans le cas 3 contre environ 5.5 nœuds pour les 2 autres cas. La route met donc en jeu plus de nœuds et, par conséquent, la probabilité pour que survienne une défaillance augmente sur la route retenue par le cas 3.

Une autre faiblesse de notre amélioration réside dans le fait que les délais ETE des paquets de données sont plus importants dans notre version, comparativement à la version initiale. Dans le cas 3, le délai ETE s'établit avant la défaillance à 0.014 s contre 0.0095 s pour les deux autres cas. Nous constatons par contre qu'en moyenne, ce délai augmente peu après la défaillance et s'établit alors à 0.0146 s contre 0.0119 s pour le cas

1 et 0.0113 s pour le cas 2. Le délai ETE obtenu dans le cas 3 est donc plus stable que celui observé dans les autres cas. Nous en concluons qu'à ce niveau là encore notre amélioration apporte un gain en terme de gestion de la qualité de service en cas de défaillance.

CHAPITRE V
CONCLUSION

Ce chapitre se veut une synthèse des travaux réalisés dans le cadre de ce mémoire ainsi qu'une présentation des travaux à accomplir pour améliorer les mécanismes de prise en compte de la densité dans les réseaux ad-hoc.

5.1 Synthèse des travaux et originalité des contributions

Les travaux présentés dans ce mémoire visent à améliorer la gestion de la qualité de service dans les réseaux mobiles ad-hoc par la prise en compte de la *densité* définie comme étant le nombre d'unités mobiles *disponibles* dans la portée radio d'un nœud. Notre approche s'est appuyée sur une analyse approfondie des outils de prise en compte de la qualité de service dans les réseaux ad-hoc en vue d'en dégager les forces/faiblesses. Nous avons alors imaginé la notion de densité et décrit comment un nœud peut exploiter cette information pour améliorer la qualité de service offerte. Nous avons dégagé deux axes :

- instauration d'un mécanisme de choix de route ;
- prévision anticipée de l'échec d'un re-routage local.

Le mécanisme de choix de route vise à sélectionner parmi plusieurs concurrentes celle dont la maintenance est la plus facile à réaliser. Les résultats des simulations confirment notre raisonnement théorique : les temps de re-routage sont améliorés. Par ailleurs, notre mécanisme permet également d'améliorer les taux de pertes de paquets. La qualité de service s'en trouve donc encore renforcée.

Nous avons également mis en évidence une situation dans laquelle une tentative de re-routage local est néfaste au réseau : si la densité est trop faible autour du nœud qui initie le re-routage local, celui-ci est voué à l'échec. Il est donc préférable de retourner directement un RERR à la source directement après avoir détecté une rupture de lien.

Pour valider notre travail, nous avons implémenté notre amélioration sur le protocole AODV à l'aide du logiciel de simulation réseau *Opnet Modeler*. Nous avons alors testé notre protocole dans une configuration hétérogène donnée. Les résultats obtenus sont encourageants : les pertes de paquets lors de la défaillance d'un nœud ont pu être fortement réduites par rapport aux versions initiales. Par ailleurs les temps de re-routage suite à une défaillance sont sensiblement améliorés.

Notre mémoire constitue ainsi un travail original dans le sens où il exploite un nouveau paramètre (la *densité*) qui permet de tirer parti de la spécificité de la répartition géographique des nœuds du réseau l'environnement pour améliorer la gestion de la qualité de service.

5.2 Limitations des travaux

La principale limitation de notre processus de choix de route réside dans la perte de temps dans le routage initial. Cette perte de temps est liée à l'attente de la réception d'autres RREQs en provenance de la source avant de sélectionner la meilleure route au sens de la facilité de réparation. Une autre limitation provient du fait que les routes mises en jeu suite à notre amélioration sont plus longues. Les délais ETE sont donc plus importants.

5.3 Travaux futurs

Nous avons implémenté notre travail sur la base d'un protocole de routage source qui ne réalise pas de réservation de bande passante. La disponibilité des nœuds est donc pour nous une donnée statique complètement indépendante de l'état du réseau. Il serait pourtant intéressant d'implémenter la prise en compte de la *densité* dans le cadre d'un protocole de routage avec réservation de ressources. Il deviendrait alors possible d'ajuster la *disponibilité* d'un nœud à son état d'engorgement. La *densité* d'un nœud, directement liée à la *disponibilité* de ses voisins, refléterait davantage l'état du réseau au voisinage du nœud. Il deviendrait en particulier possible de limiter l'effet d'attraction d'une forte concentration d'unités mobiles fortement engorgées.

Une autre piste de travail serait de réaliser une étude de performance approfondie pour des répartitions des nœuds mobiles inspirées de situations plausibles dans des environnements réels. Dans cette optique, une bonne approche consisterait à étudier les mouvements des individus en milieu urbain pour en dégager les scénarii de déplacement mis en jeu lors des simulations.

Enfin, un autre point important serait de tirer parti de la différence de propagation des ondes radio entre l'intérieur et l'extérieur. En effet, en milieu urbain, les paquets se propagent plus vite à l'extérieur car les nœuds sont plus espacés. Les routes utilisées par la version initiale de AODV sont donc enclines à passer par l'extérieur où la densité est moindre qu'à l'intérieur (et où les nœuds sont de surcroît plus mobiles). Pour améliorer la gestion de la QdS en cas de défaillance d'un nœud, il serait donc préférable de choisir une route le long de laquelle la densité est plus forte à l'aide de notre méthode.

BIBLIOGRAPHIE

[CHA 01] Chakrabarti S. et Mishra A., "QoS issues in ad hoc wireless networks", *IEEE International Conference on Communications*, Février 2001, pp. 142–148.

[SHA 02] Shah S.H et Nahrstedt K., "Predictive location-based QoS routing in mobile ad hoc networks", *IEEE International Conference on Communications*, 2002, pp. 1022 –1027.

[LIA 02] Liao W.H., Tseng Y.C. et Shih K.P., "A TDMA based bandwidth reservation protocol for QoS routing in a wireless mobile ad hoc network", *IEEE International Conference on Communications*, 2002, pp. 3186–3190.

[CHEN 99] Chen S., *Routing support for providing guaranteed end-to-end quality of service*, these de Ph.D., Univ. of IL at Urbana-Champaign, 1999.

[AAD 01] Aad I. et Castelluccia C., "Differentiation mechanisms for IEEE 802.11", *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, Avril 2001, pp. 209-218.

[BLA 98] Blake S., Black D., Carlson M., Davies E., Wang Z. et Weiss W., *An architecture for differentiated services*, RFC 2475, IETF, Décembre 1998.

[BRA 94] Braden R., Clark D. et Shenker S., *Integrated services in the internet architecture : an overview*, RFC 1633, IETF, Juin 1994.

[GER 97] Gerla M. et Lin C.R., "Asynchronous multimedia multihop wireless networks", IEEE Computer and Communications Societies, Avril 1997, pp. 118-125.

[CHA 02] Chaudet C., "Qualité de service et réseaux ad-hoc"l, Inria, France, Novembre 2001.

[LEE 2000] Lee S.B., Ahn G.S., Zhang X, et Campbell A.T., "INSIGNIA : An ip-based quality of service framework for mobile ad hoc network" *Journal of Parallel and Distributed Computing (Academic Press), Special issue on Wireless and Mobile Computing and Communications,* 2000., pp. 374-406.

[XIA 00] Xiao H., Seah W.K.G., Lo A. et Chua K.C., "A flexible quality of service model for mobile ad hoc networks" *IEEE Vehicular Technology Conference,* Japon, Mai 2000, pp. 445-449.

[PER 94] Perkins C.E. et Bhagwat P., "Highly dynamic destination-sequenced distance-vector routing for mobile computer", *ACM Conference on Communications Architectures*, 1994, pp. 234-244.

[JOH 96] Johnson D.B., Maltz D.A., "Dynamic source routing in ad hoc wireless networks", *Mobile Computing*, T.Imielinski et H. Korth, Mc Kluwer Academic Publishers, Mars 1996, pp. 153-181.

[PER 2000] Perkins C.E., Royer E.M. et Das S.R., *Ad hoc on demand distance Vector routing,* IETF, Internet Draft, Mars 2000.

[PER 99] Perkins C.E. et Royer E.M., "Ad hoc on demand distance vector algorithm", *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, USA, Février 1999.