Thèse

THÈSE UNIVERSITÉ DE LILLE 1
pour obtenir le grade de
DOCTEUR EN INFORMATIQUE

présentée par
Nicolas GOUVY
ÉCOLE DOCTORALE : SPI
LABORATOIRE : LIFL – UMR8022
Centre Inria Lille – Nord europe

Routage géographique dans les réseaux de capteurs et actionneurs

Thèse soutenue le 19 Septembre 2013
devant le jury composé de :

Wilfrid Perruquetti
Professeur à l'École Centrale de Lille / *Président*

Isabelle Guérin Lassous
Professeur des Universités à Lyon 1 / *Rapportrice*

Marcelo Dias de Amorim
Directeur de recherche CNRS / *Rapporteur*

Antoine Gallais
Maître de conférence à l'Université de Strasbourg / *Examinateur*

David Simplot-Ryl
Directeur du Centre Inria Lille – Nord Europe
Professeur des Universités à Lille 1 / *Directeur de thèse*

Nathalie Mitton
Chargée de recherche Inria / *Co-directrice de thèse*

The network is the computer.
Sun microsystems

Remerciements

Je tiens tout d'abord à remercier David Simplot-Ryl, et surtout Nathalie Mitton pour m'avoir encadré tout au long de cette thèse.
Je remercie Wilfrid Perruquetti, Professeur à l'École Centrale de Lille, qui mefait l'honneur de présider ce jury.
Je remercie Isabelle Guérin Lassous, Professeur des Universités à Lyon 1, et Marcelo Dias de Amorim, Directeur de recherche CNRS, d'avoir bien voulu accepter la charge de rapporteur.
Je remercie Antoine Gallais, Maître de conférence à l'Université de Strasbourg, d'avoir bien voulu juger ce travail.

Sur un plan plus personnel, je remercie mes parents pour m'avoir soutenu tout au long de mes études. Je remercie ma soeur et mes amis, et particulièrement Paméla et Adel, pour m'avoir soutenu dans les périodes de doute. Merci aussi à Claudine, qui a eu le courage de relire et corriger ce manuscrit alors que rien ne l'y obligeait.

Enfin je remercie toutes les personnes, Directeurs, assistantes – et particulièrement Anne Rejl –, secrétaires, institutions, laboratoires...qui ont d'une façon ou d'une autre contribué à la réussite de cette thèse et du projet MajecSTIC 2012.

À vous tous, un grand merci.

Résumé

Cette thèse se positionne dans le contexte des réseaux sans fil multi-sauts tels les réseaux de capteurs ou les réseaux de capteurs/actionneurs ou encore de robots mobiles. Ces réseaux sont composés d'entités (nœuds) indépendantes (c.-à-d. les robots) possédant des capacités limitées en termes de taille mémoire, de capacité de calcul et sont soumis à des contraintes énergétiques fortes (ces composants reposent sur des batteries). Ils communiquent exclusivement par voie radio, il n'y a donc aucune infrastructure fixe. Pour pouvoir relayer les messages d'un robot à une station de base, on utilise des protocoles dits « de routage» qui ont en charge de déterminer quel robot doit relayer le message, de façon locale et distribuée, sans connaissance globale du réseau.

Nous nous sommes basé sur CoMNet, le premier protocole de routage géographique utilisant la mobilité contrôlée tout en garantissant la connexité de celui-ci. CoMNet va, à chaque routage, relocaliser le prochain saut selon un schéma de relocalisation prédéfini de manière à adapter la topologie du réseau à son trafic, et ce afin d'économiser de l'énergie. CoMNet propose trois schémas de relocalisation différents adaptés à différents environnements, et l'on en choisit un au démarrage du réseau. Toutefois, CoMNet, en faisant bouger le prochain nœud N, va certes adapter la topologie, mais aussi modifier le voisinage de ce même nœud. Quand ça sera à lui de transmettre le message il n'aura plus forcément les mêmes possibilités, ses voisins ayant changé. La relocalisation d'un nœud N va servir de base pour celle de $N + 1$ et les suivants dans le routage.

En réponse à ce problème, nous avons proposé MobileR (Mobile Recursivity). MobileR va, à chaque étape du routage, essayer d'anticiper sur plusieurs sauts pour choisir le prochain nœud. Il va calculer la relocalisation des voisins N et pour chacun d'entre eux les multiples $N + 1$ possibles, etc. MobileR va donc calculer à chaque étape du routage les coûts sur progrès de chacun des chemins (avec nœuds relocalisés) possibles. Le paquet sera alors transmis, au premier nœud du chemin qui minimise ce ratio.

Le principe même de relocaliser les nœuds apporte son lot de nouveaux problèmes. Ainsi, dans les réseaux de capteurs, il y a souvent plusieurs nœuds sources qui détectent un même événement et vont émettre des messages à router vers l'unique station de base. Les chemins de routage de ces différents messages sont physiquement proches – vu qu'ils sont liés à un même événement – et ce d'autant plus qu'on se rapproche de la station de base. Ces chemins vont finir par se croiser, et le nœud de croisement va sans cesse être relocalisé par chacun des chemins.

C'est pourquoi j'ai proposé le protocole de routage PAMAL (PAth Merging ALgorithm) en réponse à un problème introduit par la mobilité. En effet, PAMAL permet de détecter ces intersections et de les gérer localement. Il va arrêter ces oscillations parasites, provoquer une fusion des chemins de routage en amont du nœud d'intersection et une agrégation de paquets en aval. PAMAL propose ainsi une amélioration de la durée de vie du réseau allant jusqu'à 37% avec un mécanisme d'agrégation très simple.

La mobilité contrôlée permet aussi d'envisager de nouvelles solutions à des anciens

problèmes. Le protocole GRR (Greedy Routing Recovery) propose ainsi un mécanisme de récupération pour augmenter le taux de délivrance des messages dans les réseaux de capteurs/actionneurs avec obstacle(s). En effet aucun des protocoles de routage reposant sur des actionneurs n'implémente un procédé pour contourner les obstacles ou les zones de faible densité où le routage glouton simple est impossible. Ils routent tous les messages de manière gloutonne vers la destination. Le routage échoue alors quand un nœud n'a plus de voisin plus proche de la destination que lui-même. C'est pourquoi GRR va, quand le routage glouton simple de proche en proche échoue, appliquer un nouveau schéma de relocalisation qui va permettre de contourner l'obstacle tout en restaurant le routage glouton. L'obstacle va ainsi être circonvenu en relocalisant des nœuds tout autour. Ainsi, les routages suivants seront gloutons. Sans pour autant garantir la délivrance de 100% des messages, nos simulations montrent que le mécanisme de récupération de GRR permet de router avec succès dans 72% des cas sur des topologies où CoMNet échoue dans tous les cas.

Abstract

This thesis is about wireless multi-hop networks such as wireless sensors networks or hybrid sensor/actuator networks and actuator networks. Those kinds of networks are composed of independent entities (nodes, *i.e.* the robots) which have very limited computing and memory capabilities. Moreover, they are battery powered and have to work in an efficient way. They communicate through the radio medium and do not require any static infrastructure. In order to relay messages between actuators up to the base station, we use what is called "routing protocols". In order to be efficient, those protocols have to find a routing path, over multiple robots, in a local and distributed manner, without any global knowledge about the network.

My works rely on CoMNet, the first geographic routing protocol which relies on the controlled mobility of the robots while guaranteeing network connectivity despite their movements. CoMNet relocates next hop node on each routing step, according to a predefined relocation pattern. Its aims to adapt the network topology to the routed traffic in order to save energy. Nevertheless, CoMNet does not consider the consequences of those relocations more than in a one-hop way. By making node N relocating node $N+1$ on the routing path, CoMNet adapts the topology locally. But CoMNet ignores the fact that making this $N+1$ node move changes its routing possibilities too.
We proposed MobileR (Mobile Recursivity), a new routing protocol which takes this phenomena into account. MobileR, at each routing step, anticipates the routing in a multi-hop manner through computations over its one-hop neighbors. It makes the current forwarding node compute all the possible routing paths toward the destination using relocation patterns recursively. The cost-over-progress of each path is computed and the next hop selected is the first hop on the best computed path. Still, the protocol is completely localized and no information, except the forwarding packet, is shared between nodes.

On the one hand, the relocating nodes principle is not without consequences. It brings new issues to solve. For instance, in wireless sensor networks, events are likely to be detected by multiple sensors. Consequently, on events occurrence, multiple sensors transmit message toward the destination. But those source nodes are geographically close, as they report the same event. Hence the routing paths for the data they transmit are very close and even merge close to the destination. This phenomena has to be considered. In current routing protocol for actuators, those common routing path parts provoke useless oscillation and premature node death as relocating scheme compete for the same nodes.
As a response to this phenomena, I propose the PAMAL (PAth Merging ALgorithm) routing algorithm. PAMAL detects those routing path crossing and handles them in a purely localized way. It makes nodes oscillation stop and provokes a path merging upstream and uses a packet aggregation downstream. Thanks to this behavior, PAMAL makes the network lifetime increase up to 37% with the simplest possible aggregation.

But on the other hand, controlled mobility also makes possible new answers to old routing issues in wireless sensors networks. The Greedy Routing Recovery (GRR) routing protocol takes controlled mobility into account in order to increase delivery rate on topology with holes or obstacles. Indeed, None of the few existing routing protocols for actuator networks proposes any mechanism to bypass obstacles or topology holes where greedy routing is impossible. They all rely on a greedy forwarding routing strategy toward destination node. Hence they all fail when the current holding node has no neighbor closer to the destination than itself.
GRR includes a dedicated relocation pattern which will make it bypass routing holes and create a routing path on which greedy forwarding will be possible. The obstacle, or hole, is circumvented by relocating nodes all around. Thanks to this light recovery, next routing are going to be in a fully greedy forwarding way.Simulation results show that the light recovery mechanism we use in GRR has a hit ratio of 72% over network topologies where traditional CoMNet fails.

Table des matières

Chapitre 1

Introduction

Nous vivons aujourd'hui dans un monde hyper connecté. En effet, les progrès de l'électronique et de nouvelles technologies ont permis l'émergence de nouveaux moyens de communication. D'abord totalement statiques, à cause des besoins d'infrastructures adaptées, les dispositifs de connexion à ces réseaux sont rapidement devenus transportables, puis portables avec la généralisation des technologies sans fil comme le WiFi ou les réseaux cellulaires. C'est le passage du filaire au WiFi qui a rendu les ordinateurs vraiment portables. Les derniers réseaux cellulaires (2.5G, 3G, 4G) ont permis eux l'internet au creux de la main avec les PDA, puis les smartphones. On le voit, le monde hyper-connecté devient aussi hyper-mobile. Les technologies continuent d'évoluer et proposent depuis quelque temps déjà des réseaux sans infrastructure fixe, ou réseaux *ad-hoc*. Ces nouveaux réseaux ont permis l'émergence de nouvelles applications difficilement envisageables autrement, comme les réseaux véhiculaires, les systèmes de surveillance sans fil, etc.

Le monde de demain sera encore plus connecté, on parle même de réseaux ubiquitaires. Parmi ces réseaux *ad-hoc* qui émergent, on distingue les réseaux de capteurs. Ce sont des réseaux dont les éléments constitutifs, les capteurs, sont particulièrement contraints. Chaque capteur consiste en un petit système embarqué aux ressources matérielles très limitées. Ces réseaux sont dès lors généralement employés pour la détection, le traitement, et la transmission de données relatives à un événement ou à une requête. Le nombre de capteurs y varie d'une poignée à plusieurs milliers. Pour des raisons de coût, les capacités techniques des capteurs sont limitées et ils sont souvent alimentés par des batteries afin de se fondre dans l'environnement. Cela impose une forme de coopération entre tous ces éléments communicants car à ces limites techniques vient s'ajouter la nature dynamique et peu fiable des communications radio. Dès lors, il s'agit de concevoir des protocoles de communication, appelés protocoles de routage, conçus avec les contraintes de ces réseaux.

La première approche pour le routage dans les réseaux développée pour ces réseaux fut l'inondation. Un capteur qui a un message à transmettre va alors le retransmettre à

tous les capteurs qui sont à sa portée. Et ces capteurs vont réitérer le même processus. Cette approche a pour avantage d'être très simple, mais est hautement consommatrice en énergie et bande passante. Elle a pour objectif principal la fiabilité en dé-dupliquant les messages à chaque re-transmission. Néanmoins, les interférences provoquées par les multiples retransmissions simultanées viennent perturber la bonne réception des messages.
Les protocoles de routage suivants se sont alors efforcés de limiter les retransmissions et donc d'augmenter l'efficacité énergétique des protocoles en diminuant le coût du routage. Tout cela a pour but de permettre des réseaux de capteurs fonctionnant le plus longtemps possible sans intervention humaine.

Un des paradigmes de routage les plus populaires aujourd'hui dans les réseaux de capteurs est le routage géographique. Il requiert que les nœuds du réseau soient informés de leur position et de celle de la destination. Dans ces protocoles géographiques, il y a généralement un mécanisme glouton qui permet la transmission des paquets vers la destination. Ce paradigme est très apprécié car il présente les caractéristiques suivantes :

- l'absence de boucles : afin de préserver la durée de vie du réseau, le protocole doit éviter de faire faire des boucles au message à router, quitte à le supprimer pour préserver le réseau.
- la localité : les décisions de routage doivent être prises localement par chacun des éléments du réseau en fonction des informations dont il dispose sur lui et ses voisins directs.
- l'absence de mémoire : les décisions de routage sont prises à chaque étape indépendamment des choix précédents.
- le passage à l'échelle : conséquence de la localité et de l'absence de mémoire, ces protocoles fonctionnent de manière efficace dans des réseaux composés d'une poignée ou de milliers de nœuds.

Avec ce paradigme, combiné à des mécanismes de récupération dans le cas où le routage glouton échoue, on a atteint un seuil d'efficacité difficile à dépasser aujourd'hui dans le cadre de réseaux de capteurs sans fil et statiques.

C'est pourquoi dans cette thèse nous nous sommes intéressé à l'utilisation d'actionneurs, ou de robots, pour améliorer les performances des protocoles des réseaux de capteurs. En effet, de nombreuses études ont montré que l'ajout de robots non seulement ouvre la possibilité de nouvelles applications mais surtout permet de meilleures performances à un coût moindre [WSC08]. L'objectif de cette thèse consiste à étudier et à proposer une famille de protocoles de routage géographique pour les réseaux de capteurs et actionneurs qui garantisse certains critères tels la livraison du message, une consommation minimale d'énergie, en fonction des besoins de l'application. De tels protocoles existent pour des réseaux de capteurs, mais très peu prennent avantage de la mobilité des actionneurs pour améliorer les performances de l'algorithme.
C'est pourquoi, dans ce document, nous proposons un ensemble de protocoles de routage géographique pour réseaux d'actionneurs qui vont tirer avantage de la mobilité pour améliorer le routage en réduisant sa consommation énergétique et en améliorer les

performances globales.

1.1 Contributions de cette thèse

Notre travail s'est axé autour de trois propositions de routage dans les réseaux d'actionneurs : MobileR, PAMAL et GRR.

À l'origine de notre première proposition, MobileR, pour *Mobile Recursivity*, vient un constat : les protocoles de routage pour actionneurs existants adaptent la topologie du réseau à son trafic, mais ils ignorent totalement leur impact sur le processus de routage. En repositionnant les nœuds sur les chemins de routage, ils adaptent certes la topologie, mais ils modifient aussi les possibilités de routage de chacun de ces nœuds. MobileR, lui, va avoir une approche différente. Il va, à chaque étape, essayer d'anticiper au maximum le routage sur plusieurs sauts pour effectuer le meilleur choix possible. Il va calculer la relocalisation des voisins N et pour chacun d'entre eux les multiples $N+1$ possibles, etc. MobileR calcule donc en fait les chemins de routage possibles vers la destination sur le voisinage du nœud courant. Des simulations viennent comparer les résultats de MobileR à la littérature. Il s'en dégage que l'anticipation de MobileR permet des économies d'énergie non négligeables.

PAMAL, pour *PAth Merging ALgorithm*, traite une nouvelle problématique de la mobilité quand plusieurs sources du réseau émettent des paquets vers la même destination. L'adaptation locale de la topologie modifie alors plusieurs chemins indépendamment. Or, plus on approche de la destination, plus la probabilité que ces chemins partagent des nœuds est élevée. Ces nœuds vont alors sans cesse être déplacés sur chaque chemin et donc mourir prématurément. La mobilité amène donc de nouveaux problèmes. PAMAL va détecter ces intersections de chemins. Il va s'en servir afin de faire fusionner physiquement les chemins de routage en amont des points d'intersections. Il va aussi provoquer une agrégation de paquets sur la partie commune des chemins de routage afin d'économiser de l'énergie.

Enfin, le protocole GRR, pour *Greedy Routing Recovery*, propose lui une nouvelle approche pour résoudre un problème intrinsèque aux approches gloutonnes dans les réseaux de capteurs : le problème de l'*extremum* local. Notre protocole va tirer parti de la mobilité contrôlée afin de modifier la topologie en périphérie de l'obstacle et permettre de le contourner. L'obstacle va ainsi être circonvenu par GRR en relocalisant des nœuds tout autour en utilisant un principe de relocalisation des nœuds qui permet aussi de prolonger le routage glouton dans le contournement.

Ce travail a été réalisé dans le cadre de notre thèse effectuée au sein du laboratoire LIFL et du centre de recherche Inria Lille – Nord Europe dans l'équipe FUN, sous la direction du Pr David Simplot-Ryl et du Dr Nathalie Mitton.

1.2 Organisation de ce document

Dans ce premier chapitre nous introduisons d'abord le contexte du sujet d'étude de notre thèse. Un résumé des contributions détaille ensuite nos travaux. Puis, nous décrivons les notations et modèles qui sont utilisés tout au long du document et dans les simulations.

L'état de l'art est détaillé dans le chapitre 2. MobileR est décrit dans le chapitre 3. Le chapitre 4 est consacré au protocole PAMAL, tandis que le chapitre 5 est dédié à GRR. Enfin, le chapitre 6 conclut nos travaux et vient présenter plusieurs perspectives de recherches sur l'ensemble des propositions que nous avons pu faire et sur les réseaux de capteurs à mobilité contrôlée en général.

1.3 Modèles et notations utilisés

Un réseau sans fil hybride capteur/actionneur peut être modélisé par un graphe noté G tel que G=(V,E) où V représente l'ensemble des capteurs et actionneurs, aussi appelés nœuds, et E l'ensemble des liens radio existants. Nous notons (A, B) l'existence d'un lien radio bidirectionnel entre les nœuds A et B. Ainsi (A,B) équivaut à (B,A) : $\forall A, B \in G$: (A,B) $\Leftrightarrow$ (B,A). Nous notons par $N(A)$ l'ensemble des liens radio de A avec les autres nœuds du réseau. Ce voisinage à un saut est l'ensemble des nœuds avec lesquels A peut établir une communication directe, sans intermédiaire. $N(A) = \{B|(A, B) \in E\}$.

Nous notons $(S\ D)$ la droite passant par les nœuds S et D, le chemin de routage entre ces deux nœuds étant noté $S \rightarrow \cdots \rightarrow D$. La distance euclidienne entre S et D est notée |SD|.

Nous notons $N_D(A)$ le sous-ensemble des voisins de $N(A)$ qui sont plus proches de la destination que lui même, *i.e.* $N_D(A) = \{B \in N(A) \,|\, |BD| < |AD|\}$.

Nous appellerons capteur un nœud statique, alors que le terme actionneur désigne un nœud dont la mobilité est contrôlée.

Nous utilisons $\delta(A)$ pour noter le degré du nœud A dans le réseau. Le degré d'un nœud dans le réseau est le nombre de liens directs qu'un nœud du réseau possède avec d'autres nœuds du réseau. Soit $\delta(A) = |N(A)|$. La notation A indique le nœud A mais aussi sa position selon le contexte d'utilisation. La nouvelle position d'un nœud A est noté $A\prime$.

Nous appelons nœud courant un nœud qui a un message à router.

Nous notons C(A, r) le cercle centré sur le nœud A et de rayon *rayon*.

Nous supposons que tous les nœuds sont capables d'obtenir leur position géographique et qu'ils sont capables de moduler leur puissance d'émission afin de faire varier leur portée d'émission de 0 à R (R>0).

1.3.1 Modèles de coût

1.3.1.1 Coût radio

Les différents protocoles que nous avons proposés sont indépendants du modèle de calcul des coûts de transmission radio. Néanmoins, pour pouvoir se comparer à la littérature, nous utilisons le modèle de coût des transmissions radio défini dans [RM99] :

$$C_{radio}(r) = C + r^{\alpha} \tag{1.1}$$

Dans cette équation :
- la variable r représente la distance euclidienne parcourue par la transmission radio,
- C est une constante correspondant au coût de traitement du signal par l'électronique,
- α est une constante réelle ($1 < \alpha < 8$) qui représente l'atténuation du signal dans le milieu de propagation.

1.3.1.2 Coût de déplacement

Le modèle de coût employé pour les déplacements est issu de [LNS07] :

$$C_{mvt}(r) = a \times r \tag{1.2}$$

Où :
- r est la distance euclidienne à parcourir par l'actionneur,
- a est une constante dépendant du matériel utilisé.

1.4 Disque unitaire

Dans le modèle du disque unitaire, ou *Unit Disk Graph*, deux nœuds peuvent communiquer si, et seulement si, la distance les séparant est au plus égale à R, où R est le rayon de transmission, égal pour chaque nœud. Le disque unitaire est donc défini par la position des nœuds et leur rayon de communication commun R.

Le modèle du disque unitaire est utilisé pour modéliser les réseaux de capteurs sans fil, les réseaux *ad-hoc*, les réseaux véhiculaires mais aussi les réseaux d'actionneurs. Cependant, et même si de nombreux protocoles réseaux de la couche routage sont conçus sur les hypothèses du disque unitaire, le modèle du disque unitaire ne peut être considéré comme réaliste car les variations de la puissance du signal radio reçu n'y sont pas représentées.

Chapitre 2

État de l'art

Selon le principe de base du routage géographique, quand un nœud source émet un paquet, il inclut sa destination dans ses méta-données. Le paquet est alors retransmis de nœud intermédiaire en nœud intermédiaire jusqu'à la destination. On suppose que la position de la destination est connue. Soit parce que dans le paquet, soit parce que tous les puits sont connus a priori, soit parce qu'il existe un mécanisme de localisation des nœuds en parallèle. Ainsi, quand un nœud intermédiaire reçoit le paquet, il peut le transmettre à un de ses voisins dans la direction de la destination. Et, de saut en saut, de retransmission en retransmission, le paquet atteint – idéalement– sa destination. La stratégie du routage est importante, c'est elle qui détermine quel nœud, parmi ceux dans la direction de la destination, sera sélectionné comme prochain nœud pour retransmettre.

Si la stratégie est essentielle, la notion de la position géographique l'est tout autant. Ainsi, chaque nœud du réseau doit connaître sa propre position géographique. Pour ce faire, il fait appel à système de localisation voire un composant matériel. Il peut s'agir d'une puce de type GPS [KH05] ou de tout autre mécanisme [CHH01]. Il existe de multiples techniques de positionnement absolu ou relatif, comme par exemple en mesurant la puissance des signaux reçus [CHH01]. Une revue des différents mécanismes peut être trouvée dans [NN01].

En plus de la sienne, chaque nœud possède la connaissance de ses voisins immédiats avec qui il peut communiquer sans intermédiaire (voisinage à 1 saut). C'est en se basant sur sa position propre, celle de ses voisins et la position de la destination que les choix de routage sont faits.

On le voit, les protocoles de routage géographique glouton ont plusieurs avantages intrinsèques :

— un coût mémoire relativement faible : il ne s'agit plus de mémoriser des tables de routage, mais plutôt de maintenir à jour des tables de voisinages, indépendantes du routage,
— la localité : ces protocoles restent locaux car les choix de routage se basent sur les informations que détient le nœud courant,

— le passage à l'échelle est idéal : ces protocoles se basant sur des informations locales, ils peuvent fonctionner sur des réseaux de grande échelle.

Dans ce chapitre, nous dressons un état de l'art des principaux protocoles de routage géographique glouton pour les réseaux statiques. Nous présentons ensuite les protocoles de routage de la littérature reposant sur l'hypothèse de l'utilisation d'un ou plusieurs actionneurs, c'est-à-dire de nœuds possédant la faculté de se déplacer par eux-mêmes. Cette capacité ouvre de nouvelles possibilités permettant d'envisager d'adapter la topologie du réseau pour un routage plus efficace.

2.1 Routage géographique glouton

Le premier protocole de routage à tirer parti de la position géographique est LAR [KV00], pour *Location Aided Routing*. LAR est un protocole de routage réactif. C'est-à-dire que les chemins de routage entre la source et la destination sont établis si, et seulement si, il existe un message à router. LAR est très similaire au protocole DSR [BMJ+98] qu'il optimise en utilisant la géographie. LAR estime ainsi une zone circulaire dans laquelle il y a une forte probabilité que la destination du message se trouve. La position et la taille de la région sont estimées en se basant sur les informations suivantes à propos de la destination :

— sa position à un instant donné dans le passé,
— la vitesse moyenne de déplacement et la direction associées à cet instant.

LAR provoque alors une inondation de requête de route dans cette zone géographique, là où DSR ne limiterait pas l'inondation. LAR a donc pour avantage de limiter l'inondation – très consommatrice d'énergie – à seulement une partie du réseau. On le voit, LAR n'utilise donc pas un routage géographique à proprement parler. L'information géographique n'y est pas utilisée pour choisir le prochain saut, mais pour limiter la consommation énergétique et permettre d'améliorer la durée de vie du réseau. Toutefois, il reste le précurseur dans l'utilisation de l'information géographique.

L'utilisation de la position géographique pour le routage proprement dit vient réellement, pour la première fois, avec le protocole MFR [TK84], pour *Most Forward within Radius*. Dans MFR, le nœud courant E transmet le paquet à son voisin qui permet de maximiser le progrès. Le progrès est d'autant plus important que la distance entre la projection orthogonale d'un voisin sur (ED) et la destination est petite. Les auteurs de [Fin87] font simplement sélectionner par le nœud courant son voisin le plus proche de la destination comme prochain nœud.

Ces deux algorithmes, en plus d'être statiques, visent donc à minimiser le nombre total de sauts en privilégiant des sauts les plus longs possible. Or, selon le modèle pour les coûts radio défini par l'équation1.1, la consommation énergétique due aux radio transmissions augmente exponentiellement avec la distance. Ces algorithmes sont donc inadaptés à l'hypothèse de réseaux sans fil avec des nœuds fonctionnant sur une batterie.

À l'inverse, le protocole NFP [HL86] sélectionne comme prochain saut le voisin le plus près géographiquement du nœud courant. NFP favorise donc les sauts les plus petits possible. Cette approche, qui suppose que les nœuds possèdent une portée radio ajustable, permet de réduire les interférences radio [MWH01], et donc de limiter la surconsommation due aux retransmissions.

Encore une fois, en se basant sur l'équation 1.1, ce principe de multiplier les sauts courts est en contradiction avec l'efficacité énergétique. En effet, le médium radio présente un coût d'activation non négligeable, cf l'équation 1.1. Cet algorithme va donc être, lui aussi, très consommateur en énergie.

Le protocole RPM [NK84], pour *Random Progress Method*, utilise lui aussi l'hypothèse d'une portée radio ajustable. Le nœud courant y sélectionne comme prochain saut un de ses voisins qui permet de réduire la distance euclidienne jusqu'à la destination, aléatoirement. La portée radio – supposée être de rayon uniforme – est alors ajustée pour ne pas communiquer avec un autre voisin plus loin que lui. Ainsi on minimise les interférences et la consommation en limitant la puissance d'émission.

Si une approche aléatoire peut être une approche justifiable dans les réseaux statiques, car en changeant de relais aléatoirement on répartit les coûts du routage sur un plus grand nombre de nœuds, elle l'est beaucoup moins dans des réseaux avec mobilité contrôlée. On vise en effet à adapter la topologie du réseau à son trafic. Or, si l'on change le chemin de routage aléatoirement à chaque saut, il est impossible d'adapter la topologie de façon efficace.

Dans DIR [KSU99], pour *DIRectional Routing*, l'approche est différente et repose sur un calcul d'angle. DIR sélectionne comme prochain saut le voisin du nœud courant qui va minimiser l'angle de progression vers la destination. Le nœud courant E va sélectionner comme prochain saut le nœud G appartenant à son voisinage qui minimise l'angle $\angle EGD$. On le voit, DIR peut provoquer des boucles [SL01a] dans le routage, car il ne se limite pas à choisir dans ses voisins plus proches de la destination que lui, comme par exemple MFR. C'est donc un principe de routage non utilisable en l'état car le moindre routage risque de mettre en danger l'intégrité du réseau en y « tournant » tant que les nœuds ont assez de batterie.

Le protocole COP [KNS06], pour *Cost-Over-Progress*, adopte comme métrique de sélection le ratio du coût sur progrès qu'il cherche à minimiser. Le nœud courant va donc calculer le coût de transmission à chaque voisin, par rapport au gain (réduction de la distance euclidienne jusqu'à la destination) qu'il permet. C'est le nœud qui minimise ce coût sur progrès auquel est transmis le paquet.
Ce principe de sélection, qui prend en compte le coût, est transposable aux protocoles de routage aux réseaux mobiles si l'on inclut les coûts de déplacement en plus des coûts de transmission. Le progrès peut aussi être augmenté en considérant le déplacement des nœuds.

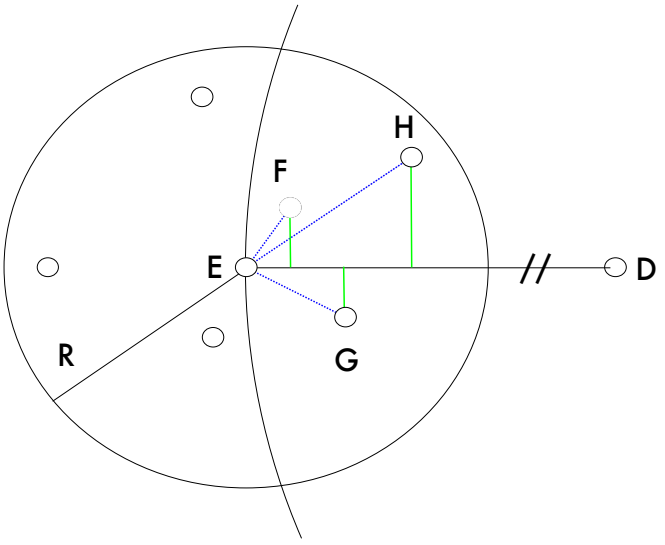


FIGURE 2.1 – Le nœud E a un message pour D. MFR sélectionne H, NFP le nœud E, DIR G

2.2 Routage avec mobilité

Une autre approche dans le routage apparue récemment dans les réseaux de capteurs sans fil consiste en l'introduction d'actionneurs. Les actionneurs sont dans notre cas des capteurs sans fil capable d'agir sur leur environnement. Les actionneurs employés sont ainsi capables de communiquer sans fil mais aussi de se déplacer de façon autonome, sur ordre ou par décision propre.

Une étude [WSC08] a démontré que l'introduction d'actionneurs dans les réseaux de capteurs permet de meilleures performances énergétiques que l'approche traditionnelle qui consiste à multiplier les capteurs. En effet, si l'augmentation de la densité en capteurs permet, avec un protocole de routage adapté, de réduire la longueur – et donc le coût – des transmissions radio, elle provoque une multiplication des interférences radio. L'augmentation de densité a ses limites.

À notre connaissance, c'est dans [LR00] qu'on trouve la première utilisation de la mobilité contrôlée dans un contexte de routage.
Les auteurs y considèrent le routage dans un réseau composé de nœuds mobiles, chaque nœud suivant une trajectoire indépendante mais pré-définie. On peut donc calculer avec précision la position de chaque nœud du réseau à n'importe quel instant.

Cette prédictibilité est exploitée dans un algorithme dénommé Optimal Routing Path. Cet algorithme va faire calculer un *chemin de routage en mouvement* jusqu'à la destination par le nœud courant, chemin qui repose sur les nœuds en mouvement du réseau.

Optimal Routing Path diffère des algorithmes existants car il prend en compte la

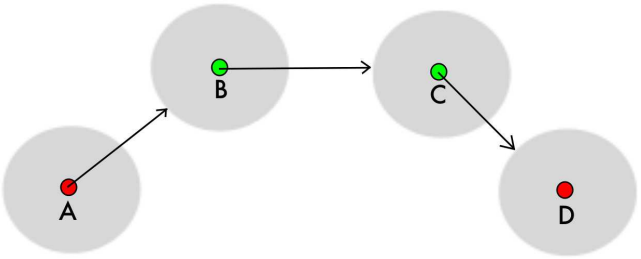


FIGURE 2.2 – Le nœud A a un message pour D. A modifie sa trajectoire, et celles de B et C pour router le message.

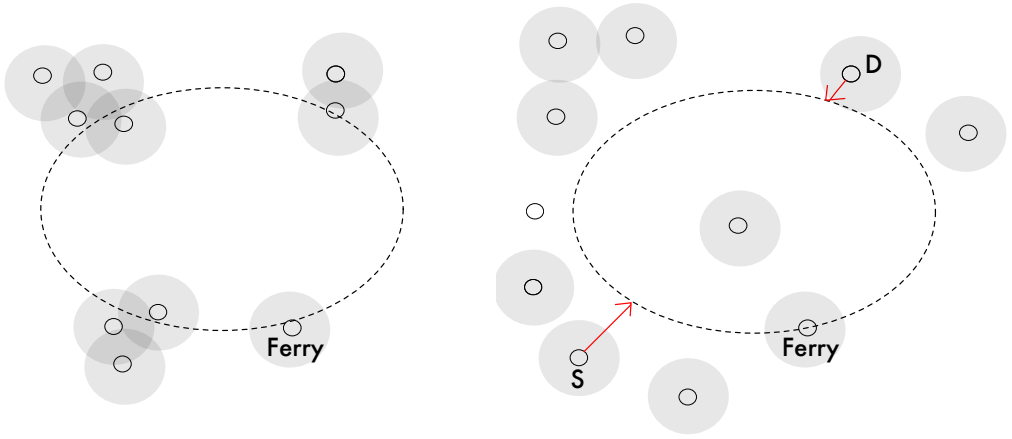
possibilité de faire dévier légèrement et temporairement chaque nœud de sa trajectoire. Ce changement de trajectoire vise à rendre possibles des transmissions radio qui ne pourraient l'être autrement à cause d'un rayon de transmission trop faible, etc. En utilisant la connaissance globale des mouvements des nœuds et la possibilité de modifier les trajectoires, Optimal Routing Path retourne le chemin le plus court en termes de temps pour router un message entre deux nœuds mobiles.

La figure 2.2 représente un cas d'utilisation d'Optimal Routing Path. Le réseau est composé de quatre nœuds mobiles A, B, C et D. A a un message à transmettre au nœud D qui est, et restera, hors de portée de communication radio de A. Une solution pourrait être de faire se déplacer A à portée de communication de D. Mais cette approche serait très énergivore, A devant parcourir une longue distance mais aussi retourner à sa trajectoire d'origine. Dès lors, considérant que A peut connaître par le calcul la position de chacun des nœuds du réseau, A calcule *le chemin de routage en mouvement* de A à D en utilisant des nœuds intermédiaires. Le résultat de l'algorithme est que A dévie de sa trajectoire pour se rapprocher de B et lui transmettre le message. B agit de la même façon vers C qui se dévie jusqu'à être à portée de D pour lui transmettre directement le message.

Toutefois, les hypothèses d'omniscience sur lesquelles repose Optimal Routing Path rendent cette approche difficilement transposable à l'univers des réseaux de capteurs possédant des contraintes en termes de puissance de calcul et de mémoire. Cependant, l'idée de contrôler la mobilité était lancée. Un nouveau paradigme de routage a alors émergé appelé le « message ferrying ». Ce paradigme limite la complexité du problème en restreignant le besoin de la connaissance du mouvement à seulement quelques nœuds.

2.2.1 Le message ferrying

Le *message ferrying* est un paradigme de routage dans les réseaux sans fil dans lesquels il existe au moins un nœud mobile. Ce nœud parcourt le réseau en suivant une trajectoire prédéfinie, connue par l'ensemble des nœuds, ainsi que sa position de départ. Dès lors, sa position est prédictible via calcul par tous les nœuds du réseau à chaque instant. Le *ferry* peut donc recevoir des messages sur son trajet, les stocker, parcourir le réseau et finalement les ré-émettre. C'est une modification importante du fonctionnement du routage. On passe du *recevoir, stocker, ré-émettre* à recevoir, stocker, déplacer, ré-émettre. Ce paradigme implique de distinguer au moins deux entités réseaux logiques différentes : le *ferry*, et les autres nœuds du réseau. Par contre, leurs caractéristiques matérielles peuvent être identiques. La différence est d'ordre fonctionnel : la navette n'a que le routage comme activité, tandis que les autres nœuds peuvent avoir plusieurs activités, être statiques ou mobiles... Ce paradigme, bien que similaire, est différent du routage opportuniste. Le mouvement du *ferry* est proactif et non aléatoire : la navette va suivre sa trajectoire et router les paquets quelle que soit l'activité du réseau. Il n' y a pas d'aléa, en théorie, dans le message *ferrying*.



(a) Réseau statique (b) Réseau mobile

FIGURE 2.3 – Les ferries en action transportant physiquement les messages dans leur mémoire avant ré-émission.

Le message ferrying est utilisé pour la première fois dans [ZA03]. Leur contexte d'utilisation est alors d'améliorer la connectivité d'un réseau, voire d'instaurer la connexité dans un réseau partitionné comme l'illustre la figure 2.3a. Le calcul des trajectoires des ferries est alors de première importance. Il doit minimiser la latence introduite par le déplacement physique du *ferry*. En effet, les nœuds du réseau doivent avoir assez de mémoire pour stocker temporairement les messages pour le *ferry* durant l'intervalle de temps défini par deux passages. Autrement les nœuds devront écraser (et donc perdre) des données. De même, la capacité mémoire du *ferry* n'est pas infinie. Le problème du calcul de ces trajectoires est NP-Complet et seulement une solution approximative peut

être donnée [ZA03].

Dans [ZA03] les auteurs assimilent le calcul de la trajectoire du *ferry* au problème du voyageur de commerce (Travelling Salesman Problem). L'algorithme proposé calcule alors une approximation du chemin optimal faisant passer le *ferry* à portée de tous les nœuds du réseau. Le trajet est adapté dans un second temps pour satisfaire à des besoins de bande passante. Cette approche fait donc parcourir le réseau au *ferry* d'un point de rendez-vous à l'autre. Le *ferry* transporte donc physiquement les messages d'un point à un autre du réseau avant de les retransmettre.

Dans les réseaux où tous les nœuds sont mobiles, le *ferry* peut être utilisé afin d'éviter les déplacements trop importants. Cette possibilité est étudiée dans [ZAZ04]. Les auteurs y distinguent deux cas : le *Node Initiated Message Ferry Scheme* (NIMFS), illustré dans la figure 2.3b, et le *Ferry Initiated Message Ferry Scheme* (FIMFS).

Dans le NIMFS, si un nœud S a un message pour D, S va aller à la rencontre du *ferry* pour lui transférer directement le message. Le *ferry* peut alors transporter le message jusqu'à un autre endroit, sans jamais avoir eu à modifier sa trajectoire. Le message est finalement retransmis à D directement quand il est à portée du *ferry*.

Dans le FIMFS, quand un nœud a un message à envoyer, il prévient le *ferry* qui va le rejoindre. En d'autres termes, c'est le *ferry* qui modifie sa trajectoire volontairement de façon à réceptionner le message via une communication de courte portée. Quand le paquet est récupéré, la route est recalculée de façon à délivrer le message. Deux heuristiques peuvent être utilisées, soit il se déplace vers le voisin le plus proche, ou alors il réutilise le voyageur de commerce adapté pour minimiser la perte de message.

De nombreuses approches ont par la suite été proposées pour optimiser le calcul de la route du *ferry*. Une avancée majeure présentée dans [ZAZ05] a consisté à multiplier les *ferries* au lieu d'en utiliser un seul. La difficulté de calculer de calculer la route étant déjà clairement établie comme NP-difficile, nous pouvons en déduire que le calcul de plusieurs routes l'est lui aussi.

Ce n'était plus qu'une question de temps avant qu'on étende l'hypothèse de mobilité contrôlée pour le routage à tous les nœuds du réseau.

2.2.2 Routage dans les réseaux d'actionneurs

Le premier protocole considérant un réseau formé pour toute ou grande partie d'actionneurs est [GLM+04]. Son approche est celle d'un routage en deux temps. Le chemin de routage est d'abord construit via une requête de route (*route request*, RQ) transmise de manière gloutonne dans le réseau. Quand la requête arrive à destination, une réponse (*route reply*, RP) est renvoyée sur le même chemin, en sens inverse. Les nœuds sur le chemin sont alors déplacés sur la droite virtuelle (SD) entre la source et la destination. Précisément les nœuds sont relocalisés sur l'isobarycentre entre le nœud prédécesseur et le nœud successeur du chemin de routage. Non seulement ce protocole passe difficilement à l'échelle à cause des RR/RQ, mais de plus son schéma déplacement risque de provoquer des mouvements de type zig-zag inutiles et énergivores.

Ce problème est traité dans [CLN09]. Les auteurs y proposent de n'effectuer le mouvement que virtuellement tant que la stabilité n'est pas atteinte. Les nœuds sur le chemin de routage ne bougent donc pas tant que l'algorithme de routage fait évoluer leur position. Cette approche suppose toutefois que les chemins de routage sont déjà fixés.

2.2.2.1 MobileCOP

Dans MobileCOP[LNS07], on retrouve le principe de sélection gloutonne du prochain nœud selon une approche du coût sur progrès [KNS06]. MobileCOP propose de sélectionner les nœuds de façon à en avoir un nombre optimum sur le chemin de routage, ou de minimiser le coût de la consommation énergétique par rapport au progrès. Une fois que les nœuds de routage ont été sélectionnés lors de la RQ, ils sont déplacés lors de l'étape RR. Leur nouvelle position est calculée par la destination de manière à les positionner de façon équidistante sur (SD). Cette approche est très consommatrice en mémoire, car les nœuds doivent mémoriser le chemin de routage, et est très longue à se mettre en place, ce qui introduit une latence non contrôlée. Qui plus est, les mouvements sont faits de façon à adapter une route précise, sans aucune considération pour le réseau dans sa globalité. L'approche de coût sur progrès n'est aussi que partielle car l'algorithme ne considère pas du tout le coût énergétique du déplacement des nœuds.

2.2.2.2 RPCM

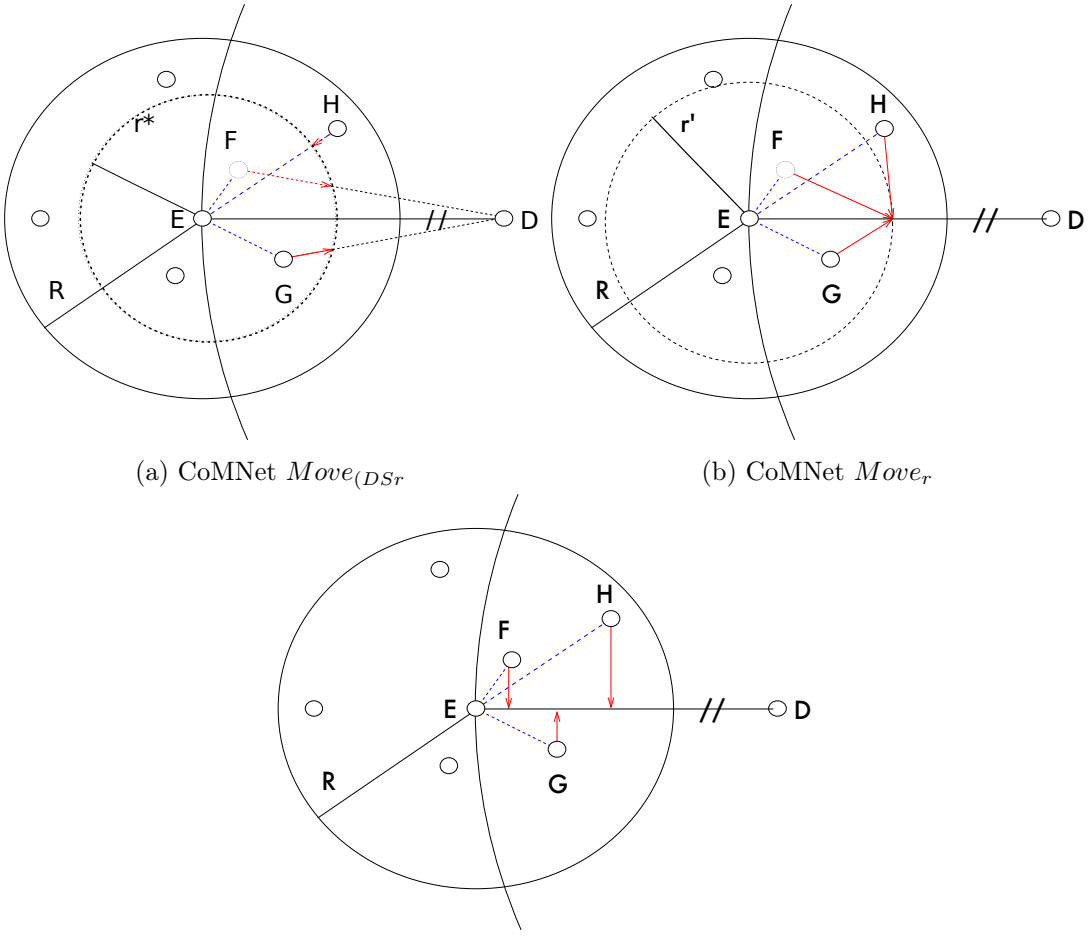
Le protocole RPCM [LNC10], pour *Routing Protocol Based on Mobility*, est aussi un protocole avec RQ/RP. Il repose même sur l'envoi simultané de multiples RR vers la destination : chaque voisin du nœud source va en recevoir une. Chaque voisin va ajouter ses informations (identifiant et position) dans un champ du paquet et le retransmettre. La destination va donc recevoir de multiples RQ contenant chacune un chemin de routage possible. Après un temps prédéfini, la destination sélectionne le chemin qui minimise la quantité de mouvement totale des nœuds qui en sont membres. Dans RPCM, les nœuds du chemin de routage sont relocalisés de façon équidistante sur la droite (SD). La RP contient donc le chemin retenu et les nœuds, en la retransmettant vers la source, se déplacent. Cette approche peut difficilement passer à l'échelle : les auteurs ne précisent pas comment le protocole se comporte si la RP est trop petite en terme de capacité pour contenir l'intégralité du chemin. Il faut en effet, pour chaque nœud sur le chemin de routage, retenir au minimum identifiant et nouvelles positions. De plus, l'impact de la mort d'un nœud sur ce chemin nécessite un nouvel appel au processus RP/RQ entre la source et la destination, et non une réparation locale.

2.2.2.3 CoMNet

Une autre approche plus récente est celle proposée par CoMNet [HMSR11], pour *Connectivity preservation Mobile routing protocol for actuator and sensor NETworks*. CoMNet exploite aussi la mobilité contrôlée pour adapter la topologie au trafic. Mais,

contrairement aux précédentes propositions, les chemins de routage y sont construits à la volée. Il n'y a pas de mécanisme de RR/RP dans ce protocole : CoMNet est un protocole sans mémoire, il ne nécessite pas de mémoriser des informations sur le chemin de routage que ce soit dans les nœuds ou dans les paquets routés. Qui plus est, CoMNet est le premier protocole de routage reposant sur la mobilité contrôlée qui :

— prend en compte à la fois les coûts de radio transmission et de déplacement dans une approche coût sur progrès,
— est indépendant des modèles de coût,
— garantit le maintien de la connexité du réseau à tout instant, malgré les déplacements,
— propose trois schémas de déplacement différents pour s'adapter au mieux aux différentes conditions environnementales.



(a) CoMNet $Move_{(DSr}$

(b) CoMNet $Move_r$

(c) CoMNet Orouting On the Move

FIGURE 2.4 – Les différents schémas de déplacements de CoMNet.

Ces différents schémas visent à adapter au mieux les quantités de mouvements et les distances de transmission radio selon l'application et l'importance relatives des coûts :

- $CoMNet - Move_{(DSr)}$ aligne les nœuds sur la droite virtuelle (SD) entre la source S et la destination D du message. En plus d'y être alignés, les différents nœuds sur le chemin de routage sont placés de façon équidistante. Ils sont espacés d'une distance r^*, la distance optimale de radio-transmission. On le voit, ce schéma vise à modérer à la fois les déplacements et les transmissions radio.
- $CoMNet - ORouting\ on\ the\ Move$ aligne lui aussi les nœuds du chemin de routage sur la droite (SD). La nouvelle position d'un nœud correspond ici à sa projection orthogonale sur la droite (SD). Ce schéma vise à minimiser les coûts de transmission radio. Ainsi si un nœud E sélectionne F comme prochain nœud sur le chemin de routage, il envoie à F un ordre de déplacement via un *beacon* avec sa nouvelle position. Quand F a atteint sa nouvelle position F', E lui transmet alors le paquet. Ce mécanisme permet de réduire les communications radio en ne transmettant les paquets que sur des distances courtes. Des distances qui sont plus courtes qu'elles n'auraient pu l'être sans la mobilité contrôlée.
- $CoMNet - Move_r$ provoque la relocalisation des nœuds à l'intersection du cercle centré sur le nœud courant et de rayon r', et la droite (ED). Ici, dans ce shéma de déplacement, l'objectif est de limiter le déplacement des nœuds.

La figure 2.4 illustre le déplacement (flèche rouge) que chaque schéma applique sur les voisins du nœuds courant. Ainsi, quand le nœud courrant E a un paquet à router, il calcule pour chacun des ses voisins X plus proche de la destination que lui sa nouvelle position X' selon le schéma utilisé. E sélectionne comme prochain nœud son voisin qui minimise le ratio du coût sur progrès. Le progrès correspond ici à la réduction de la distance euclidienne vers la destination que permet le voisin relocalisé : $|ED| - |X'D|$ où X' est la position du voisin déplacé. Les résultats montrent que CoMNet, parce qu'il tient compte du coût de déplacement des nœuds et qu'il propose plusieurs schémas de déplacement, présente des résultats bien supérieurs à ceux de MobileCOP.

CoMNet innove aussi car c'est le premier protocole de routage utilisant la mobilité qui tient compte de la connexité du réseau. Ainsi, l'usage de la mobilité fait par CoMNet ne risque pas de déconnecter le réseau en plusieurs sous-réseaux indépendants. Il utilise pour cela un ensemble dominant connecté ou CDS, pour *Connected Dominated Set*. Un CDS est un super ensemble connecté de nœuds qui couvre la même surface que le réseau initial. En décidant que les nœuds membres du CDS (ils sont dits *dominants*) sont statiques, CoMNet garantit que le déplacement des autres nœuds du réseau (les nœuds *dominés)* n'affecte pas la connexité du réseau tant qu'ils restent à portée de communication d'un nœud dominant. Ainsi on garantit qu'à chaque instant il existe au moins un chemin entre deux nœuds du réseau, via le CDS comme l'illustre la figure 2.5. Le CDS peut être calculé grâce à l'algorithme local proposé dans [CSR04]. Ainsi, un nœud A est dominant si l'une des conditions suivantes n'est pas respectée :

- si Z, le sous-ensemble des nœuds de $N(A)$ avec une priorité plus haute que A, n'est pas vide $Z \neq \{\emptyset\}$

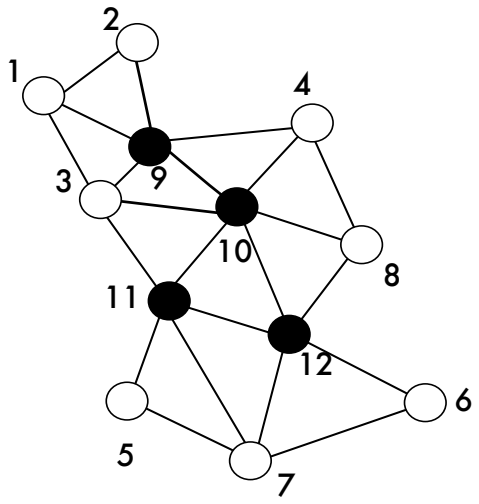


FIGURE 2.5 – CDS : les nœuds dominants sont en noir et sont statiques.

— si Z n'est pas connecté : $\forall A \in Z, \exists B \in Z \ t.q. \ A \neq B \wedge |AB| \ < portee\_radio$

— s'il existe un nœud dans $N(A)$ qui ne soit pas à portée de communication d'un nœud Z : $\forall B \in N(A), B \in Z \cup N(S)$.

La priorité d'un nœud est un critère choisi arbitrairement qui peut être leur identifiant, leur niveau de batterie...

2.3 Conclusion

La littérature concernant les protocoles de routage géographique pour les réseaux de capteurs est très vaste. Cependant, la quasi-totalité des approches proposées ignore complètement la mobilité, qu'elle soit contrôlée ou non, car elle les met bien souvent en échec. Il est très récent qu'elle soit considérée comme une primitive du routage à exploiter et non plus un aspect négatif. Toutefois, il a fallu du temps pour envisager un réseau dans lequel tous les nœuds du routage sont à la fois mobiles et contrôlables. Là encore la mobilité n'est considérée que sous un aspect particulier. Par exemple, les routages basés sur des RQ/RP ignorent totalement le risque de déconnecter le réseau par leur mobilité. À l'inverse, CoMNet, lui, la garantit, mais son approche très locale, idéale pour les grands réseaux, provoque des changements de topologie qui ont des conséquences sur les possibilités de routage dans les futurs chemins. C'est pourquoi, dans notre travail, nous avons proposé un nouvel algorithme de routage local, basé sur les principes de CoMNet, qui va, à chaque étape du routage, anticiper les conséquences possibles sur le routage des déplacements qu'il provoque pour optimiser le choix du prochain saut.

Chapitre 3

Routage récursif avec mobilité

3.1 Introduction

L'introduction des actionneurs représente une évolution majeure pour les réseaux de capteurs sans fil. On pourrait même se risquer à parler de révolution comme on l'a fait en leur temps pour les communications sans fil ou l'introduction de la position géographique. En effet, cette possibilité d'agir sur l'environnement – et non plus seulement d'en être tributaire – ouvre de nouvelles opportunités. Elle permet d'envisager des réseaux de capteurs dont la topologie même évolue dynamiquement, mais de *façon contrôlée*. Cette évolution contrôlée peut se faire pour adapter au mieux la disposition des nœuds du réseau à l'environnement qu'il surveille, à l'application, ou plus simplement à l'activité réseau.

L'objectif de ces travaux est de tirer parti de cette possibilité de mobilité contrôlée afin de proposer une nouvelle famille de protocoles de routage géographique dans les réseaux sans fil de capteurs et d'actionneurs. Or le mouvement des nœuds dans les réseaux de capteurs sans fil a longtemps été perçue comme source de problèmes pour le routage géographique. Dans ce type de protocoles de routage, où la position géographique est une donnée essentielle pour le bon acheminement des données, les mouvements sont en effet souvent la cause de perte de paquets, et donc d'échec de routage. De fait, l'efficacité de la majorité des algorithmes de routage utilisés dépend en fait des caractéristiques inhérentes à la topologie même du réseau. Ainsi les positions absolues ou relatives des différents nœuds, le nombre de liens par nœud ou leur longueur sont des paramètres critiques pour bons nombres de protocoles.

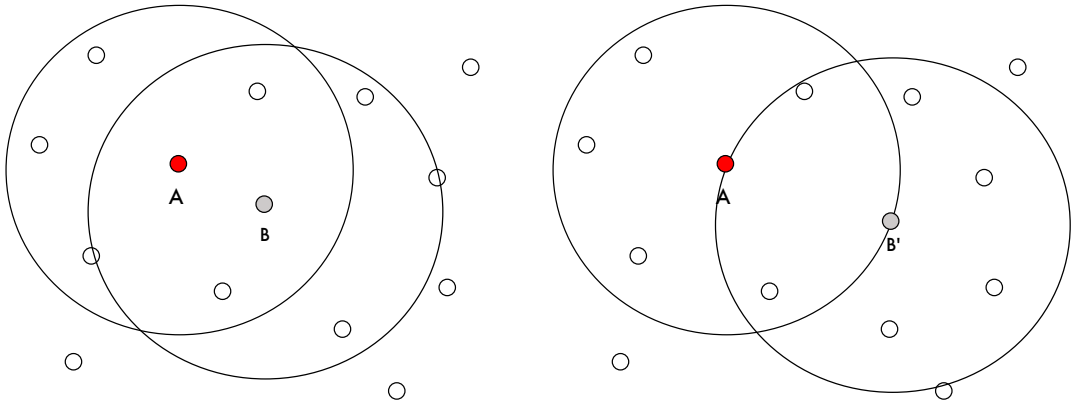
C'est pourquoi les réseaux d'actionneurs requièrent de nouveaux protocoles. Ceux-ci doivent être capables d'adapter la topologie du réseau tout en minimisant les paquets nécessaires à cette adaptation. La construction des chemins de routage doit utiliser au mieux cette mobilité locale afin d'optimiser la topologie globale. Les chemins doivent être stables tout en s'adaptant facilement aux changements. De plus, et comme mentionné dans le Chapitre 2, il s'agit de garantir la connexité du réseau en dépit de la mobilité. Le mouvement individuel d'un nœud du réseau, même minime, peut ainsi provoquer une partition physique du réseau en plusieurs sous-réseaux. Un autre aspect pas encore

abordé jusque-là est que le fait de déplacer un nœud sur un chemin de routage va aussi modifier les possibilités de routage que ce nœud aura, quand ce sera à lui de retransmettre le paquet. Le fait de déplacer un nœud modifie son voisinage. Il apparaît donc nécessaire de prendre en compte cette conséquence de la mobilité.

C'est en se basant sur cette constatation que nous avons proposé le protocole de routage MobileR pour *Mobile Recursivity*. MobileR est un protocole de routage basé sur CoMNet [HMSR11], qui va tirer parti de la mobilité pour construire des chemins de routage à la topologie optimisée. À la différence des autres protocoles existants, MobileR va anticiper au maximum ses choix de routage et de déplacement des nœuds sur plusieurs sauts tout en restant purement local. Ainsi, ses choix de routage et de déplacements permettront un routage plus performant.

3.2 Contribution

Les protocoles de routage géographique décentralisés classiques dans les réseaux de capteurs font, à chaque étape du routage, un choix : celui du prochain nœud sur le chemin de routage entre le nœud source S et la destination D. L'algorithme et les paramètres de sélection du prochain nœud sont extrêmement variables d un protocole à l'autre, mais le principe reste le même, de saut en saut, du nœud émetteur du message jusqu'à sa destination. Dès lors, on pourrait considérer – à tort – qu'il suffit pour adapter la topologie de demander à ce prochain nœud de se déplacer.



(a) Voisinage de B (b) Voisinage de B'

FIGURE 3.1 – Influence de la mobilité sur le voisinage

La figure 3.1a illustre une étape dans un réseau statique habituel. Le nœud courant A y sélectionne parmi les voisins à sa portée le prochain saut, ici B. Si l'on se place dans le cadre d'un réseau d'actionneurs A peut tout à fait sélectionner le même nœud, B, et lui demander de se déplacer en – par exemple – B' comme illustré dans la figure 3.1b. Or ce déplacement n'est pas sans conséquence. En effet, la nouvelle position B' implique une modification notable du voisinage de B. Des voisins précédemment

joignables sont désormais hors de portée de communication, alors que d'autres, précédemment inconnus, deviennent joignables. Le déplacement des nœuds entraîne donc à la fois la suppression et la création de chemins de routage possibles. Il s'agit d'en tenir compte lors des choix de routage. Une approche globale permettrait de calculer le chemin optimal en ayant la connaissance totale de tous les chemins possibles partant de A et jusqu'à la destination. Mais les approches centralisées sont peu réalistes vu leurs besoins en mémoire et la nécessité des mises à jour constantes de la connaissance de la topologie en raison des déplacements.

C'est pourquoi nous avons proposé le protocole Mobile Recursivity ou MobileR. Le principe de MobileR est que chaque nœud courant va anticiper virtuellement par le calcul le routage et les déplacements associés sur les prochains sauts. Pour ce faire, MobileR tire parti du fait que les schémas de déplacements des nœuds peuvent souvent s'appliquer plusieurs fois dans la limite de portée radio d'un nœud. Par exemple, si le schéma de déplacement consiste à espacer chacun des nœuds du chemin de routage d'une distance R/4, MobileR pourra anticiper le routage sur les quatre prochains sauts en utilisant uniquement la connaissance du voisinage à un saut du nœud courant. À chaque étape du routage virtuel, le nœud courant va faire prolonger d'un saut le chemin dans la direction de la destination. Cette progression se fait dans l'espace du voisinage à un saut du nœud courant. Et elle ne se fait qu'en avançant toujours vers la destination, afin d'éviter les boucles. Elle est donc partielle, car la prolongation du chemin ne se fait qu'avec la connaissance du nœud courant. Dans la réalité, chaque nœud a une connaissance exacte de son voisinage, alors que dans notre cas on est limité par la connaissance qu'a le nœud courant de ses possibles successeurs dans le routage. MobileR exploite donc au maximum la localité pour anticiper récursivement les chemins de routage possibles.

Il ne s'agit donc plus de calculer le « meilleur» prochain saut, mais de calculer l'ensemble des chemins rendus possibles par la mobilité contrôlée pour en retenir le « meilleur ». L'intégralité des coûts de transmission et de déplacement de chaque chemin est considérée à chaque étape de la progression. Si l'on note $x_0x_1...x_ix_{i+1}..x_n$ la suite des nœuds formant le chemin calculé entre les nœuds A et B et avec $A = x_0$ et $B = x_n$, son coût peut être calculé comme suit :

$$cout(A,B) = \sum_{i=0}^{n}(cout_{radio}(X_i, X'_{i+1}) + cout_{mvt}(X_{i+1}, X'_{i+1})) \tag{3.1}$$

où X'_i et X'_{i+1} sont les nouvelles positions respectives des nœuds X_i etX_{i+1}.

Ce coût est évalué en rapport avec le progrès que permet le chemin pour atteindre la destination. Le progrès peut être calculé comme suit :

$$progres = |AD| - |X_nD| \tag{3.2}$$

MobileR retient pour le routage le chemin qui minimise le rapport coût du chemin sur le progrès jusqu'à la destination. Mais aucune information relative aux chemins n'est

échangée entre les nœuds. Seul le message à router est retransmis de nœud en nœud, assorti - ou précédé (selon l'heuristique) - si besoin est d'un ordre de déplacement. Aucune autre information n'est jointe au message. Ce mécanisme va obliger chaque nœud courant sur le chemin de routage à calculer « son » meilleur chemin de routage *basé sur sa connaissance de son voisinage*. On espère ainsi pallier la connaissance partielle du voisinage indiqué précédemment. Sur chaque nœud du routage, un nouveau « meilleur chemin », avec des possibilités nouvelles inconnues précédemment, est recherché. De plus ce principe permet de faire des économies de transmission radio et permet un protocole sans état parfaitement passable à l'échelle.

Il n'y a donc au final qu'un seul nœud qui bouge : le premier nœud du meilleur chemin de routage.

3.3 Calcul des chemins de routage virtuels

MobileR est donc un algorithme de routage géographique purement local et sans état. Pour un nœud S qui a un message à router vers la destination D il va fonctionner comme suit. S va d'abord établir la liste des voisins à un saut qui sont plus près de la destination que lui $N_D(S)$. S va ensuite les étudier individuellement un par un.

Dans un premier temps, en se basant sur les positions respectives du nœud courant, S, et de la destination, D, le nœud courant va calculer X' la nouvelle position de son voisin X. MobileR a donc virtuellement construit un chemin de routage à un saut $S \rightarrow X'$. Le rapport sur coût sur progrès associé à ce chemin est calculé.

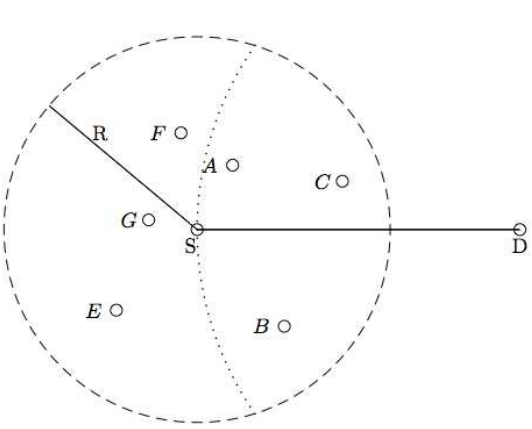
Dans un second temps, MobileR va appliquer récursivement la même méthode, et tenter d'allonger le chemin créé en partant de son extrémité X'_i. MobileR ne provoque pas de recul, à chaque étape il sélectionne un nœud plus près de la destination que l'extrémité finale relocalisée du chemin calculé considéré. Si X_{i+1} est plus proche de D que X', MobileR va alors construire le chemin $X_S \rightarrow X' \rightarrow X'_{i+1}$ et évaluer son coût sur progrès.

Et ainsi de suite, de saut en saut vers la destination, MobileR va calculer tous les chemins à un ou plusieurs sauts partant de S et estimer leur ratio coût sur progrès pour retenir celui qui le minimise le plus.

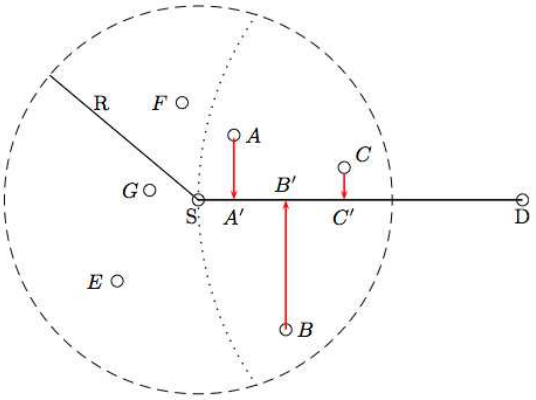
Comme on l'a expliqué précédemment, MobileR se base sur la connaissance du voisinage immédiat du nœud courant. Il n'a donc qu'une connaissance des possibilités de routage d'autant plus partielle que le chemin virtuel est long. La connaissance par le nœud courant S du voisinage d'un de ses successeurs X peut être formalisée comme suit :

$$N_D(X) = N_D(S) \cap \{v \in N_D(X) \text{ tq } |XD| \leq |SD|\}$$

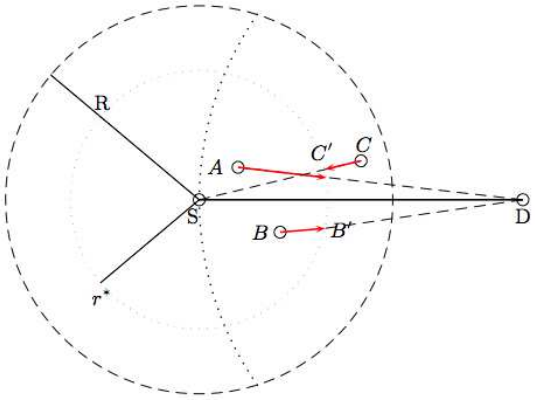
Cette connaissance partielle est un compromis entre l'énergie nécessaire pour l'améliorer (communications, mémoire, calculs...) et la précision et profondeur de l'anticipation. Elle est d'autant plus partielle que X, et surtout X' sont distants.



(a) Nœud S et son voisinage



(b) Application du schéma de déplacement CoMNet *ORouting*



(c) Application du schéma de déplacement CoMNet *MoveR*

FIGURE 3.2 – MobileR : Consommation énergétique en fonction du nombre de routages successifs.

Illustrons le fonctionnement de MobileR décrit dans l'algorithme 1 par un exemple. Dans la figure 3.2, MobileR agit comme suit pour le cas où S a un paquet à envoyer à D. Dans un premier temps, S calcule le sous-ensemble de ses voisins qui sont plus proches de D que lui-même : $N_D(S)$. Dans notre exemple on a $N_D(S) = \{A, B, C\}$. Si ce sous-ensemble est vide, le routage échoue (Alg. 1.4). Sinon, il calcule pour chacun des nœuds X de $N_D(S)$ les chemins commençant par $S \to X'$ (Alg. 1.8 / Alg. 2).

Ainsi S va calculer A', la nouvelle position de A. Tout comme B' et C' pour, respectivement, B et C. L'intégralité des coûts (transmissions, déplacements) associés aux chemins est calculée (Alg. 2.1, Alg. 2.2). S peut ainsi évaluer le coût sur progrès du chemin $S \to A'$ (Alg. 2.5).
MobileR va étendre ce chemin récursivement en partant de A'. Pour ce faire $N_D(A')$ est calculé, mais ce calcul est bien fait par S et uniquement avec la connaissance qu'il

Algorithm 1 $CalculChemin(S, D)$

$copMinimal \leftarrow +\infty$;
$copChemin \leftarrow 0$;
$prochainNoeud \leftarrow -1$;
if $N_D(S) \neq \emptyset$ **then**
    NULL ; ▷ Erreur : le routage échoue
else
    for all $X \in N_D(S)$ **do**
        $copChemin \leftarrow cheminRecursif(S, 0, X, S, D)$;
        if $copChemin < copMinimal$ **then** ;
            $copMinimal \leftarrow copChemin$;
            $prochainNoeud \leftarrow X$; ▷ X est le premier noeud sur le meilleur chemin
        end if
    end for
end if
return $prochainNoeud$;

a des voisins que A aurait en A' (Alg. 2.6). S va considérer chacun des nœuds dans cet ensemble (Alg. 2.7) de façon à étendre le chemin $S \rightarrow A'$, si c'est possible. Et ainsi de suite. Dans la figure 3.2c le chemin $S \rightarrow A' \rightarrow C'$ peut être construit.

Algorithm 2 $cheminRecursif(X'_{i-1}, coutChemin, X, S, D)$

$coutRadio \leftarrow cout_{radio}(|X'_{i-1}X|)$;
$coutMvt \leftarrow cout_{mvt}(|XX'|)$;
$cout \leftarrow coutRadio + coutMvt + coutChemin$;
$progres \leftarrow |SD| - |X'D|$;
$min \leftarrow \frac{cout}{progres}$;
for all $X_{i+1} \in (N_D(X')) \cap N_D(S))$ **do**
    $C \leftarrow cheminRecursif(X', cout, X_{i+1}, S, D)$; ▷ Prolongation récursive
    if $C < min$ **then**
        $min \leftarrow C$; ▷ Mémorisation du COP minimal des sous-chemins
    end if
end for
return min ; ▷ Retourne le COP minimal calculé

Si, pour une raison quelconque un nœud X ne peut pas se déplacer, alors $X' = X$ comme illustré dans l'algorithme 3.

Algorithm 3 relocalisation(nœudAEtudier)

1: **if** ($aPorteeCDS(X')$ *ET* (!$estDominant(noeudAEtudier)$) **then**
2: **return** X' ;
3: **else**
4: **return** X ;
5: **end if**

S retient finalement le premier saut du « meilleur chemin » possible : celui qui minimise le coût sur progrès (Alg. 2.9). Néanmoins, ce chemin est le meilleur selon la connaissance de S. Même si des chemins sont calculés, comme dans [EMSR08], ils ne sont pas stockés en mémoire. En effet, le nombre de chemins possibles croît avec la densité de la topologie. Non stockés, ils ne sont pas non plus communiqués aux nœuds en aval dans le chemin de routage (Alg. 2.12). Le nœud courant va simplement transmettre le paquet au premier nœud sur le chemin qu'il a retenu. Sans autres données. Son successeur dans le routage devra donc recalculer tous les chemins possibles. Ce mécanisme permet à chaque étape du routage de tendre vers l'optimum, car le calcul des chemins est fait avec un voisinage le plus précis possible.

On le voit, la profondeur de la récursivité, et donc la complexité du calcul, est d'autant plus importante que la densité du réseau est élevée. Pour permettre un passage à l'échelle sans heurt, on peut alors limiter la profondeur de la récursivité ou réduire le sous-ensemble des voisins sur lequel la récusrivité s'applique à chaque étape (voisins RNG, etc...). On perd alors en « optimalité » (locale) ce qu'on gagne en temps de calcul.

Afin de proposer un routage le plus efficace possible, deux schémas de déplacements des nœuds ont été considérés. MobileR-$MultipleORouting$ vise à réduire les coûts de communication radio en réduisant les distances de transmission. Alors que MobileR-$MultipleMoveR$ va lui viser à minimiser les déplacements des nœuds. Le premier schéma est adapté aux environnements où les déplacements sont peu coûteux en terme d'énergie, alors que le deuxième correspond plus à des environnements où les déplacements sont très coûteux en terme d'énergie.

3.4 Les différentes variantes de MobileR

3.4.1 MobileR-$MultipleORouting$

Illustré dans la figure 3.3, ce schéma de déplacement vise à aligner les nœuds sur une ligne droite entre la source et la destination. Cette approche permet en effet de minimiser les coûts de transmission radio ([HMSR11],[LNS07]). Le trajet de déplacement d'un nœud est donc celui de sa projection orthogonale sur la droite source destination, car elle minimise la distance à parcourir.

En utilisant un paquet de type *beacon* pour envoyer l'ordre de déplacement, on peut attendre qu'un nœud ait fini de se déplacer pour lui transmettre le paquet. Étant donné que le *beacon* est beaucoup plus petit en taille qu'un message standard, cela permet une économie d'énergie substantielle. Par la suite, les coûts de transmission sont réduits, car

les nœuds sont alors plus près les uns des autres. La figure 3.3 illustre une partie des

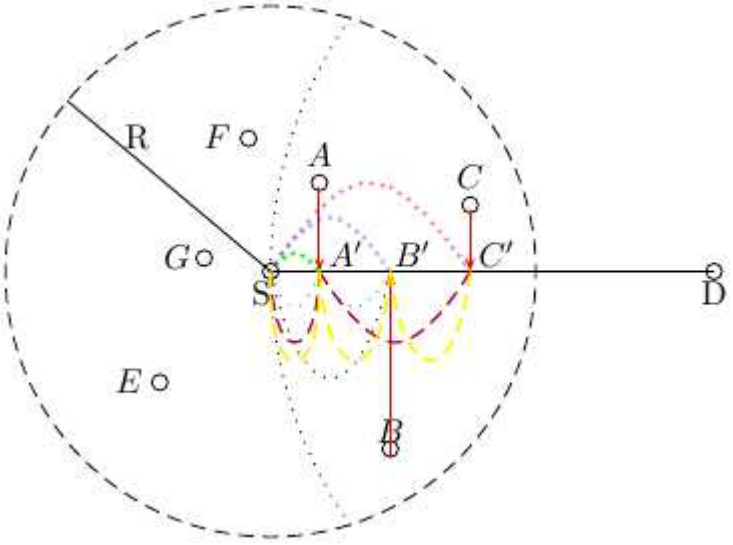


FIGURE 3.3 – Relocalisation des nœuds dans MobileR-*MultipleORouting* (flèches rouges). Une partie des différents chemins possibles est aussi représentée.

relocalisations possibles. Le chemin de S vers D en passant par A' est direct ($S \rightarrow A'$) car il n'existe pas de chemin plus court en terme d'énergie. B peut par contre être utilisé de deux manières différentes. Soit $S \rightarrow B'$, soit $S \rightarrow A' \rightarrow B'$ car A est plus près de S que B. De S à C' il existe de nombreuses possibilités : $S \rightarrow C'$, $S \rightarrow A' \rightarrow C'$, $S \rightarrow B' \rightarrow C'$, $S \rightarrow A' \rightarrow B' \rightarrow C'$. Il est à noter qu'à chaque étape de calcul, seuls sont considérés les nœuds qui se rapprochent de la destination, et ce afin d'éviter les boucles.

3.4.2 MobileR-$MultipleMove_R$

Ce schéma de déplacement vise à espacer régulièrement les différents nœuds sur le chemin de routage. Qui plus est, le déplacement des nœuds est fait de telle sorte que deux nœuds consécutifs sur le chemin de routage soient distants d'une valeur r^*. Cette valeur correspond au rayon de transmission idéal du matériel radio. Elle est issue de la formule du modèle de coût $cout_{radio}(r) = r^\alpha + c$ et est égale à $r^* = \sqrt[\alpha]{\frac{c}{\alpha-1}}$ [SL01b]. Dans ce schéma, l'ordre de déplacement est ajouté au message transféré. Le mouvement se fait donc après réception d'un message.

Le mouvement d'un nœud X dépend de sa distance par rapport à son prédécesseur S. Si la distance $|SX|$ est inférieure à r^* , le nouvelle position de X, X', se trouve à l'intersection du cercle centré en S et de rayon r' $C(S)_{r^*}$ et la droite (XD). Sinon, le déplacement se fait sur l'intersection entre $C(S)_{r^*}$ et la droite (SX).

La figure 3.4 illustre un exemple d'une étape de calcul récursif de chemin de routage

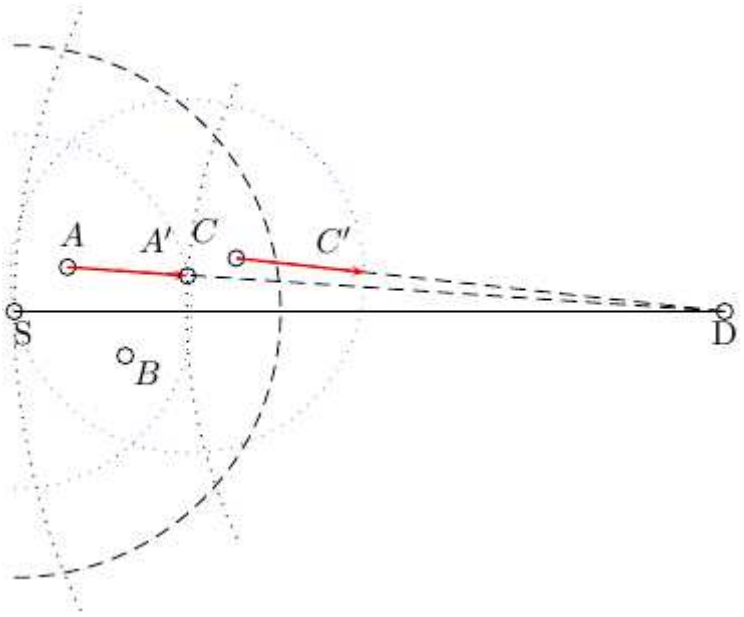


FIGURE 3.4 – Relocalisation des nœuds dans MobileR-$MultipleMove_R$.

par le nœud courant S. Elle montre les conséquences du déplacement d'un nœud sur le déplacement d'un autre. La relocalisation du nœud A en A' provoquerait la relocalisation de C en C' et donc la création du chemin $S \rightarrow A' \rightarrow C'$. De façon logique, l'anticipation est d'autant plus profonde que le ratio $\frac{r^*}{portee\ radio}$ est petit. Ainsi un nœud peut maximiser son anticipation et tendre vers la sélection du chemin optimal vers la destination sur les prochains sauts.

3.5 Simulations

Nous avons comparé les performances des différentes heuristiques de CoMNet *ORouting on the move*, $Move_{(DSr)}$ et $Move_{(r)}$ par rapport à leurs équivalents dans MobileR mais aussi avec le protocole MobileCOP. Nous avons employé le simulateur à événements discrets WSNet/Worldsens [FCF07] pour la simulation de larges réseaux de capteurs. La couche MAC utilisée est 802.11 DCF MAC layer, tandis que le modèle de propagation est celui de l'espace libre (*free space propagation*). Les paramètres utilisés dans le calcul des coûts de transmission $C_{radio}(.)$, l'énergie nécessaire pour transmettre un paquet entre deux nœuds, sont issus de la littérature [KNS06]. On suppose $c = 3 \times 10^8$ et $\alpha = 4$. Ces valeurs permettent de calculer un rayon de transmission optimal valant $r^* = 100$. Tous les nœuds sont équipés d'un dispositif de localisation dont le coût énergétique de fonctionnement est considéré égal pour tous, comme pour les paquets *Hello*. Sur chaque topologie, on lance dix routages successifs entre le nœud source et la destination. Les résultats sont la moyenne de plus de 100 simulations par type de topologie où les nœuds sont distribués de façon uniforme et chaque topologie est connexe. Chaque nœud peut

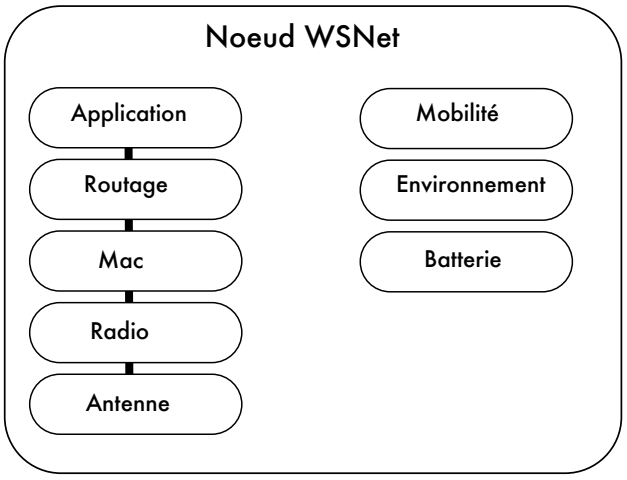


FIGURE 3.5 – Architecture WSNet

adapter sa portée de transmission entre 0 et 150 mètres.

3.5.1 Discussion sur l'outil de simulation

Les simulations ont été effectuées sur ordinateur avec le logiciel WSNet [1]. WSNet est un simulateur à large échelle pour les réseaux de capteurs sans fil développé au laboratoire CITI [FCF07]. Ses principales fonctionnalités sont la simulation de capteurs, de leur environnement et du médium radio. Elles en font un simulateur adapté à nos travaux. Dans WSNet les nœuds sont le résultat d'un assemblage de librairies logicielles représentant chacune un composant matériel ou un comportement, voire l'environnement, *c.f.* figure 3.5. Si la mobilité des nœuds est bien gérée, elle n'est toutefois qu'implémentée sous la forme de plusieurs librairies dont aucune ne permet de *contrôler* la mobilité *en cours* de simulation. La mobilité y est gérée en tant qu'entité de simulation autonome pour chaque nœud indépendante de la pile réseau. De plus la mobilité y est « gratuite », car elle n'a aucune influence sur le niveau de batterie comme le montre la figure 3.5.
Dans ce premier travail, la mobilité contrôlée est implémentée sous la forme d'une téléportation. Quand un nœud doit se déplacer, il vérifie s'il possède une quantité d'énergie suffisante pour effectuer le déplacement. Dans l'affirmative, le coût énergétique de ce déplacement est soustrait de la batterie globale de la simulation du nœud et le nœud est déplacé instantanément. Sinon, le nœud ne se déplace pas et son niveau de batterie est fixé à zéro.

Cette téléportation des nœuds n'est pas sans incidence, car instantanée. Elle réduit donc le temps nécessaire à l'auto-stabilisation des chemins adaptés et aucun paquet n'est

1. Voir http ://wsnet.gforge.inria.fr

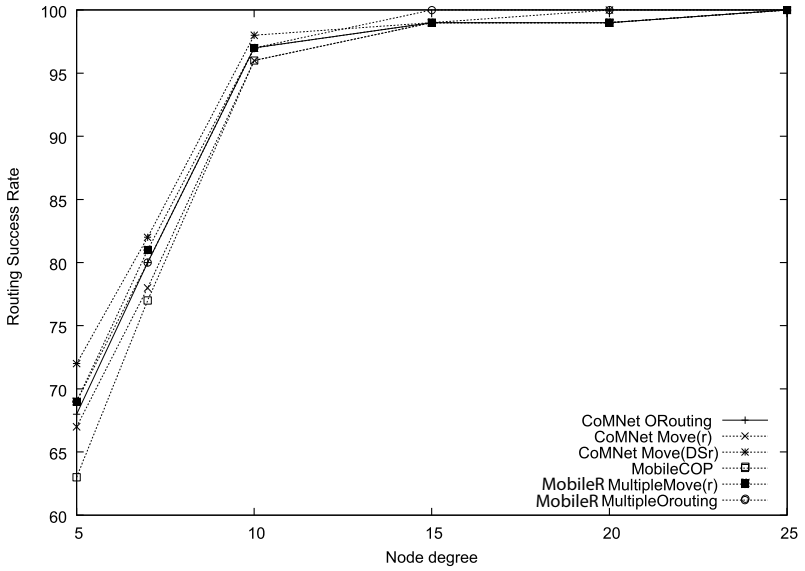


FIGURE 3.6 – MobileR : Taux de succès du routage

en attente qu'un nœud ait fini de se déplacer. De même, le mécanisme de maintien de la connexité du réseau utilise l'ensemble dominant connecté présenté dans [HMSR11]. Les déplacements étant instantanés, nous avons supposé que la mobilité d'un nœud ne met pas en danger la connexité du réseau si sa position de départ et d'arrivée sont dans le rayon de transmission d'un nœud dominant. Dans les travaux suivants, cette librairie a été affinée afin de permettre un découpage du mouvement en plusieurs étapes. Puis un autre mécanisme de maintien de la connexité a été exploré.

3.5.2 Taux de réussite

La figure 3.6 montre les taux de réussite des différents protocoles de routage par rapport au degré moyen du réseau.

Les résultats montrent que CoMNet présente de meilleurs résultats que MobileCOP pour les densités faibles car CoMNet garantit la connexité du réseau en limitant la mobilité grâce au CDS. Pour les densités plus élevées, ces deux approches présentent des résultats similaires.

Comme attendu *ORouting on the move* et MobileCOP présentent des résultats très proches. Ceux-ci s'expliquent, car leur comportement en termes de relocalisation est très similaire. L'approche la plus efficace est $Move_{(r)}$. Les variantes de MobileR surpassent leurs équivalents CoMNet quasiment à chaque fois. Cela suggère que le mécanisme d'anticipation du routage aurait tendance à réduire les trous dans la topologie grâce à la mobilité contrôlée.

3.5.3 Consommation énergétique

Nous avons évalué la consommation énergétique des différents protocoles en utilisant les modèles de coûts décrits avec les Équations. 1.1 et 1.2.

Pour pallier le manque de recherche dans les modèles de coût pour la mobilité des robots, le paramètre a de notre modèle 1.2 sera calculé comme suit :

- si émettre est aussi coûteux que de se déplacer, $C'_{radio}(.) = C_{mvt}(.)$, a est la solution de l'équation $C'_{radio}(r^*) = C_{mvt}(r^*)$.
- si émettre est beaucoup plus coûteux que de se déplacer, $C'_{radio}(.) >> C_{mvt}(.)$, alors a est la solution de l'équation $C'_{radio}(r^*) = 10^2 C_{mvt}(r^*)$.
- si se déplacer est beaucoup plus coûteux que d'émettre, $C'_{radio}(.) << C_{mvt}(.)$, alors a a est la solution de l'équation $C'_{radio}(r^*) = 10^{-2} C_{mvt}(r^*)$.

Notre approche est donc indépendante du modèle de coût. Toutefois, nous sommes conscient que certains modèles sont plus réalistes que d'autre. En général, il est plus coûteux pour un actionneur de se déplacer que d'émettre des ondes radio, mais nous nous devons de considérer tous les cas possibles.

3.5.3.1 Routages consécutifs

La figure 3.7 montre la consommation énergétique cumulée des différents protocoles de routage en fonction du nombre de routages successifs pour des topologies de degré moyen δ égal à 25. Les résultats des multiples simulations montrent que $MultipleMove_{(r)}$ est plus économe en énergie que $MultipleORouting$ quand les coûts de radio transmission et de déplacement sont égaux (figure 3.7a) que quand les coûts de radio transmission sont supérieurs (figure 3.7b). À l'inverse, leurs performances sont très proches quand il est plus coûteux de déplacer un nœud que d'émettre (figure 3.7c). Cela s'explique par le fonctionnement même des schémas de déplacement.

Dans $MultipleORouting$, le déplacement est ordonné via un *beacon* de coût énergétique négligeable. Le prochain nœud se déplace ensuite en se rapprochant du nœud courant. Le paquet n'est transmis *qu'ensuite*, quand la distance entre le nœud courant et le prochain nœud a été « optimisée ».
Le schéma de déplacement $MultipleMoveR$ provoque, lui, le transfert de l'ordre de déplacement *avec* le message à router. Et les déplacements provoqués par ce schéma y sont très réduits en terme de distance parcourue. Les communications radio sont donc particulièrement importantes dans ce schéma. Ce qui le pénalise dans le cadre du modèle de coût où les transmissions radio sont sans commune mesure avec le coût de la mobilité.

À partir de nos résultats, on constate que les différentes variantes de MobileR sont plus efficaces énergétiquement que leurs équivalents CoMNet respectifs. Cela démontre l'efficacité de l'approche de l'anticipation du routage multi-saut. Ainsi, $MultipleMove_{(r)}$ surpasse tous les autres protocoles en termes de performances énergétiques. Il fonctionne de façon particulièrement efficace dans les réseaux de haute densité. Ces performances sont à mettre en relation avec des déplacements qui s'intègrent particulièrement bien avec le principe de récursivité tout en proposant le meilleur équilibre entre déplacements

et communications radio de toutes les approches.

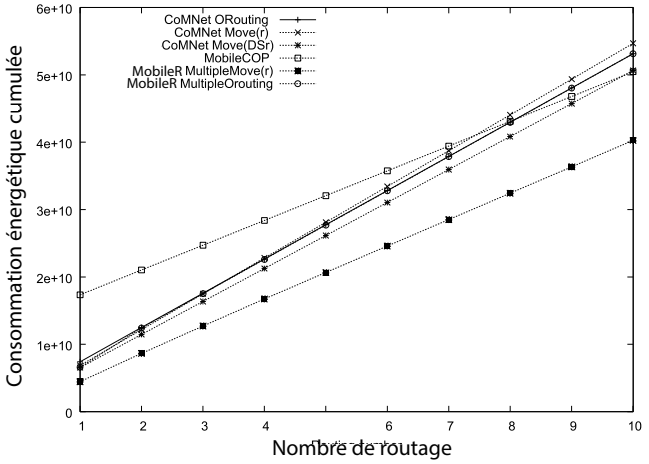
3.5.3.2 Effet de la variation du degré sur les performances

La figure 3.8 illustre la consommation énergétique cumulée pour dix routages successifs en fonction du degré moyen de la topologie sous-jacente. Les résultats de simulations montrent que cette consommation diminue lorsque le degré moyen des nœuds augmente. Plus précisément, quand le cas où le modèle de coût rend le coût de déplacement inférieur ou égal au coût de radio transmission, $MultipleMove_{(r)}$ surpasse $MultipleORouting$. Il en effet beaucoup plus économe en énergie, et l'écart va croissant avec l'augmentation de la densité. À l'inverse, dans le cas où se déplacer est beaucoup plus coûteux que les transmissions, les performances des différentes heuristiques sont similaires. En effet, notre approche tient compte du coût de la mobilité des nœuds. Dès lors, quand ce coût devient très important vis-à-vis des communications radio, les différents variants de MobileR vont limiter au maximum les déplacements de nœuds dans leur construction de chemin. Ainsi, même si les chemins construits ne sont pas du tout les mêmes, avec des distances radio qui peuvent énormément varier, le peu d'importance des coûts radio par rapport au coût de déplacement explique les résultats très similaires. C'est la (petite quantité) de mobilité, très coûteuse, qui rend les résultats si proches.

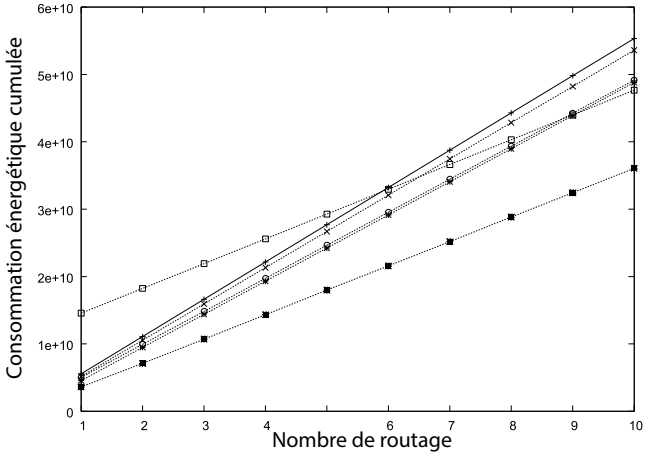
De façon générale, MobileR apparaît plus efficace en terme de consommation énergétique comparé à l'approche non récursive de CoMNet. Cela ressort particulièrement quand le degré moyen de la topologie est supérieur à 15 : $\delta \geq 15$. On peut en déduire qu'une fois un seuil atteint (ici $\delta = 15$), de plus en plus de chemins sont possibles entre les différents nœuds et leurs voisins. Dès lors MobileR tire parti de ces multiples possibilités de chemin pour calculer la meilleure solution possible en anticipant au maximum. Ces possibilités d'anticipation ne sont pas sans contrepartie. La complexité et donc le temps de calcul des chemins par le nœud courant augmente fortement avec l'augmentation du degré moyen. Toutefois cet aspect n'est particulièrement important que dans les simulations, car le médium radio est ici idéal. Un médium radio non idéal impliquerait un voisinage moins important et donc un temps de calcul moindre.

3.6 Conclusion et perspectives

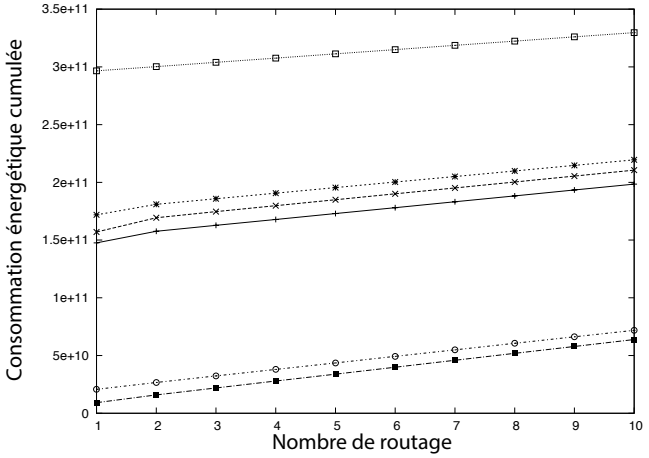
Dans ce chapitre, nous avons introduit un nouveau protocole de routage pour les réseaux de capteurs et d'actionneurs. Ce protocole tire parti de la mobilité locale permise par les actionneurs pour adapter la topologie du réseau à son trafic. Mais, contrairement aux autres protocoles, le protocole va tenir compte des modifications des possibilités de routage que la mobilité induit. Il va ainsi, à chaque étape du routage, anticiper autant que possible par le calcul les conséquences des multiples choix possibles pour n'en retenir que le meilleur. Il reste pour autant purement local. Nous avons analysé par la simulation les performances de notre proposition. Nous avons pu constater qu'elle présentait les meilleures performances énergétiques parmi les protocoles de routage pour



(a) $C_{radio} = C_{mvt}$

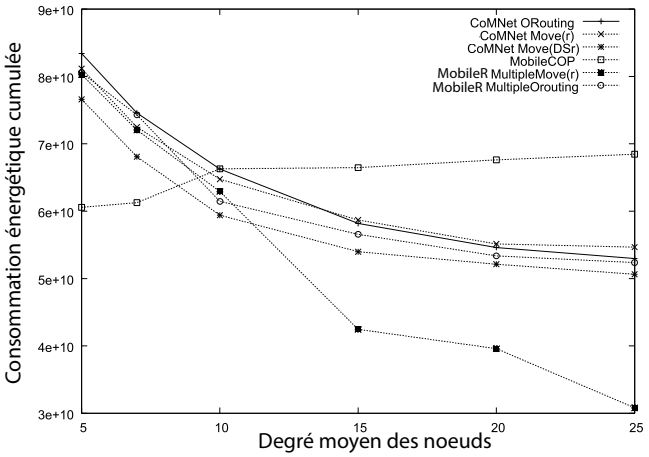


(b) $C_{radio} >> C_{mvt}$

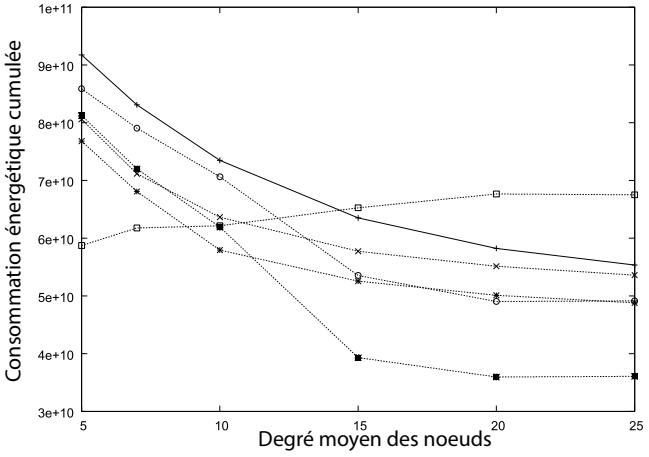


(c) $C_{radio} << C_{mvt}$

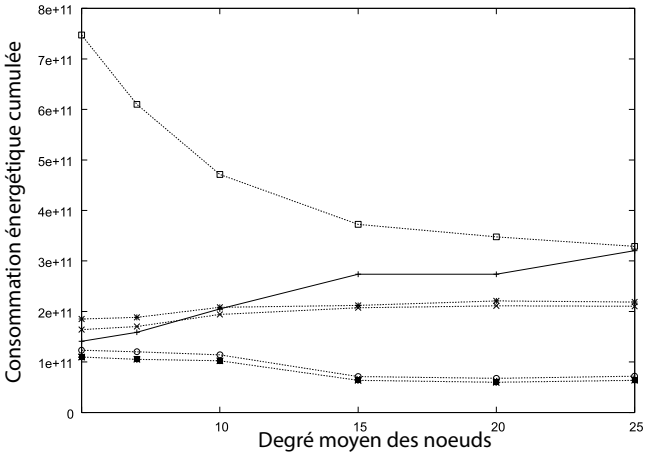
FIGURE 3.7 – MobileR : Consommation énergétique en fonction du nombre de routages successifs.



(a) $C_{radio} = C_{mvt}$



(b) $C_{radio} >> C_{mvt}$



(c) $C_{radio} << C_{mvt}$

FIGURE 3.8 – MobileR :Consommation énergétique cumulée en fonction du degré.

actionneurs, preuve que le mécanisme d'anticipation par calcul du routage mobile est efficace.

Toutefois, l'anticipation récursive est au prix d'une complexité d'autant plus importante que la densité du réseau est élevée, il s'agit donc – pour la rendre utilisable – de faire un compromis entre la profondeur de l'anticipation (ou le nombre de voisins sur laquelle on l'applique) et le surcoût en calcul. Cela ne peut se faire qu'avec des informations précises sur la topologie du réseau, le schéma de déplacement et la puissance de calcul des nœuds.

Dans le futur, il serait intéressant de reprendre le travail sur MobileR avec un modèle de simulation de mobilité plus fin. En effet, dans notre approche, nous considérons des déplacements instantanés et en simplifiant la gestion de la connexité globale du réseau. Les déplacements calculés par MobileR sont ainsi validés et utilisés en simulation si la position de départ et celle d'arrivée d'un nœud mobile sont couvertes par le CDS et non si la totalité du trajet est couverte. Il s'agirait d'en mesurer l'impact sur les performances et de tester d'autres mécanismes alternatifs de maintien de connexité du réseau. De même, MobileR étant indépendant des schémas de déplacement, il semblerait hautement intéressants de concevoir des schémas de déplacement *pensés* avec l'hypothèse de récursivité appliquée sur chaque nœud courant.

Chapitre 4

Mobilité et Routage Multi-Sources

4.1 Introduction

Comme on l'a vu dans les chapitres précédents, l'auto-adaptation de la topologie du réseau à son activité dans les réseaux de capteurs et d'actionneurs présente de nombreux avantages. Cette approche récente présente en effet des résultats supérieurs aux approches de routage traditionnelles. L'opération de routage dans les réseaux de capteurs est une activité essentielle. En effet, les capteurs sont nombreux, et ont des capacités trop limitées en termes de puissance de calcul et de batterie pour pouvoir délivrer directement leurs messages à la station de base. Ils doivent donc coopérer pour mener à bien un routage le plus efficace possible. Par efficace, on entend un routage dont le coût énergétique ne provoque pas une surcharge trop importante par rapport au poids énergétique du message.

Une des bonnes approches limitant la surcharge énergétique consiste à concevoir des protocoles locaux. Les choix de routage sont pris au fur et à mesure du routage, en se basant uniquement sur des informations détenues à chaque fois par le nœud courant. Ces informations sont généralement la position des nœuds voisins du nœud courant, sa propre position, la position de la source du paquet, et enfin celle de la station de base. Le protocole MobileR que nous avons proposé, et son prédécesseur, CoMNet, sont eux aussi locaux. Ils utilisent la possibilité de déplacement des nœuds pour adapter localement la topologie du réseau au trafic. Globalement, tous les protocoles de routage pour actionneurs visent à aligner (plus ou moins précisément) les nœuds sur la droite virtuelle formée par le nœud source S et la destination D. L'idée sous-jacente est qu'en adaptant cette topologie, le routage sera moins coûteux en terme de coût radio car la distance entre les nœuds impliqués dans le routage est réduite.

Cette approche du mouvement peut être source de conflits. On se place en effet ici dans le cadre des réseaux de capteurs. Les réseaux de capteurs sont principalement utilisés pour monitorer l'environnement et en détecter les changements. Ils sont par exemple employés dans le cadre de la détection de feu de forêt. L'importance de leur application implique une certaine redondance de ces capteurs. Il existe alors une forte

probabilité qu'un même événement soit détecté simultanément par plusieurs capteurs proches. Ceux-ci vont alors émettre des données à router vers la station de base. Ces capteurs sources sont donc physiquement proches, et leurs chemins de routage le sont eux aussi d'autant plus qu'on se rapproche de la destination du message. Dès lors, certains nœuds du réseau appartiennent à plusieurs de ces chemins de routage. Or, dans les réseaux d'actionneurs, les nœuds appartenant à plusieurs chemins de routage, ou nœuds d'intersection NI, vont sans cesse être déplacés afin d'adapter la topologie des multiples chemins auxquels ils appartiennent. Les NI vont donc mourir prématurément à cause de cette oscillation entre les chemins. Ce phénomène est d'autant plus critique que la mort d'un de ces nœuds d'intersection ne va pas impacter seulement une mais plusieurs routes : toutes les routes passant par ce nœud d'intersection.

C'est pourquoi nous proposons PAMAL, pour *PAth Merging ALgorithm*. PAMAL est un protocole de routage géographique pour actionneurs qui adapte la topologie du réseau en relocalisant les nœuds des différents chemins de routage. Mais PAMAL va aussi détecter et tirer parti *localement* de ces intersections des chemins de routage. PAMAL va faire stopper les oscillations des nœuds d'intersections (NI). Il va aussi altérer localement les schémas de routage en ces points précis. La relocalisation des nœuds situés en amont des nœuds d'intersection est alors impactée en aval de ces nœuds au fur et à mesure du routage. PAMAL va dès lors provoquer la fusion deux à deux des différents chemins de routage qui s'entrecroisent, et ce de plus en plus loin de la destination. Sur cette partie fusionnée des chemins nous proposons de réaliser une agrégation des paquets permettant ainsi des économies d'énergie et de bande passante.

Dans ce chapitre, nous décrivons dans un premier temps dans la Section 4.2 les motivations nous ayant conduit à envisager un nouvel algorithme de routage. Nous y expliquons et illustrons pourquoi les spécificités des réseaux de capteurs doivent être prises en considération pour proposer des protocoles de routage efficaces et économes en énergie. Notre proposition est exposée dans la Section 4.3. Nous donnons ensuite le détail des algorithmes nécessaires au bon fonctionnement de notre proposition. Nous proposons aussi une nouvelle approche de routage géographique multi-schéma afin d'adapter au mieux la topologie du réseau. Les simulations viennent enfin illustrer nos résultats et valider notre proposition. PAMAL est ainsi comparé à CoMNet. Les résultats comportementaux de la Section 4.5.3 confirment la fusion physique progressive des chemins de routage, tandis que les résultats de consommation de la section 4.5.2 montrent une augmentation significative de la durée de vie du réseau.

4.2 Motivations

L'idée d'utiliser la mobilité comme une primitive du routage afin d'améliorer la durée de vie des capteurs a fait ses preuves. Toutefois, toutes les approches existantes reposent sur l'hypothèse d'un routage avec une unique source et une unique destination. Les solutions de la littérature consistent à déplacer les nœuds sur le chemin de routage entre le nœud source S et la destination D. Cette approche vise à diminuer les

coûts de communication radio en adaptant la topologie du réseau à son trafic. Toutefois l'activité des réseaux de capteurs est souvent déclenchée en réponse à un événement (incendie, détection de présence...). Un même événement a de fortes probabilités de déclencher l'envoi de messages par plusieurs capteurs, géographiquement proches. Ces nœuds sources Si vont chacun établir et adapter physiquement des chemins de routage $Si \rightarrow \cdots \rightarrow D$.

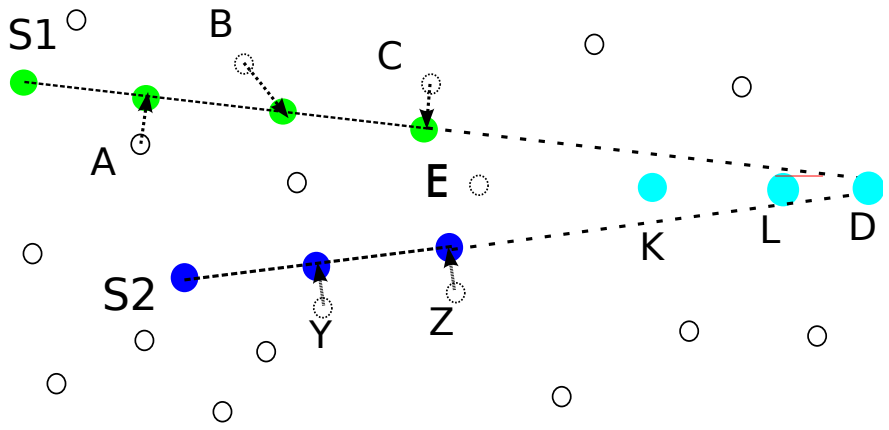
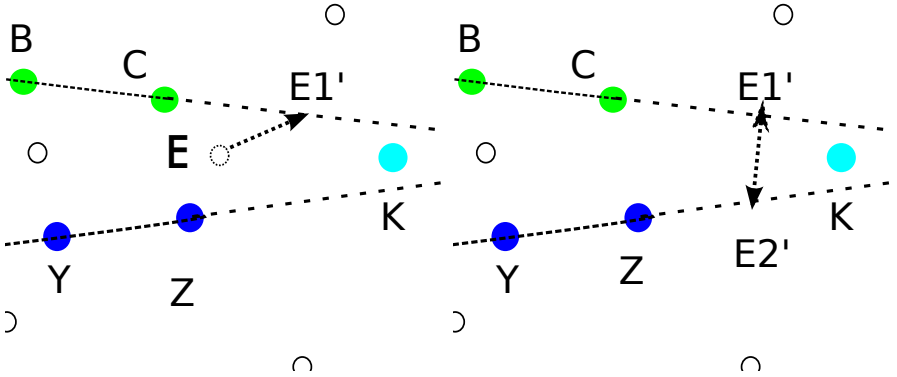


FIGURE 4.1 – PAMAL : Routage deux sources $S1$ et $S2$ vers D.

La figure 4.1 illustre l'exemple d'un événement détecté par deux nœuds source $S1$ et $S2$. On a donc l'apparition de deux chemins de routage $S1 \rightarrow \cdots \rightarrow D$ et $S2 \rightarrow \cdots \rightarrow D$. On voit que sur ces deux chemins, les schémas de déplacement vont entrer en conflit, au niveau de la sélection du nœud E (figures 4.2a et 4.2b).



(a) PAMAL : E se déplace en $E1$ sur le chemin $S_1 \rightarrow \cdots \rightarrow D$

(b) PAMAL : E se déplace en $E2$ sur le chemin $S_2 \rightarrow \cdots \rightarrow D$

FIGURE 4.2 – Oscillation du nœud E entre les positions $E1$ et $E2$.

On considère dans notre cas que c'est $S1$ qui détecte l'événement en premier. Le chemin $S1 \rightarrow \cdots \rightarrow D$ est donc adapté. Au premier saut, A est déplacé en A', B

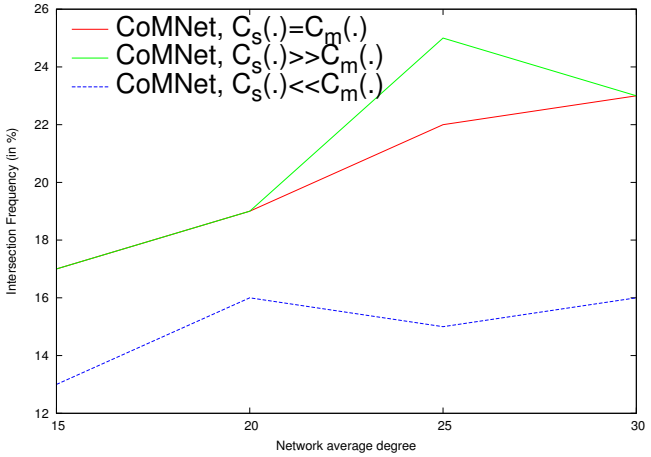


FIGURE 4.3 – Fréquence des intersections

en B', C en C' et E est déplacé en $E1'$. Les nœuds K et L ne sont ici pas déplacés car la distance euclidienne entre leur position actuelle et celle sur $(S1D)$ est inférieure au déplacement minimum requis. $S2$ détecte à son tour l'événement : son message est routé et Y et Z sont déplacés en respectivement Y' et Z'. Le seul nœud à portée de Z' permettant de progresser vers la destination est $E1'$, il va donc être déplacé sur $(S2D)$ en $E2'$. Le nœud va alors sans cesse être déplacé de $E1'$ en $E2'$ au fur et à mesure des routages.

Nous avons cherché à en évaluer la fréquence de ces intersections de chemins de routage. La figure 4.3 représente la fréquence d'intersection de deux chemins dont les sources ont été choisies aléatoirement en fonction du degré de la topologie δ (δ = 15, 20, 25). Le routage entre $S1 \rightarrow \cdots \rightarrow D$ et $S1 \rightarrow \cdots \rightarrow D$ sont effectués grâce au protocole CoMNet. Les résultats montrent que la fréquence d'intersection des chemins de routage varie de 13% à plus de 20%. Elle augmente avec le degré moyen pour atteindre un maximum à un degré de 25 voire 30. Ces résultats prouvent l'importance de prendre en compte ces intersections et leur impact, d'autant que dans ces simulations les sources sont placées aléatoirement. Dans un réseau réel, les sources relatives à un même événement sont géographiquement proches et les intersections d'autant plus fréquentes.

4.3 Proposition

L'efficacité et la légèreté que permet un protocole de routage géographique localisé sont essentielles pour les réseaux d'actionneurs sans architecture. Nous allons donc centrer notre proposition sur le nœud sur lequel l'intersection des chemins a lieu. Dans PAMAL, quand un nœud détecte qu'il reçoit des paquets d'origines différentes (et de même destination), il devient un nœud d'intersection NI.

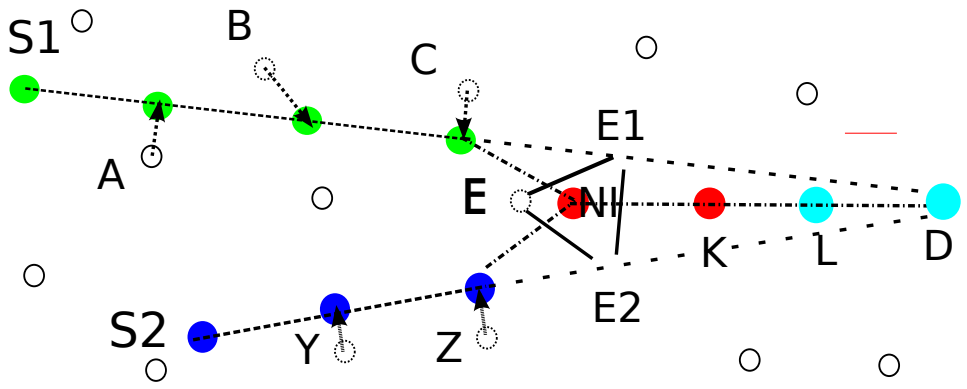


FIGURE 4.4 – Apparition d'un nœud d'intersection NI

En tant que nœud d'intersection, il va alors arrêter d'osciller et commencer à agréger les paquets des deux chemins. Ainsi, sur la figure 4.4, le nœud E reçoit des paquets provenant des chemins $S1 \rightarrow \cdots \rightarrow D$ et $S2 \rightarrow \cdots \rightarrow D$. Le nœud C, qui appartient au chemin $S1 \rightarrow \cdots \rightarrow D$, demande à E de se relocaliser en $E1$. Z, qui appartient à $S2 \rightarrow \cdots \rightarrow D$, lui demande d'aller en $E2$. E va alors devenir un NI. Il va se relocaliser en l'isobarycentre de $E1$, $E2$, E et y rester. Les paquets routés sur la partie commune aux deux chemins $NI \rightarrow \cdots \rightarrow D$ sont agrégés. L'arrêt des oscillations permet d'éviter une mort prématurée du nœud NI à cause de déplacements incessants. En lui faisant rejoindre l'isobarycentre des positions, on vise à lui donner une position d'équilibre entre les ordres de relocalisation qui se succèdent en provenance des différents chemins. Dans notre travail, nous avons aussi proposé d'effectuer de l'agrégation de données sur ce nœud afin de diminuer la surcharge due au protocole de routage. Le mécanisme est des plus basiques car il n'est pas l'objet de ce travail. Chaque paquet agrégé correspond ainsi à la fusion de deux paquets non agrégés. Pour chaque paquet agrégé, on économise donc la valeur d'un entête MAC.

Une fois arrivé en sa nouvelle position, le NI avertit son voisinage de son existence et de sa position. Tous les nœuds qui reçoivent ce message sont alors informés qu'ils doivent router un paquet pour D, et doivent le faire passer par le NI. Le NI remplace donc la destination D dans leurs schémas de déplacement. La figure 4.5 propose un agrandissement de la figures 4.4. Le nœud E, devenu un NI, s'est relocalisé. Il en avertit ses voisins. L'apparition du NI va alors provoquer une relocalisation de C sur la droite $(B\ NI)$ et non plus sur $(S1\ NI)$, et une relocalisation de Z sur $(Y\ NI)$ au lieu de $(S2\ NI)$ et faire fusionner les chemins de routage $S1 \rightarrow \cdots \rightarrow D$ et $S2 \rightarrow \cdots \rightarrow D$.

Le NI remplaçant – localement – la destination, il va provoquer une fusion des chemins de routage en amont de sa position. Cette fusion va progressivement permettre l'émergence d'un autre NI pour les mêmes sources/destination encore plus en amont. Dans cette hypothèse, l'ancien NI, qui recevrait alors des paquets déjà agrégés, avertit son voisinage qu'il n'est plus un NI et reprend un comportement standard. Routage

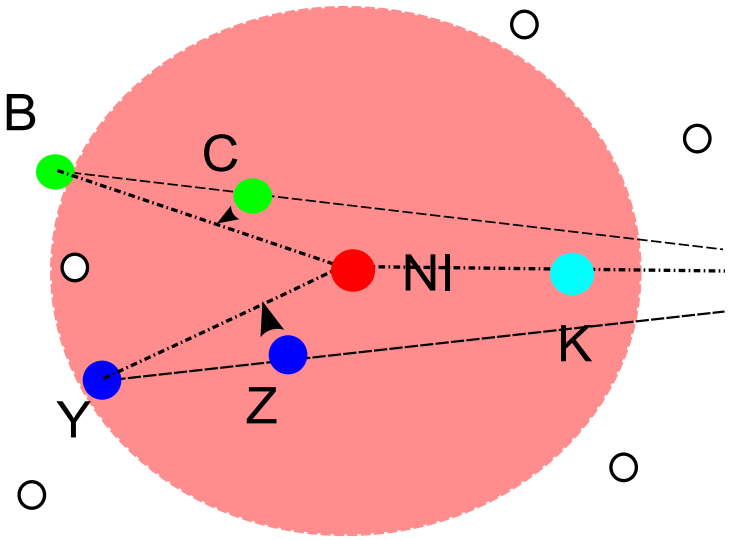


FIGURE 4.5 – L'apparition du NI modifie les relocalisations.

après routage, la partie fusionnée des chemins de routage s'allonge. Les algorithmes de routage permettant le fonctionnement de PAMAL sont détaillés dans la section suivante.

4.4 Algorithme

4.4.1 Gestion des nœuds d'intersection

Un nœud devient automatiquement un nœud d'intersection dès lors qu'il constate qu'il reçoit des paquets provenant de plusieurs sources différentes. Pour ce faire, pour chaque paquet reçu, il va vérifier les méta-données du paquet pour en vérifier la source (Alg. 4.3). Si la source est nouvelle, elle est mémorisée (Alg. 4.8). Sinon, le nœud devient un nœud d'intersection (Alg. 4.10). Il avertit alors via un *beacon* ses voisins de le considérer comme destination dans leurs schémas de déplacement.
Un nœud d'intersection redevient un nœud standard dès lors qu'il reçoit des paquets déjà agrégés (Alg. 4.4).

Selon que le nœud courant est un nœud d'intersection ou pas, le paquet est transmis agrégé ou non. La sélection du prochain saut est effectuée grâce à un algorithme de routage glouton multi-schéma : à chaque étape du routage, il considère l'application de plusieurs schémas de déplacement pour appliquer celui qui minimise le coût sur progrès du routage vers la destination.

4.4.2 Routage glouton multi-schéma

Comme on l'a vu dans le chapitre précédent, le schéma de déplacement a une importance majeure dans nos protocoles de routage pour réseaux de capteurs et actionneurs. C'est lui qui conditionne quelle sera la nouvelle position du prochain nœud dans le rou-

Algorithm 4 receptionPaquet(p) -Execute par un nœud A

```
1: sourcePrecedente                                  ▷ initialise au démarrage a -1
2: if estUnNI(A) then
3:     if estAggrege(p) then
4:         redevientStandard();
5:         transmet(p, prochainNud(A, D));
6:     end if
7: else if sourcePrecedente = −1 then
8:     sourcePrecedente ← source(p);
9:     transmet(p, prochainNud(A, D);
10: else if source(p) ≠ sourcePrecedente then
11:    demarreAggregation();
12:    devientIN();
13:    transmetEtAggrege(p, prochainNud(A, D));
14: end if
```

tage. Ces schémas sont un compromis entre la quantité de mouvement qu'ils imposent et l'économie provoquée sur les communications radio. On a vu ainsi les protocoles évoluer, en proposant un seul (MobileCOP), puis plusieurs schémas (CoMNet). Mais dans tous les cas, le protocole impose de choisir un seul et unique schéma de déplacement qui sera utilisé pour toute la durée de vie du réseau. L'approche de MobileR exploite les schémas de déplacement récursivement en anticipant autant que faire se peut le routage sur plusieurs sauts. Mais encore une fois, le schéma utilisé reste le même pendant toute la durée de vie du réseau.

Dans ce protocole, nous avons cherché à exploiter au mieux les possibilités de déplacement. Pourquoi alors se limiter à un seul schéma de déplacement dans le routage ? Nous avons donc proposé, à chaque étape du routage, de considérer l'utilisation de *chacun* des schémas de déplacement. Nous pouvons ainsi réaliser des économies d'énergie au prix d'une complexité plus élevée. La complexité n'atteint toutefois pas celle de MobileR, car l'on n'essaie pas d'anticiper récursivement sur plusieurs sauts. On se limite à appliquer, pour chacun des prochains nœuds possibles dans la direction de la destination, le calcul du ratio coût sur progrès selon les trois schémas de déplacement proposés par CoMNet comme le détaille l'algorithme 5.

Un nœud courant A exécute l'algorithme 5. Dans un premier temps, A vérifie s'il a été informé de l'existence d'un nœud d'intersection pour le couple source-destination du paquet considéré. Dans l'affirmative (Alg. 5.4), A va utiliser la position du NI comme position de la destination D dans les schémas de déplacement (Alg. 5.5). A calcule alors, pour chacun de ses voisins V, le rapport du coût sur progrès de leur déplacement selon les trois schémas de déplacements (Alg. 5.10) et retient le nœud – et sa nouvelle position– qui minimise ce rapport.

Algorithm 5 prochainNœud(A,D)

$prochain \leftarrow -1$
$prochain\prime \leftarrow \{0,0,0\}$ // next hop relocation
$minCOP \leftarrow +\infty$ // best COP over $N_D(a)$
if $estUnNIpourDestinationDe(p)$ **then**
    $D \leftarrow intersectionNode$;
end if
if $N_D(A) = \emptyset$ **then**
return NULL ; ▷ le routage échoue
else
    for all $\{V \in N_D(A)\}$ **do**
        $V\prime_{or}, v\prime_{mr}, v\prime_{mdsr} \leftarrow \{0,0,0\}$
        $V\prime_{or} \leftarrow \text{CoMNet} - \text{ORouting}(A, V, D)$; ▷ $V\prime$ selon CoMNet-ORouting
        $V\prime_{mr} \leftarrow \text{CoMNet} - \text{Move}_{\text{R}}(A, V, D)$; ▷ $V\prime$ selon CoMNet-Move$_R$
        $V\prime'_{mdsr} \leftarrow \text{CoMNet} - \text{Move}_{\text{DSR}}(A, V, D)$; ▷ $V\prime$ selon CoMNet-Move$_{DSR}$

$$vCOP \leftarrow min \begin{cases} COP(A, V, V\prime_{or}); \\ COP(A, V, V\prime_{mr}); \\ COP(A, V, V\prime_{mdsr}; \end{cases}$$

        if $(vCOP < minCOP)$ **then**
            $prochain \leftarrow v$;
            $prochain\prime \leftarrow$ position associée pour le COP
            $minCOP \leftarrow vCOP$;
        end if
    end for
return $\{prochain, prochain\prime\}$ ▷ Le routage réussit.
end if

4.5 Résultats de simulation

Nous évaluons les performances de PAMAL comparées à CoMNet, le protocole de routage pour actionneurs le plus efficace à notre connaissance. Les résultats sont la moyenne de 25 simulations, réalisées sur le simulateur WSNET pour des topologies dont le degré moyen varie de 15 à 25. On appelle densité d'un réseau le nombre moyen de voisins d'un nœud du réseau. Les topologies sont des carrés de 500x500m avec des nœuds uniformément et aléatoirement répartis. Les paquets sont de taille 200 ou 400 octets avec un entête de 40 octets. Le facteur d'agrégation a été fixé à 1 pour 2. Tous les capteurs sont des actionneurs capables de se déplacer.

4.5.1 Modèle de simulation

Afin de pouvoir mesurer l'impact de l'agrégation de données, nous avons adapté le modèle de coût des transmissions radio défini dans [RM99] :

$$C_{radio}(r) = c + r^{\alpha} \tag{4.1}$$

Nous y avons introduit la taille d'un paquet q en octets :

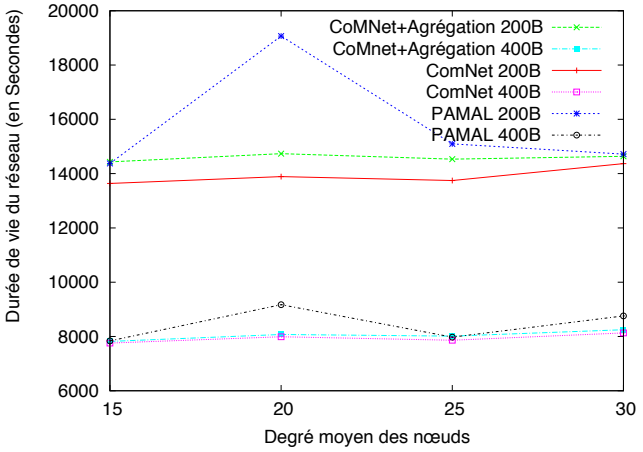
$$C_{radio}(r) = c + q * r^{\alpha} \tag{4.2}$$

Dans cette équation :
— la variable r représente la distance euclidienne parcourue par la transmission radio,
— la variable q représente la taille totale d'un paquet (données + méta données) en octets,
— C est une constante correspondant au coût de traitement du signal par l'électronique,
— α est une constante réelle ($1 < \alpha < 8$) qui représente l'atténuation du signal dans le milieu de propagation.

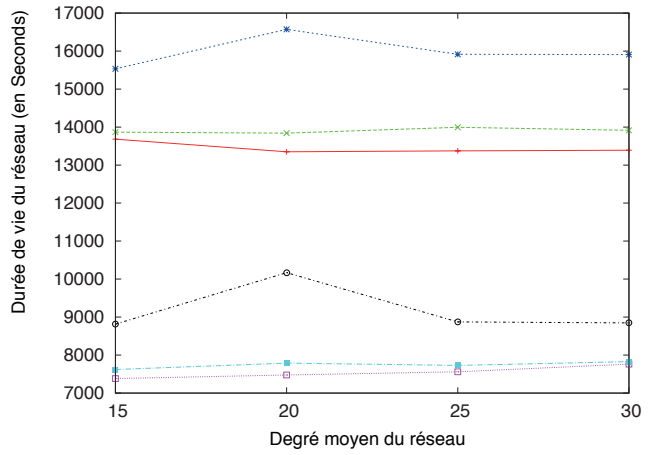
De cette manière, on rend mesurable la consommation énergétique en fonction de plusieurs tailles de paquets.

4.5.2 Durée de vie du réseau

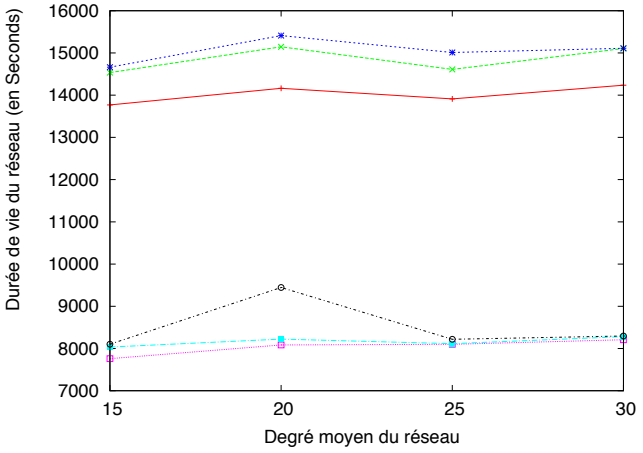
La figure 4.6 montre la durée de vie moyenne du réseau en fonction de sa densité, de la taille des paquets routés et du protocole employé. Ici, on appelle durée de vie moyenne du réseau la moyenne des dates maximales de réception par la destination d'un paquet émis par chacune des deux différentes sources. Les différents modèles de coût ont aussi été employés afin de tester leur impact et leurs résultats sont illustrés par les différentes sous-figures. Nous nous sommes limités à deux sources dans les simulations mais notre proposition reste transposable à des cas avec n sources.



(a) $C_{radio} = C_{mvt}$



(b) $C_{radio} << C_{mvt}$



(c) $C_{radio} >> C_{mvt}$

FIGURE 4.6 – Durée de vie du réseau en fonction de son degré, du modèle de coût et du protocole utilisé.

Afin de distinguer le gain en termes de durée de vie du réseau provoqué par l'agrégation du gain apporté par le mécanisme de fusion des chemins, nous avons comparé les performances relatives du CoMNet original – sans agrégation donc –, d'un CoMNet avec agrégation et de PAMAL. L'agrégation est partie prenante dans le routage de PAMAL.

L'agrégation en elle-même apporte effectivement un gain dans la durée de vie du réseau, quels que soient le modèle de coût ou le degré moyen du réseau. Toutefois, celui-ci reste modeste. Ce gain est à mettre en relation avec l'augmentation de la charge utile des paquets fusionnés grâce à la diminution de la part de l'entête dans la taille globale du paquet. Ainsi, avec des paquets émis par les sources de 200 octets, un entête de 40 octets représente 16% de surcharge. En fusionnant deux paquets, on arrive à 400 octets pour toujours 40 octets d'entête soit 9% de surcharge. Dans le cadre de paquets de 400 octets fusionnés la surcharge passe elle de 9% à 5%.
Cette réduction de la surcharge des entêtes n'est toutefois effective que pour une partie très courte des chemins de routage pour CoMNet ou CoMNet avec agrégation car ils n'intègrent aucun mécanisme de fusion des chemins.

Le gain dû à l'agrégation des paquets est amplifié grâce au mécanisme de fusion des chemins qui va en amplifier l'utilisation. En effet, l'apparition du NI va provoquer une convergence physique des différents chemins de routage en amont de sa position. Un autre NI, plus près des sources, peut donc finir par apparaître, et ainsi, de NI en NI, la partie fusionnée des chemins de routage va s'allonger en remontant vers les différentes sources. L'agrégation est donc faite sur une portion commune de plus en plus longue. C'est ce mécanisme, associé au blocage de l'oscillation du NI, qui permet à PAMAL d'avoir un gain sur la durée de vie du réseau beaucoup plus important. Ce gain peut atteindre jusqu'à 37% par rapport au CoMNet original (voir figure 4.6a).

Indépendamment des modèles de coût, on observe que le gain est toujours maximum quand le degré moyen dans la topologie est égal à 20. Ce gain diminue fortement si la densité ré-augmente. Il s'explique par des raisons liées à la fois à la topologie et au mécanisme de maintien de la connexité. Quand $\delta < 20$, le mécanisme de maintien de la connexité (CDS) limite encore fortement la mobilité. À l'opposé, quand le degré moyen augmente, il y a tellement de chemins possibles que les intersections sont moins fréquentes et ne se font que tardivement – près de la destination.

4.5.3 Adaptations de la topologie

Il est assez difficile de mesurer les modifications topologiques provoquées par des routages simultanés de façon pertinente. C'est pourquoi les figures 4.7 et 4.8 montrent l'évolution de la longueur de la partie commune du chemin de routage – en nombre de sauts – en fonction du temps. De cette manière, on visualise au mieux l'apport de PAMAL par rapport à CoMNet.

De façon générale, et quels que soient les paramètres pris en compte (taille des pa-

quets, modèle de coût...), on constate effectivement que PAMAL provoque une fusion des chemins de routage. Cette fusion apparaît significativement plus longue avec PAMAL qu'avec CoMNet (avec ou sans agrégation). Elle y est aussi beaucoup plus rapide. Cela s'explique par le mécanisme d'avertissement des nœuds d'intersection développé dans PAMAL. Dans CoMNet, on observe que la longueur de la partie fusionnée des chemins est beaucoup plus stable. On n'observe qu'un pic à la fin des simulations. Ce pic, présent aussi avec PAMAL mais plus tard, a lieu quand la plupart des simulations sont terminées. Il ne reste alors que les topologies où les sources sont très proches les unes des autres, quasiment alignées, d'où la longueur particulièrement importante de la partie fusionnée. Ces topologies survivent plus longtemps car l'augmentation de la charge utile des paquets rendue possible par l'agrégation y est maximisée.

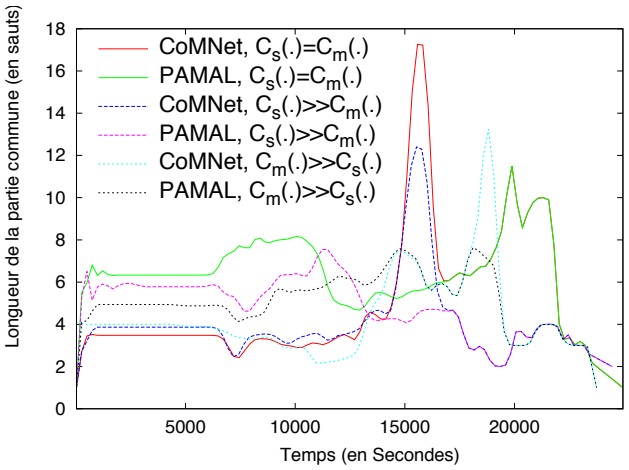
Par rapport au degré moyen des nœuds, les résultats montrent que les performances entre PAMAL et CoMNet sont variables selon le degré des topologies réseaux. PAMAL présente ainsi une fusion des chemins plus longue que CoMNet, mais cette longueur varie selon le degré. Ce phénomène s'explique par le CDS sous-jacent qui maintient la connexité du réseau. L'écart entre PAMAL et CoMNet est minimal quand le degré moyen est bas ($\delta = 15$) : la plupart des nœuds doit rester immobile pour maintenir la connexité du réseau. On observe une nette amélioration des résultats de PAMAL quand on atteint un degré moyen de $\delta = 20$. À ce niveau de degré, le CDS ne recouvre plus la totalité des nœuds du réseau, la liberté de mouvement est réellement présente pour une partie des nœuds du réseau. Cette liberté est toujours présente quand le degré est de $\delta = 25$, mais le nombre de chemins de routage possible augmente aussi. Les intersections des chemins se font donc plus près de la destination que dans les cas précédents. Dès lors, la fusion des chemins de routage est plus lente.

Par rapport à la taille des paquets, on observe que la fluctuation de la fusion est plus élevée avec des paquets plus lourds. C'est logique car plus les paquets sont gros, plus importante est l'énergie nécessaire pour leur transmission. Conséquemment, les nœuds meurent plus vite et la topologie évolue elle aussi plus vite.

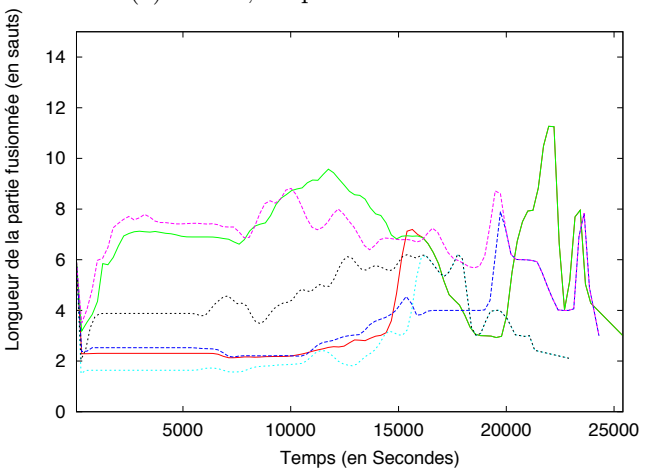
Les différents modèles de coût ont un impact plus marqué sur le comportement des simulations. Quand $C_{radio} < C_{mvt}$ ou $C_{radio} = C_{mvt}$, la quantité de mouvement dans les simulations augmente, la fusion des chemins peut donc se faire si le degré moyen de la topologie est suffisant ($\delta = 20$). Quand le coût de déplacement est élevé ($C_{mvt} > C_{radio}$), PAMAL présente toujours des chemins fusionnés plus longs, même si l'écart diminue car la mobilité est plus chère et donc moins employée. Conséquemment, il y a une convergence moins efficace des chemins de routage.

4.6 Conclusion et perspectives

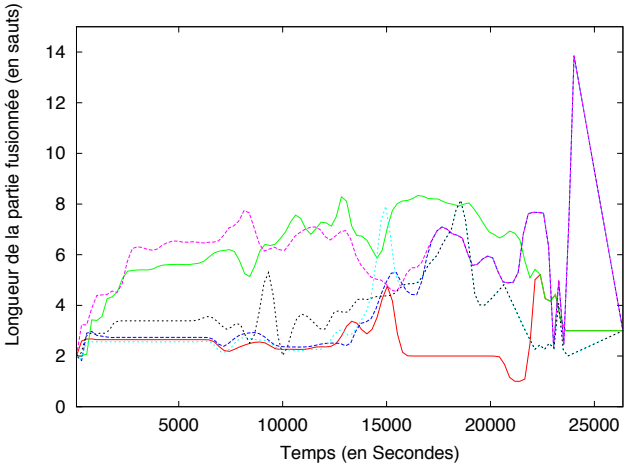
Dans ce chapitre, nous avons proposé le protocole de routage PAMAL, pour *Path Merging Algorithm*. PAMAL est un protocole de routage pour réseau de capteurs et



(a) $\delta = 15$, Paquets de 200 octets.

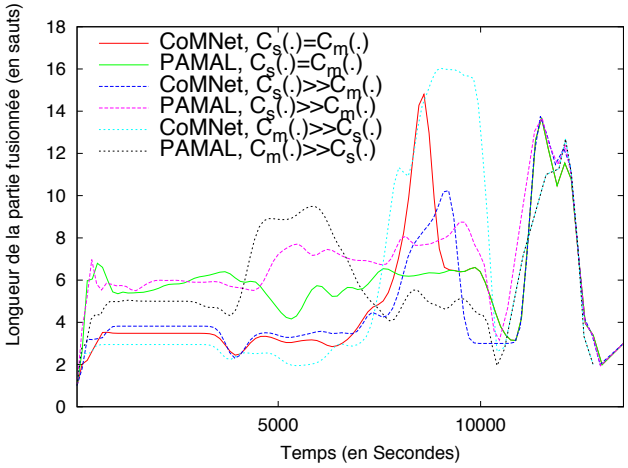


(b) $\delta = 20$, Paquets de 200 octets.

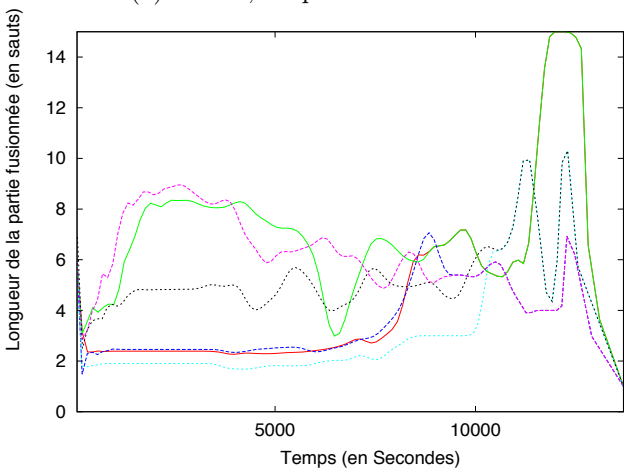


(c) $\delta = 25$, Paquets de 200 octets.

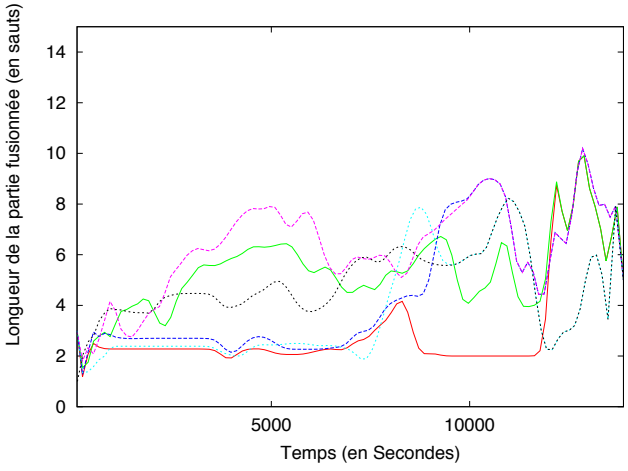
FIGURE 4.7 – PAMAL : longueur des chemins fusionnés. Paquets de 200 octets.



(a) $\delta = 15$, Paquets de 400 octets.



(b) $\delta = 20$, Paquets de 400 octets.



(c) $\delta = 25$, Paquets de 400 octets.

FIGURE 4.8 – PAMAL : longueur des chemins fusionnés. Paquets de 400 octets.

actionneurs qui tire parti de la mobilité contrôlée des actionneurs pour optimiser la topologie du réseau d'un point de vue énergétique. Grâce à l'intégration d'un mécanisme de détection des intersections, il permet de mettre un terme aux oscillations des nœuds appartenant à plusieurs chemins de routage. Il provoque ensuite la fusion de ces différents chemins, la longueur fusionnée allant croissant au cours du temps, tout en y réalisant une agrégation de paquets. De plus, PAMAL reste purement local, et donc adapté à de multiples topologies réseau et passe à l'échelle. Nos résultats montrent que PAMAL améliore significativement la durée de vie du réseau. En effet, la durée de vie moyenne d'un réseau de capteurs avec deux sources émettant vers une même destination augmente de 37% grâce à la gestion des intersections. Les résultats confirment une fusion des chemins de routage, la partie fusionnée étant de plus en plus longue au fur et à mesure des routages. C'est une amélioration significative par rapport aux protocoles existants.

Dans le futur, il serait intéressant de poursuivre le travail commencé en ne se limitant plus au cas de deux sources pour une destination. En effet, il est courant que les capteurs mesurent différents paramètres de l'environnement, les mesures devant être routées vers des destinations différentes selon leur type. Il s'agit alors d'envisager une approche permettant plusieurs sources et plusieurs destinations à la fois. La mobilité pourrait alors être employée pour créer physiquement un arbre de Steiner approché entre les différents sources et destinations.
De même, nous ne nous sommes intéressés ici qu'à une agrégation de données des plus simples, indépendante de tout contexte précis et donc très générale. Nous pensons qu'il serait intéressant d'étudier notre proposition avec d'autres modèles d'agrégation pour maximiser l'efficience de la fusion des chemins de routage. Enfin, nous souhaiterions explorer la possibilité d'informer les sources de l'existence de l'intersection afin d'agir dès l'origine des messages sur l'adaptation des chemins de routage.

Chapitre 5

Garantie de délivrance

5.1 Introduction

Notre objectif est de tirer parti de la possibilité de mobilité contrôlée pour améliorer le routage géographique dans les réseaux d'actioneurs. Comme on a pu le voir dans les chapitres précédents, l'introduction de la mobilité contrôlée dans les réseaux de capteurs présente de nombreux avantages. Afin de permettre des économies d'énergie, on a modifié la topologie localement pour réduire les coûts de transmission radio entre les différents nœuds impliqués dans le routage. Le routage est donc plus efficace en termes d'énergie car cette mobilité locale permet d'adapter au mieux la topologie du réseau à son trafic et de faire aussi bien localement que globalement, des économies d'énergie au cours des routages successifs.

Cependant, les approches de routage envisagées jusqu'ici ne visaient justement qu'à altérer cette topologie pour permettre un routage plus performant. Les chemins de routage utilisés sont en quelque sorte déjà existants. La mobilité n'y intervient qu'en *second* aspect. Cela est dû au fait que ces protocoles s'en tiennent à une approche gloutonne rendue possible par les informations géographiques.

Dans le routage géographique chaque nœud connaît sa position grâce à un dispositif de localisation. Via un échange de messages de type *Hello*, chaque nœud peut aussi connaître celle de ses nœuds voisins. C'est en se basant sur cette information et sur la position du voisinage que la plupart des routages géographiques proposent une approche de routage glouton. Ces mécanismes gloutons vont provoquer à chaque étape du routage le transfert du paquet du nœud courant à un de ses voisins situés plus près de la destination. De cette manière, la distance euclidienne entre le paquet et la destination est réduite à chaque transfert.

La sélection du prochain nœud est différente selon les protocoles. Toutefois, cette sélection se fait toujours dans le même sous-ensemble des voisins du nœud courant : parmi ses voisins qui sont plus proches de la destination que lui. On comprend dès lors que cette approche gloutonne échoue quand le prochain nœud n'a aucun voisin plus proche de la destination que lui. Le nœud courant constitue alors ce qu'on appelle un *extremum* local.

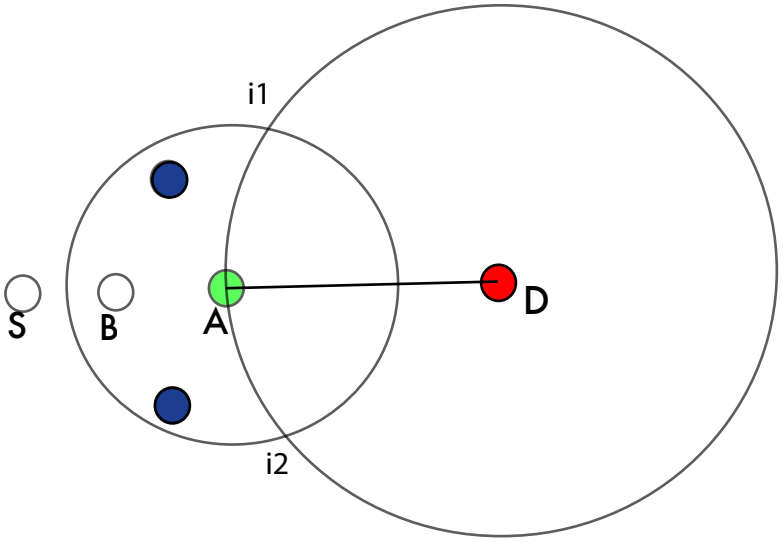


FIGURE 5.1 – Routage géographique glouton de S vers D. Échec du routage en A.

C'est sur cette constatation que nous avons proposé un nouveau protocole de routage dénommé *Greedy Routing Recovery*, ou GRR. L'idée est qu'en plus d'adapter un chemin de routage existant, on peut tout aussi bien utiliser la mobilité contrôlée pour contourner de petits obstacles. Si le routage glouton échoue, c'est que le nœud courant est sur un *extremum* local. Quand le routage glouton échoue, notre approche consiste à considérer l'ensemble des voisins du nœud courant. Fort de ce voisinage, le nœud courant va alors pouvoir créer un chemin glouton de telle manière qu'au routage suivant le routage glouton pourra emprunter sans échec le contournement ainsi créé.

5.2 La garantie de livraison dans la littérature

La problématique de garantir la bonne livraison des messages dans les protocoles de routage pour réseau de capteurs sans fil a fait l'objet de nombreuses recherches. De nombreux protocoles et algorithmes de routage ont été proposés dans la littérature mais on peut y distinguer trois catégories principales : le routage sur graphe planaire, le routage avec des arbres, et le routage heuristique de contournement d'obstacles.

5.2.1 Routage sur graphe planaire

Le protocole FACE [BMSU99] utilise le graphe planaire du réseau pour garantir la délivrance des messages. Avec FACE, le message à router parcourt la périphérie des « faces »traversées par la droite (SD). Le message progresse ainsi de face en face vers la destination. Toutefois, la longueur des chemins de routage calculés par cette approche augmente de manière significative avec le degré moyen du graphe.

Le protocole GFG [BMSU01], pour *Greedy Face Greedy*, combine une approche gloutonne avec un routage sur face. L'utilisation des faces ne se fait que lorsque le routage glouton tombe dans une impasse. Le routage sur face est donc un mécanisme de « récupération »en cas d'échec de l'approche gloutonne. Le protocole repasse en mode glouton dès lors que l'impasse est dépassée et que la position des différents voisins du nœud courant le permet. L'utilisation du routage sur faces est alors réduite, et donc les chemins de routage sont donc plus courts.

Ces approches reposent fortement sur les hypothèses du disque unitaire et nécessitent le graphe planaire du réseau qui sont des hypothèses difficiles à réaliser dans des environnements réalistes. De plus, l'introduction de la mobilité dans les graphes planaires apporterait une contrainte supplémentaire à la liberté de mouvement des nœuds.

5.2.2 Routage avec des arbres

Les réseaux étant assimilables à des graphes, une autre approche dans la littérature consiste à utiliser des arbres et à router les paquets dans ceux-ci.
GDSTR [LLM06], pour *Greedy Distributed Spanning Tree Routing*, est un protocole de routage géographique combinant une approche gloutonne à un mécanisme de récupération reposant sur les arbres quand le routage glouton échoue. Ce mécanisme de récupération ne repose ni sur l'hypothèse du disque unitaire (voir l'Introduction), ni sur la nécessité de construire le graphe planaire de son voisinage. A l'opposé, la récupération dans ce protocole repose sur une nouvelle variante des arbres couvrants que les auteurs appellent « hull trees ». Un arbre couvrant est, pour un graphe, un arbre inclut dans ce graphe et qui contient tous les nœuds de celui-ci dans sa structure.
Quand le routage glouton échoue, le protocole va alors router le paquet suivant les arbres jusqu'à atteindre la destination. Dans les *hull trees* chaque nœud connaît tous ses enfants et leur position associée. Ainsi, dès que le routage glouton échoue, le nœud courant parcourt les sous-arbres de ses enfants afin de déterminer un chemin jusqu'à la destination. Si la destination appartient à ses descendants, le paquet est transféré au premier nœud parmi ses fils sur le chemin jusqu'à la destination. Sinon le paquet est transféré de manière à remonter dans l'arbre jusqu'à ce qu'un nœud possède un chemin jusqu'à la destination via un de ses fils.
Pour éviter que l'arbre ne s' effondre avec la mort du nœud racine, les auteurs proposent de construire plusieurs *hull trees* au démarrage du réseau. Les racines de ces différents arbres doivent être les plus distantes possible pour améliorer l'efficacité du mécanisme de récupération.

Cette approche est inadaptée aux réseaux de capteurs même statiques. En effet, elle nécessite que chaque nœud possède en mémoire les arbres couvrants de tous ses enfants ainsi que leur position, et ce pour au moins deux racines différentes. La consommation mémoire de cette approche est donc très élevée. Mais cette approche est aussi inadaptée aux réseaux d'actionneurs : la récupération repose sur des arbres. Or on se propose d'adapter la topologie du réseau à son trafic et donc de faire se déplacer des nœuds.

On risque de casser les arbres créés au démarrage du réseau, ou de limiter fortement les possibilités de mouvement pour éviter de devoir recalculer les différents arbres.

5.2.3 Routage heuristique

Un exemple d'une approche qu'on qualifie d'heuristique est présenté dans [FGG06]. C'est une approche distribuée en deux étapes. Ces deux étapes se déroulent consécutivement au démarrage du réseau.

La première étape consiste à détecter localement les nœuds sur lesquels le routage glouton pourrait échouer. Pour ce faire, chaque nœud exécute un algorithme appelé « la règle de Tent »et détermine, localement, uniquement avec sa position et celle de ses voisins, les possibles *extremum* locaux. La deuxième étape est réalisée via un algorithme distribué appelé *Bound Hole*. Il va faire en sorte que chaque *extremum* local identifié par la règle de Tent recherche son prédécesseur et son successeur sur la périphérie du trou.

Conséquemment, quand le routage glouton échoue le paquet est alors routé sur la périphérie du trou. Cette approche ne nécessite donc que des informations sur les voisins immédiats. De plus les *extremum*s locaux sont clairement identifiés et peuvent bouger tant qu'ils restent connectés à leur prédécesseurs et successeurs. Toutefois, elle requiert un nombre important de messages au démarrage du réseau mais aussi en cas de changement de la topologie, autour des zones à contourner. De plus, à cause de la règle de Tent, fortement sur les hypothèses du graphe unitaire, qui, on l'a vu précédemment, sont difficilement réalistes.

Ainsi, aucune des différentes approches présentées dans cette section n'est directement transposable dans nos réseaux d'actionneurs. En effet, ces hypothèses reposent fortement sur la position des nœuds et ne supportent pas une évolution rapide des voisinages des nœuds comme l'implique la mobilité. C'est pourquoi nous proposons une toute nouvelle approche.

5.3 Contribution

Dans cette section, nous introduisons dans un premier temps notre algorithme de routage géographique de type glouton permettant le routage et l'adaptation de la topologie du réseau. Puis, dans un second temps, nous détaillons le mécanisme de récupération légère qui va permettre de construire un chemin de routage glouton quand il n'en existe pas. Enfin nous détaillons le mécanisme de routage global combinant les approches gloutonnes et de récupération légère.

5.3.1 Routage glouton

Le routage géographique glouton de GRR reprend celui de PAMAL. À chaque étape du routage, l'algorithme glouton de GRR sélectionne comme prochain nœud sur le

chemin de routage le nœud B, parmi les voisins du nœud courant dans la direction de la destination, qui résout l'équation suivante :

$$B = arg_{min_{K \in N_D(A)}} \mathrm{C_{radio}}(|AK|) + \mathrm{C_{mvt}}(|KK'|) \tag{5.1}$$

Le coût de la relocalisation de chacun de ses voisins est calculé en utilisant les trois différents schémas de mouvement introduits dans CoMNet [HMSR11] que sont :

— $CoMNet - ORouting\ on\ the\ Move$ qui aligne les nœuds du chemin de routage sur la droite virtuelle entre la source S et la destination D et notée (SD).

— $CoMNet - Move_{(DSr)}$ qui aligne les nœuds du chemin de routage sur (SD) en les positionnant de telle manière qu'ils soient espacés les uns les autres d'une distance égale à r^*, le rayon de communication optimal du médium radio employé.

— $CoMNet - Move_r$ qui déplace un nœud B à l'intersection de (BD) et du cercle $C(A, r^*)$ centré en A et de rayon r^*.

L'algorithme de routage glouton est détaillé dans l'algorithme 6. Sa seule différence avec PAMAL est que le routage glouton se poursuit s'il existe un nœud au moins aussi près de la destination que le nœud courant. Ce sous ensemble de $N(A)$ est noté $N\prime_D(A)$.

Algorithm 6 routageGlouton(A,D) - Éxecuté par le nœud courant A, qui route un paquet ayant pour destination D.

1: $prochain \leftarrow -1$; ▷ prochain saut
2: $prochain\prime \leftarrow \{0,0,0\}$; ▷ nouvelle position du prochain saut
3: $minCOP \leftarrow +\infty$ ▷ initialisation de la valeur du COP de référence
4: **for all** $\{B \in N\prime_D(A)\}$ **do**
5: $B\prime_{or}, B\prime_{mr}, B\prime_{mdsr} \leftarrow \{0,0,0\}$;
6: $B\prime_{or} \leftarrow \mathrm{ORouting}(A, B, D)$;
7: $B\prime_{mr} \leftarrow \mathrm{Move_R}(A, B, D)$;
8: $B\prime_{mdsr} \leftarrow \mathrm{Move_{DSR}}(A, B, D)$;
9: $bCOP \leftarrow min[COP(A, B, B\prime_{or}), COP(A, B, B\prime_{mr}), COP(A, B, B\prime_{mdsr}]$;
10: **if** $(bCOP < minCOP)$ **then** ▷ Comparaison avec la valeur de référence
11: $prochain \leftarrow B$; ▷ M.à.j. du prochain saut
12: $prochain\prime \leftarrow B\prime$; ▷ Sauvegarde de la position associée
13: $minCOP \leftarrow bCOP$;
14: **end if**
15: **end forreturn** $prochain, prochain\prime$ ▷ Retourne le prochain saut avec sa relocalisation

Toutefois, et comme on l'a vu, le routage glouton échoue dès lors que le nœud courant est un *extremum* local. En d'autres termes, si le nœud courant ne possède pas de voisin plus proche de la destination que lui-même, alors le routage glouton échoue.
Nous nous proposons donc de modifier la topologie du réseau autour de cet *extremum* local afin qu'il n'en soit plus un, et que le routage glouton puisse se poursuivre.

5.3.2 Récupération légère

Le mécanisme de récupération légère de GRR prend le relais dès lors que le routage glouton n'est plus possible. Comme on l'a vu, le routage glouton échoue dès lors que le sous-ensemble du voisinage du nœud courant constitué par les nœuds situés plus près de la destination que lui même est vide $N_D(.)$. Le principe est d'utiliser un nouveau schéma de déplacement de récupération pour créer un contournement de l'obstacle autour de la zone à circonvenir et vers la destination de telle sorte que le routage glouton puisse reprendre dessus.

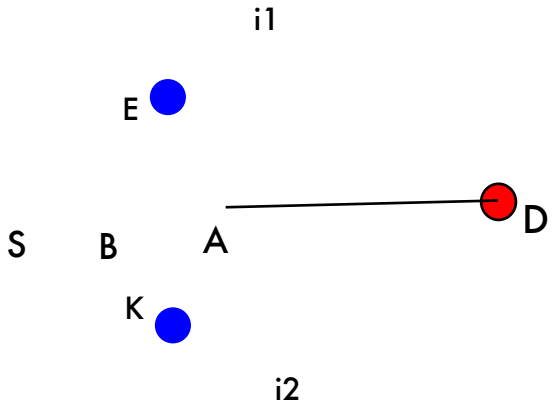


FIGURE 5.2 – Routage géographique glouton de S vers D. Échec en A. Points de contournement en $i1$ et $i2$.

En effet, comme on le voit sur la Figure 5.2, aucun des schémas de déplacement de CoMNet ne peut être appliqué avec succès par le nœud courant A. Son voisinage vers la destination $N_D(A)$ étant vide, il ne peut plus faire progresser le paquet dans sa direction. Toutefois, il est possible de restaurer le routage glouton, en contournant l'obstacle, sur un saut, si on déplace un des nœuds du voisinage de A, appartenant donc à $N_(A)$, de façon à ce qu'il appartienne à $N_D(A)$. Tous les nœuds de $N(A)$ sont susceptibles d'être utilisés, et donc déplacés, pour construire cette nouvelle route sauf son prédécesseur sur le chemin de routage. Nous empêchons le déplacement du nœud précédent car le but est de construire un chemin en aval du nœud courant, pas de détruire le chemin existant en amont. La connexité est garantie à chaque instant car le nœud qui se déplace le fait en restant à portée du CDS du réseau, ou en étant à portée de ses voisins RNG.

Si le routage a échoué, c'est que le nœud courant se trouve face à un obstacle ou à une zone de densité trop faible pour permettre un routage. Nous n'avons aucune information sur cette zone, si ce n'est la distance entre le nœud courant et la destination. Nous nous proposons alors de contourner cet obstacle, et de permettre de plus un routage glouton

sur ce contournement.

Il s'agit donc de maximiser le contournement de l'obstacle, tout en permettant une approche gloutonne pour les routages suivants. Pour cela nous relocalisons un des voisins du nœud courant. Nous proposons d'utiliser les points particuliers que sont les points d'intersection entre les cercles $C_1(A, |r^*|)$ et $C_2(D, |AD|)$ et qu'on appellera des points de contournement. Ces points de contournement sont notés i_1 et i_2 comme indiqué sur la figure 5.2. Ces points de contournement reposent sur l'hypothèse du disque unitaire.

En ordonnant le déplacement d'un de ses voisins sur un de ces points particuliers, le nœud courant va pouvoir, au prochain routage, avoir un chemin de routage glouton, car la distance entre chacun de ces points et la destination est égale à celle entre le nœud courant et la destination. Mais ces points de contournement permettent aussi d'optimiser le coût de radio transmission car la distance entre le nœud courant et ces points de contournement correspond au rayon optimal de radio transmission. L'utilisation de $N\prime_D(.)$ au lieu de $N_D(.)$ permet ici d'éviter les boucles.

Le point $i1$ contourne l'obstacle « par le haut », alors que le point $i2$ le contourne « par le bas ». Afin d'éviter des boucles, consommatrices d'énergie, nous limitons la possibilité de changer le sens du contournement. Ainsi, lorsqu'un nœud passe un paquet en récupération légère, il sélectionne comme prochain saut son voisin (hors nœud précédent) de sorte à minimiser le coût total pour rejoindre un des points particuliers :

$$B = arg_{min_{K \in N(A)}} \mathrm{C_{radio}}(|AK|) + \mathrm{C_{mvt}}(|\mathrm{Ki'_x}|) \tag{5.2}$$

Ce coût total est calculé pour chaque voisin du nœud courant, avec chacun des deux points particuliers comme relocalisation considérée. La transition entre le routage glouton et la récupération légère détermine aussi le sens du contournement.

Si le nœud qui effectue la transition calcule que le premier routage de la récupération légère de ce paquet doit se faire en passant par i_1, et donc qu'on contourne l'obstacle par le bas, les contournements suivants se feront aussi par le bas. Le sens de contournement, haut ou bas, en utilisant $i1$ ou $i2$, est donc défini lors de l'entrée en récupération légère. Cette contrainte s'explique car il s'agit d'éviter des routages qui bouclent autour de l'obstacle et entre les points de contournement.
Ce sens peut être changé une fois, et une seule, si la récupération légère venait à être bloquée (voisinage du nœud courant– hors nœud précédent – vide), afin de tester le contournement dans l'autre sens. Si la récupération légère se bloque à nouveau, le paquet est supprimé et le routage échoue.

En terme d'implémentation, stocker le type de routage (glouton ou récupération) avec l'orientation associée, ne nécessite en tout et pour tout que deux bits.

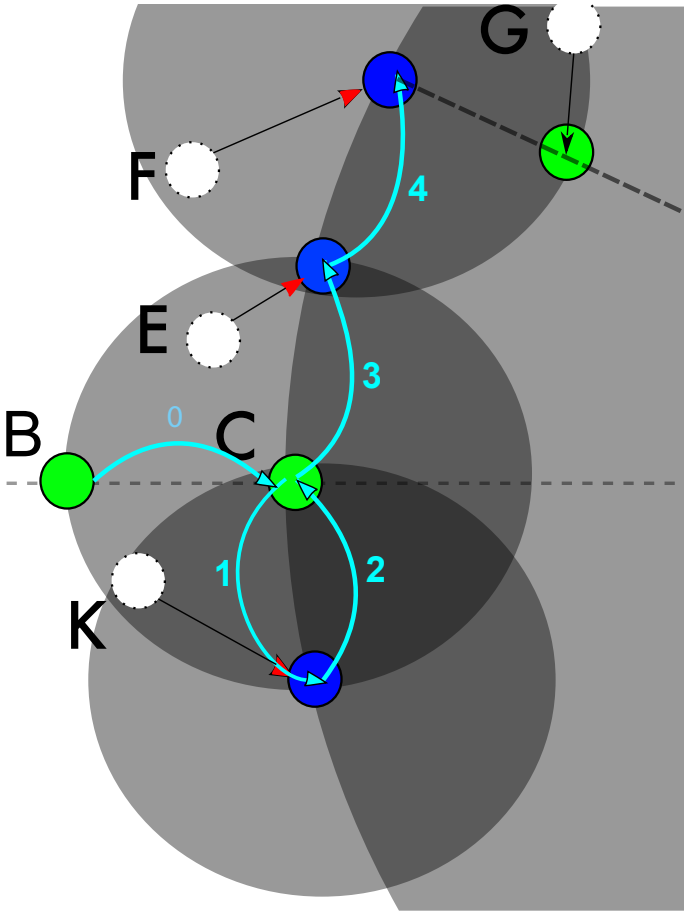


FIGURE 5.3 – Échec du routage glouton en C. Récupération légère en passant successivement par (et en les relocalisant) K , C, E, F puis reprise du glouton en G

La figure 5.3 illustre un exemple du routage en récupération légère. Un routage glouton est effectué de B à C. Le nœud C ne peut lui plus poursuivre avec cet algorithme. Il passe alors le paquet en récupération légère (et change le bit de routage associé). Le nœud courant calcule alors ses points de contournement. Ensuite, pour chacun de ses voisins K et E, il calcule le coût total de leur relocalisation en $i1$ et $i2$. C'est le voisin K qui minimise ce coût si on le relocalise en $i2$. Le contournement se fera donc par le bas. Le paquet est alors transmis à K assorti d'un ordre de déplacement en $i2$ (ordre qui ne nécessite qu'un bit). K se déplace donc jusqu'à $i2$.
La récupération est alors bloquée car il n'existe pas de voisin. L'orientation de la récupération est donc changée, et le contournement reprend par le haut en passant par C puis F. En F, la récupération légère est abandonnée car le routage glouton peut reprendre pour transférer le paquet à G, etc...

L'algorithme 7 détaille le fonctionnement de la récupération légère. Quand un nœud gère un paquet en récupération légère, il commence par vérifier si son voisinage n'est

Algorithm 7 routageRLegere(C,D,P) - Éxecuté par C quand $N\prime_D(C)$ est vide. D la destination, P le paquet a router.

1: $prochain \leftarrow -1$; ▷ Prochain saut dans le routage
2: $prochain\prime \leftarrow \{0,0,0\}$; ▷ Relocalisation du prochain saut
3: $minCOST \leftarrow +\infty$; ▷ lowest total cost computed over all $N_D(A)$
4: $(i_1, i_2) \leftarrow intersections(C_1(A, |r^*|), C_2(D, |AD|))$;
5: **if** $(N(A) - noeudPrecent = \emptyset) ET (P- > orientationLocked == FAUX)$ **then**
6: **return** -1 ; ▷ Le routage échoue
7: **else if** $(N(A) - noeudPrecent = \emptyset) AND (P- > orientationLocked == FAUX)$ **then**
8: $prochain = noeudPrecent$; $prochain\prime = position(noeudPrecent)$;
9: $P- > orientationLocked \leftarrow VRAI$; ▷ Orientation changée et verrouillée
10: **return** $prochain, prochain'$; ▷ Paquet renvoyé au précédent
11: **end if**
12: **if** $(P- > orientation = NULL)$ **then** ▷ Si 1er passage en récupération légère
13: **for all** $\{K \in (N_D(C) - noeudPrecedent)\}$ **do** ▷ Pour chacun des voisins K
14: $tCOST \leftarrow \mathrm{C_{radio}}(|AK|) + \mathrm{C_{mvt}}(|Ki_1|)$; ▷ calcul du cout de reloc. en i_1
15: $tCOST2 \leftarrow \mathrm{C_{radio}}(|AK|) + \mathrm{C_{mvt}}(|Ki_2|)$ ▷ calcul du cout de reloc. en i_2
16: **if** $(tCOST1 < minCOST)$ **then**
17: $prochain \leftarrow K$; $prochain\prime \leftarrow i_1$;
18: $P- > orientation \leftarrow i_1$;
19: $minCOST \leftarrow tCOST1$
20: **end if**
21: **if** $(tCOST2 < minCOST)$ **then**
22: $prochain \leftarrow K$; $prochain\prime \leftarrow i_2$;
23: $P- > orientation \leftarrow i_2$;
24: $minCOST \leftarrow tCOST2$
25: **end if**
26: **end for**
27: **else** ▷ Si poursuite d'un récupération légère lancée précédemment
28: $i_x \leftarrow 0$;
29: **if** (P->orientation = UP) **then** $i_x = i_1$ ▷ On garde l'orientation
30: **else** $i_x = i_2$
31: **end if**
32: **for all** $\{K \in (N_D(C) - noeudPrecedent)\}$ **do**
33: $tCOST \leftarrow \mathrm{C_{radio}}(|AK|) + \mathrm{C_{mvt}}(|Ki_x|)$;
34: **if** $(tCOST < minCOST)$ **then**
35: $prochain \leftarrow K$; $prochain\prime \leftarrow i_x$;
36: $P- > orientation \leftarrow i_x$;
37: $minCOST \leftarrow tCOST2$
38: **end if**
39: **end for**
40: **end if**
41: **return** $prochain, prochain'$ ▷ Retourne le prochain saut et sa nouvelle position

pas vide.
Dans le cas où son voisinage est vide, et que la récupération légère a déjà changé d'orientation dans un routage précédent pour cette même récupération (Alg. 7.5), le paquet est supprimé. Sinon, l'orientation est changée et le paquet est retransmis au nœud précédent afin de reprendre le contournement dans l'autre sens (Alg. 7.7).
Dans le cas où son voisinage n'est pas vide et que c'est le prochain saut le premier en récupération (pas d'orientation fixée), il calcule ses points particuliers et retient comme prochain saut le voisin qui minimise le coût total. L'orientation est aussi fixée (Alg. 7.12). Par contre, si l'orientation était déjà fixée, le sens du routage est maintenu (Alg. 7.27).

Le détail des transitions et passages entre routage glouton et récupération légère est décrit dans la section suivante.

5.3.3 L'algorithme de routage

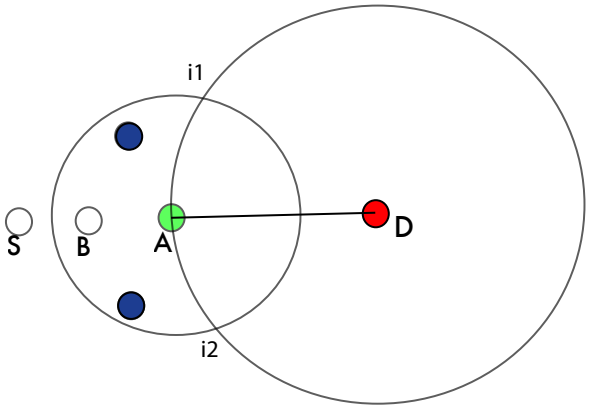


FIGURE 5.4 – $N_D(A)$ est vide, le routage glouton vers D échoue A.

L'échec du routage glouton est dû à l'absence de voisins plus proche de la destination que le nœud courant. La figure 5.4 illustre une telle configuration. Le nœud courant Ane possède aucun voisin plus proche de la destination D qu'il ne l'est lui même. A représente un *extremum* local du chemin de routage vers la destination D, il fait face à un trou de couverture. Le routage glouton requiert au moins que A soit un voisin dans la zone définie par l'intersection de $C_1(A, r*)$ le disque centré en A et de rayon r*, et $C_2(D, |AD|)$ le disque centré en D et de rayon $|AD|$ pour réussir.

C'est pourquoi nous proposons le protocole *Greedy Routing Recovery* qui combine à la fois un routage glouton et un mécanisme de récupération dit légère. Cette récupération légère vise à modifier la topologie du réseau afin de rendre possible le routage glouton. L'algorithme 8 présente les transitions entre les différents paradigmes de routage.

Quand un nœud reçoit un paquet, il vérifie selon quel algorithme il a été routé lors de son dernier transfert grâce à ses méta-données (Alg 8.1). C'est à partir de cette information, et de la connaissance de la position de ses voisins que le choix de continuer

Algorithm 8 GRR(A,D, P) - Le noeud A a un paquet P à router pour D

```
1: switch P->routage do
2:     case routageGlouton
3:         if N_D(A) ≠ ∅ then
4:             prochain, prochain' ← routageGlouton(A, D, P) ;
5:         else
6:             P->recuperationLegere ;
7:             prochain, prochain' ← recuperationLegere(A, D, P) ;
8:         end if
9:         transfererPaquet(P, prochain, prochain')
10:        fin ;
11:    case routageRLegere
12:        if N_D(A) ≠ ∅ then
13:            P->routageGlouton ;
14:            prochain, prochain' ← routageGlouton(A, D) ;
15:        else
16:            prochain, prochain' ← recuperationLegere(A, D, P) ;
17:            if prochain == −1 then return NULL          ▷ Echec du routage
18:            end if
19:        end if
20:        transfererPaquet(P, prochain, prochain') ;
21:        fin ;
```

ou non, avec le même algorithme de routage est effectué.

Dans le cas où le paquet était routé selon l'algorithme glouton (Alg 8.2), et qu'il est toujours possible de le faire, le routage continue comme tel (Alg 8.4). En revanche, s'il n'est plus possible de continuer de cette manière (Alg 8.5), le paquet va alors être modifié pour être routé selon l'algorithme de récupération (Alg 7). Le nœud courant va alors déterminer lequel de ses voisins minimise le total des coûts (déplacement et transmission) pour rejoindre un des points de contournement. Les méta-données sont modifiées de façon à indiquer la position de l'entrée en récupération légère et le sens initialement choisi. Le paquet est alors transmis au voisin du nœud courant qui minimise la somme des coûts de transmission et de déplacement. Le paquet est finalement transmis (Alg 8.9).

Par contre, si le paquet que le nœud a reçu était routé selon l'algorithme de récupération (Alg 8.11), il va d'abord commencer par exécuter l'ordre de déplacement inclus. Ensuite, une fois arrivé à sa nouvelle position, le nœud va actualiser sa connaissance de son voisinage. S'il possède au moins un voisin plus proche de la destination que lui, et qui, en plus, est plus proche de la destination que la position où le paquet est entré en récupération, le nœud va alors repasser ce paquet en routage glouton (Alg 8.12). Sinon, le paquet reste routé selon l'algorithme de la récupération légère, si possible (Alg 8.16).

En effet, la récupération peut échouer s'il est impossible de continuer selon l'orientation courante alors que celle-ci a déjà été changée précédemment pour cette récupération (Alg 8.17). Dans ce cas le paquet est supprimé.

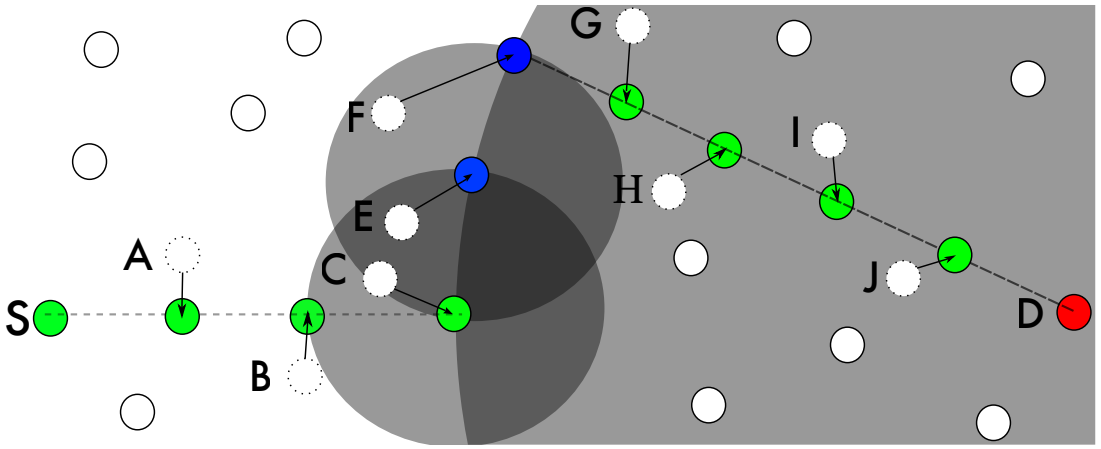


FIGURE 5.5 – Routage GRR de S à D.
Les nœuds de routage glouton en vert, les nœuds avec la récupération légère en bleu.

La figure 5.5 illustre un routage complet entre les nœuds S et D. Le routage est glouton sur trois sauts, et les nœuds A, B et C sont alignés sur la droite (SD) par l'algorithme de routage. Toutefois, en C, l'algorithme de routage doit passer le paquet en récupération légère car il n'existe pas de nœud plus proche de la destination D que C. Le nœud E est alors sélectionné : C lui transmet le paquet assorti d'un ordre de déplacement sur le point de contournement retenu. E va donc recevoir ce paquet dont les méta-données indiquent qu'il doit être routé selon la récupération légère. En tout premier lieu, E va exécuter l'ordre de déplacement contenu dans le paquet. Dans un second temps, E vérifie son voisinage en sa nouvelle position afin de vérifier si le routage peut éventuellement repartir en mode glouton. C'est toujours impossible dans notre exemple. Le nœud courant E va donc transférer le paquet en récupération légère à F avec l'ordre de se déplacer. Le routage glouton redevient possible après F car son voisin G est plus proche de la destination que lui, et aussi plus proche que C de cette même destination. Enfin le paquet est routé, de manière gloutonne, de H, I, J jusqu'à la destination D.

5.4 Simulations

Dans le cadre de notre démarche d'exploiter la mobilité de façon contrôlée nous avons souhaité tester une alternative à l'ensemble dominant connecté (*Connected Dominated Set* ou CDS). En effet, le CDS nécessite une topologie de densité suffisante pour rendre possible la mobilité. Un nouveau mécanisme de maintien de connexité du réseau basé sur le graphe des voisins relatifs a été implémenté.

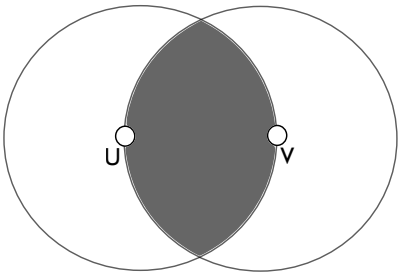


FIGURE 5.6 – Calcul du RNG. L'arrête (U, V) existe ssi la zone grise est vide

5.4.1 Graphe des voisins relatifs

Le graphe des voisins relatifs, ou *Relative Neighorhood graph* (RNG) [Tou80], est un graphe planaire qui est utilisé dans de multiples constructions topologiques. Un graphe planaire est un graphe dont aucune des arrêtes reliant les nœuds ne s'intersectent, sinon sur un nœud.

Le RNG peut être calculé localement comme suit. Une arête (U, V) du graphe du réseau est présente dans le sous-graphe du RNG si, et seulement si, elle n'est pas l'arrête la plus longue du triangle UVW où W est un des voisins de U. Dans la figure 5.6, l'arête (U, V) appartient au RNG s'il existe un nœud dans la zone grise. Pour chaque nœud W dans la zone grise, l'arête (U, V) est la plus longue arête du triangle UVW.

$$\forall W \neq U, V \in E : |UV| \geq max[|UW|, |VW|] \tag{5.3}$$

Une fois qu'un nœud du graphe a déterminé sa position et celle de ses voisins (via des messages *HELLO*), il peut déterminer pour chacune de ses arêtes s'il existe un autre voisin dans la région interdite. La construction du RNG est donc purement locale et ne nécessite aucun message additionnel. L'algorithme de calcul du RNG est résumé dans l'algorithme 9. Le degré moyen d'un nœud dans le RNG est d'environ 2,5 [Tou80].

Algorithm 9 Éxecuté sur chaque nœud du réseau U

```
1: for all V ∈ N(U) do
2:     for all W ∈ N(U) do
3:         if W == V then
4:             continue
5:         else if |UV| > max[|UW|, |VW|] then
6:             supprimer (U, V) ;
7:             fin ;
8:         end if
9:     end for
10: end for
```

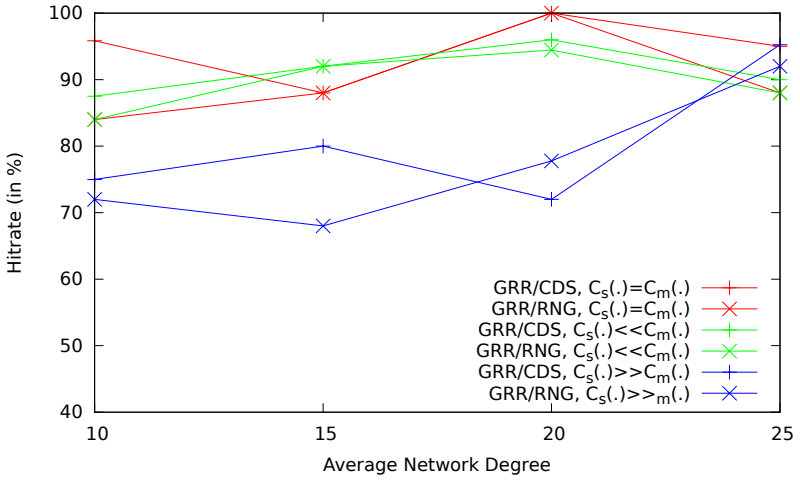


FIGURE 5.7 – Pourcentage de topologies où GRR route avec succès en fonction de δ

Ainsi on a un nouveau mécanisme de maintien de la connexité purement local. Si un nœud qui se déplace reste connecté à ses voisins RNG, il n'altère pas la connexité globale du réseau et on garantit qu'à tout instant il existe un chemin entre deux nœuds du réseau [RSR11]. Ce mécanisme est très différent du principe du CDS qui autorise, lui, la mobilité des nœuds dominés tant qu'il reste à porté d'un nœud dominant, appartenant au CDS.

5.4.2 Taux de réussite

La figure 5.7 représente le pourcentage de topologies sur lesquelles le routage réussit avec GRR selon plusieurs degrés réseau moyen et plusieurs modèles de coûts.
Les résultats sont très similaires que l'on utilise GRR avec le CDS ou le RNG, et ce quel que soit le degré moyen. Quand le degré est faible, la plupart des nœuds appartiennent au CDS et donc la mobilité est limitée. De même, si l'on considère le RNG, les voisins RNG sont d'autant plus éloignés que la densité du réseau est faible. Quand le degré du réseau augmente, le pourcentage de topologies sur lesquelles GRR réussit augmente étant donné qu'il y a de plus en plus de nœuds libres de mouvement pour GRR-CDS. Parallèlement, on observe que pour GRR-RNG, le nombre de voisins augmentant, les voisins RNG sont donc plus près, et donc la liberté de mouvement augmente elle aussi.

Par rapport au modèle de coût, nous notons que le pourcentage de réussite est maximal quand le modèle de coût est $C_{radio}(.) == C_{mvt}(.)$. Si on appelle A le nœud courant, ce modèle de coût fait que tous les nœuds de $N(A)$ sont considérés de façon égale.
Le taux de succès est minimal quand $C_{radio}(.) >> C_{mvt}(.)$ car ce modèle provoque la sélection de nœuds qui sont proches de A afin de réduire les coûts de transmission. Cependant, ces nœuds ont une longue distance à parcourir pour rejoindre le point de contournement et restaurer le routage glouton.Or, la probabilité que le nœud sélectionné

rejoigne cette position est d'autant plus faible que le nœud en est éloigné car le mécanisme de maintien de la connexité du réseau peut l'en empêcher.
Le cas moyen est observé quand $C_{radio}(.) << C_{mvt}(.)$: A sélectionne alors un nœud proche d'un point de contournement i_x.

5.4.3 Nombre et longueur moyens des passages en récupération

La figure 5.8 représente le nombre moyen de transitions entre le routage glouton et le mécanisme de récupération en fonction du temps. Les résultats montrent qu'au démarrage du réseau le nombre de récupérations tend à diminuer fortement à cause de la construction du contournement de notre approche. Dans un second temps, ces nœuds fortement sollicités aussi bien en termes de déplacement qu'en terme de transmission radio, meurent. Le nombre de passages en récupération légère augmente alors vu que GRR tente de reconstruire les routes. Toutefois il est de plus en plus difficile de déplacer les nœuds, ce qui explique que le nombre de passages en récupération ne diminue plus.

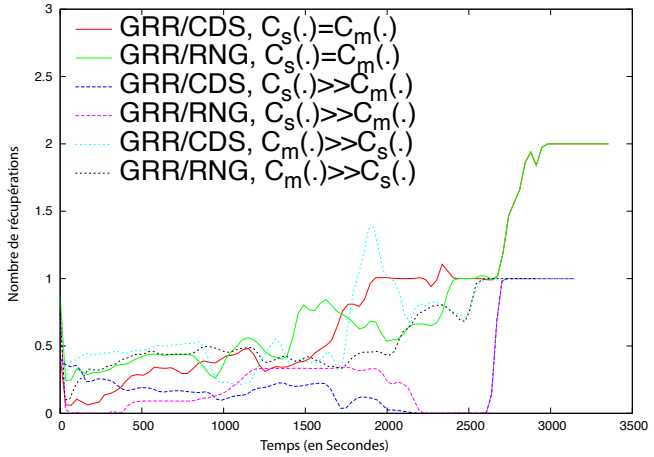
Cette analyse est confirmée par la figure 5.9 qui représente la longueur moyenne de la partie récupération. Après une baisse initiale, le nombre de sauts se stabilise avant d'augmenter à nouveau. En effet, le trou dans la topologie est contourné par GRR. Mais GRR doit faire face à un trou de routage toujours plus grand au fur et à mesure du temps et de la mort des nœuds sur sa périphérie. Il est à noter que le routage n'aboutirait en aucun cas sans la récupération légère de GRR.

5.4.4 Évolution du coût du routage

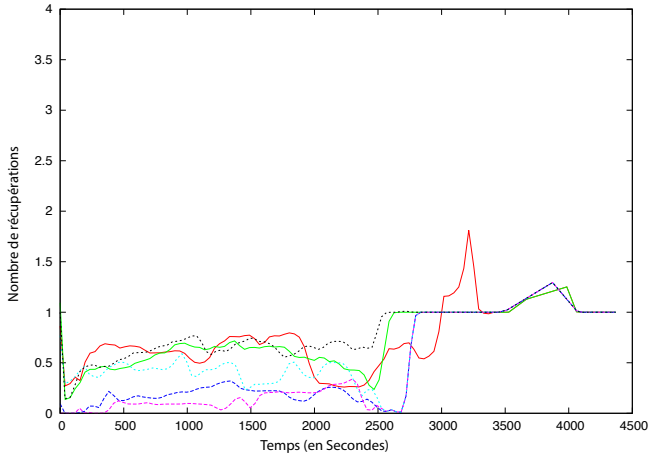
La figure 5.10 représente le coût énergétique total du routage d'un paquet mené jusqu'à destination en fonction du temps. Ce coût total intègre les coûts de transmission radio entre les nœuds, auxquels s'additionnent les coûts de déplacement des nœuds pour adapter la topologie. Les résultats montrent un pic initial, indépendamment du modèle de coût, du degré moyen du réseau... Ce phénomène s'explique par la construction initiale du chemin de routage glouton. Sur ces topologies, le routage purement glouton échoue. GRR fait donc appel à son mécanisme de récupération légère afin de les contourner, mais aussi de construire une nouvelle route. Cette construction signifie un déplacement physique d'un nombre important de nœuds sur le chemin de routage, d'où le surcoût initial du routage. Dans un second temps, le coût des paquets tend à se stabiliser. Cela s'explique parce que les contournement nécessaires ont été créé précédemment et sont alors utilisés. Enfin, le coût du routage augmente de façon significative en fin de simulation. GRR doit en effet faire face à des trous de couverture de plus en plus grands.

5.5 Conclusion

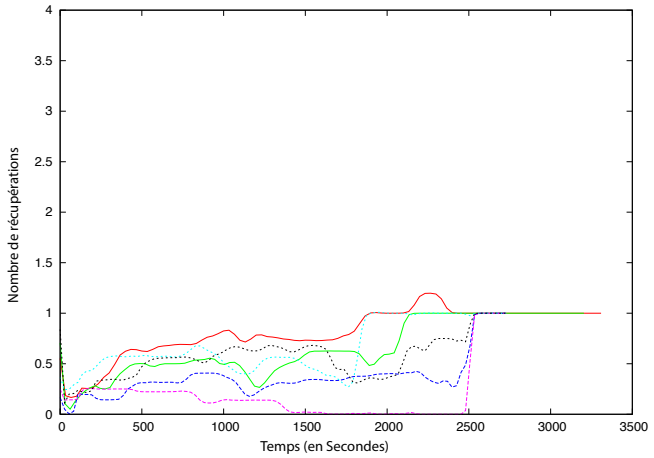
Dans ce chapitre, nous avons proposé un protocole de routage géographique pouvant s'appliquer sur notre hypothèse d'un réseau d'actionneurs. Notre approche tire parti de la mobilité contrôlée afin de contourner au mieux localement les obstacles pour restaurer



(a) $\delta = 10$

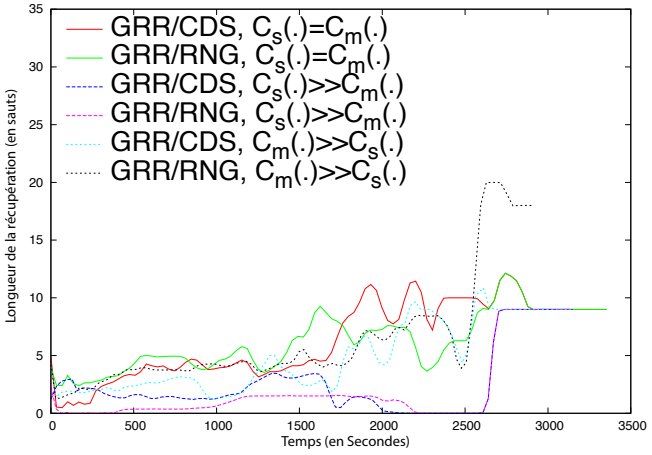


(b) $\delta = 15$

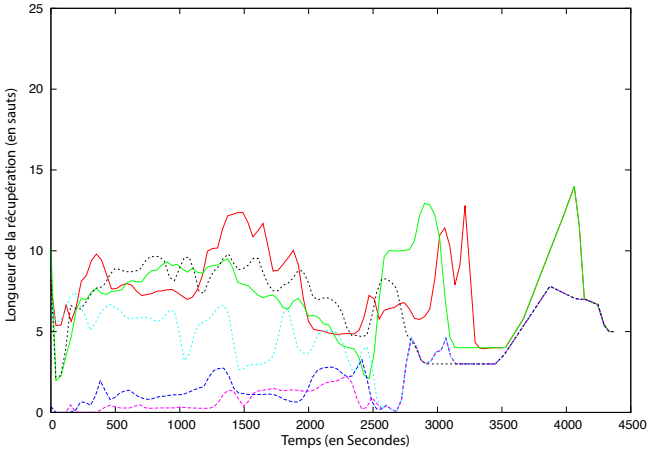


(c) $\delta = 20$

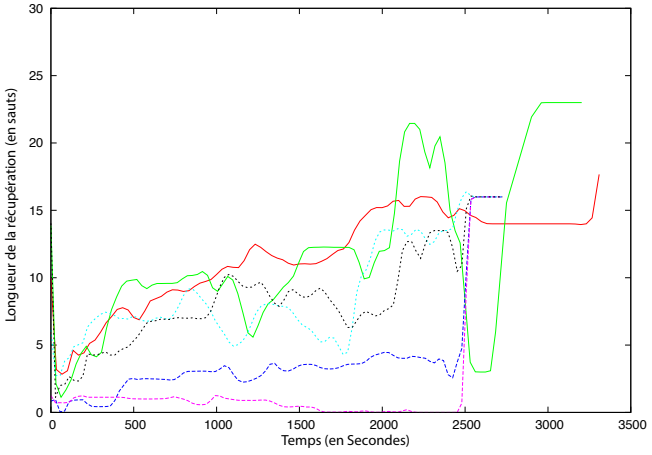
FIGURE 5.8 – Nombre moyen de passages en récupération en fonction du temps.



(a) $\delta = 10$

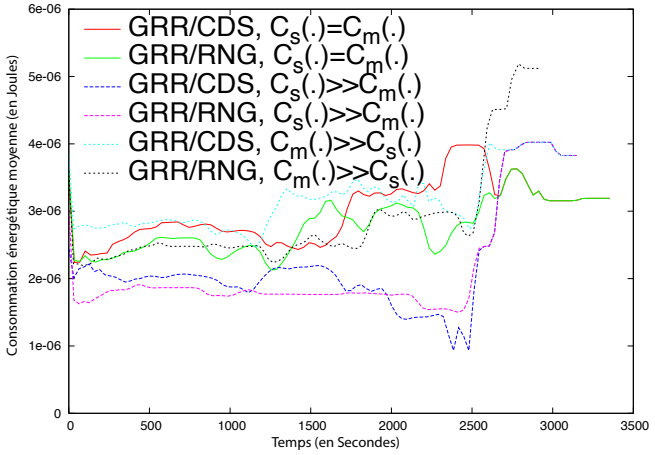


(b) $\delta = 15$

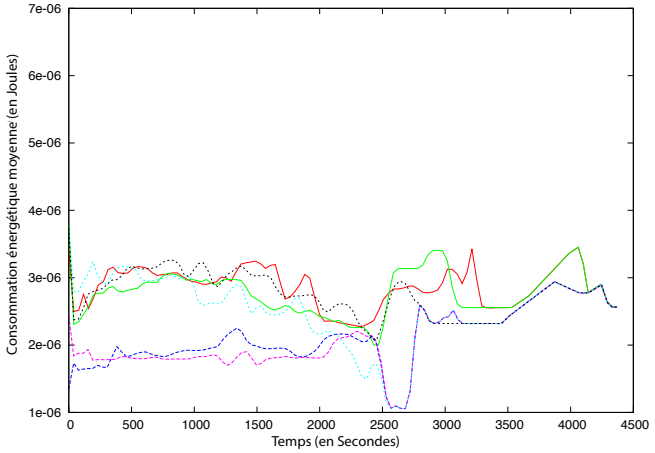


(c) $\delta = 20$

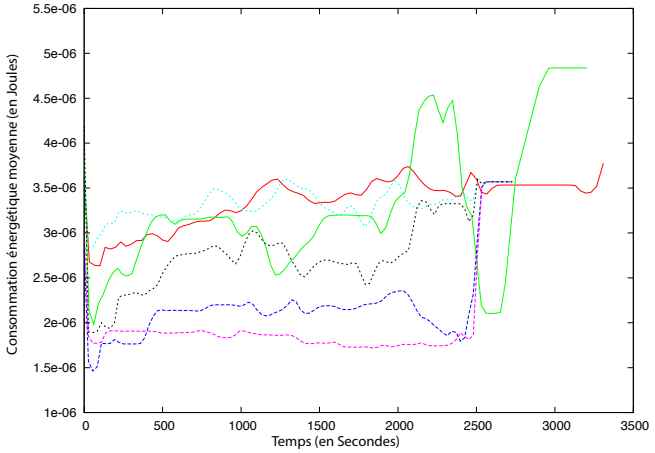
FIGURE 5.9 – Longueur moyenne de la phase de récupération au cours du temps



(a) $\delta = 10$



(b) $\delta = 15$



(c) $\delta = 20$

FIGURE 5.10 – Coût énergétique moyen d'un paquet en fonction du temps.

autant que possible un routage globalement glouton entre la source et la destination. Plusieurs solutions de maintien de la connexité du réseau ont été employé et leur impacts respectifs ont été analysés.
Afin d'évaluer les performances de notre approche, nous l'avons comparé au protocole de routage dans les réseaux d'actionneurs : CoMNet. Selon les résultats, notre proposition présente de meilleurs résultats en termes de performances énergétiques et, surtout, en termes de succès de routage.

Pour la continuité de ce travail, nous souhaiterions combiner le mécanisme de récupération légère à un autre mécanisme de récupération, plus lourd. En effet la récupération légère ne suffit pas pour garantir la délivrance des messages dans la totalité des cas. Très légère dans son implémentation (quelques bits dans les paquets) et dans ses calculs, elle repose toutefois sur l'hypothèse du disque unitaire pour ses contournements, hypothèse difficile à réaliser dans des conditions réelles. D'autres pistes restent à explorer, comme par exemple une approche heuristique de contournement des obstacles.

Chapitre 6

Conclusion et perspectives

6.1 Conclusion

L'objectif de nos travaux a été de proposer de nouvelles approches de routage géographique dans les réseaux sans fil composés d'actionneurs. Pour cela, il m'a donc fallu concevoir des solutions de routage pouvant s'appliquer dans des réseaux composés de capteurs mobiles (ou actionneurs). Dans leur élaboration, nous nous sommes particulièrement attaché à n'utiliser que des informations locales afin de permettre un routage le plus léger possible en termes de consommation énergétique, tout en étant applicable à des réseaux de grande taille. On économise ainsi des messages, des ressources et on augmente la durée de vie du réseau. Toutes les solutions proposées ont été évaluées, via des simulations, et comparées aux meilleures solutions de la littérature pour en déterminer à la fois la pertinence et les performances.

Dans un premier temps, notre travail a été motivé par un constat : l'utilisation des protocoles de routage pour actionneurs altère la topologie du réseau pour permettre des économies d'énergie. Mais en altérant la topologie pour optimiser la communication avec le prochain saut, on modifie aussi le voisinage de ce prochain saut. Notre algorithme va donc essayer d'anticiper au mieux les conséquences du routage en explorant par le calcul l'espace de tous les chemins de routage possibles sur son voisinage vers la destination. Ainsi le choix du prochain nœud est toujours fait de façon localisée, mais avec une heuristique qui a une vision plus « globale » que dans les propositions de la littérature.

Dans un second temps, nous avons souhaité répondre à un nouveau problème de conflits, spécifique aux réseaux d'actionneurs et à leurs applications. En effet, les réseaux de capteurs/actionneurs sont principalement utilisés pour la détection d'événements. Or ces événements sont souvent détectés par plusieurs nœuds du réseau, qui vont chacun construire un chemin de routage jusqu'à la destination en adaptant la topologie. La destination étant la même, ces multiples chemins vont partager des nœuds. Ces nœuds partagés vont osciller entre les différents chemins en réponse aux demandes successives

des schémas de déplacement de chacun des chemins passant par ce nœud d'intersection. Nous avons donc proposé une solution localisée permettant de stopper ces oscillations. Elle provoque aussi une fusion physique des chemins de routage en amont des intersections, à laquelle on a ajouté une agrégation des paquets sur les chemins fusionnés. De l'énergie est économisée : le réseau fonctionne plus longtemps.

Enfin, nous nous sommes attaché à proposer une nouvelle approche, basée sur la mobilité contrôlée, pour permettre le contournement d'obstacles. Notre proposition s'efforce d'adapter la topologie autour de l'obstacle pour permettre de créer un contournement sur lequel un algorithme glouton fonctionne.

Nos travaux ont donc confirmé les apports que permet la mobilité contrôlée au routage. Elle permet des topologies adaptées au trafic, mais aussi de rendre possible des routages qui ne le seraient pas sans cette mobilitée contôlée. Nos travaux, centrés sur l'élaboration de nouveau protocole de routage, sont aboutis et ouvrent de plus la voie à de nouvelles recherches et autres applications.

6.2 Perspectives

Au cours de nos travaux, nous avons régulièrement ébauché les perspectives ouvertes par nos réalisations. De façon générale, elles mériteraient toutes un approfondissement et des optimisations. On pourrait aussi étudier leur combinaison, comme par exemple l'introduction du principe de l'anticipation via la récursivité dans le contournement d'obstacle. La proposition de nouveaux schémas de déplacement reste aussi un axe de recherche à considérer, ou l'introduction du critère de temps dans les calculs de coûts.

De façon plus générale, nous sommes persuadés de la nécessité de passer à un modèle d'environnement moins « idéal ». Le but de cette thèse était de proposer des approches de routage, nous n'avons donc considéré que des conditions idéales permettant d'étudier au mieux l'impact de notre utilisation de la mobilité. Il s'agit maintenant de relâcher une à une les hypothèses simplifiant notre modèle, notamment au niveau des transmissions radio avec la couche MAC idéale ou le modèle de propagation respectant le graphe unitaire. Les protocoles de routage dépendent en effet fortement des protocoles de niveaux inférieurs et des caractéristiques du matériel et de l'environnement. Cela implique de nouveaux travaux, notamment sur les protocoles de type *Hello*, car ici on maîtrise, via la mobilité contrôlée, une des causes principales d'évolution du voisinage.

Des améliorations concernant la prise en compte des différents aspects de la mobilité sont aussi à envisager. La simulation du mouvement a fait l'objet de progrès constants en terme de précision tout au long de notre travail, passant de la téléportation à un mouvement plus continu, et donc plus réaliste. Il s'agit de continuer à étudier cet aspect. Ainsi, le principe de nœuds possédant plusieurs batteries, une étant dédiée à la mobilité tandis que l'autre est dédiée au reste des activités (calculs, communications...),

implique de nouvelles perspectives de recherche sur l'utilisation des nœuds.

Il s'agit aussi de continuer à explorer les mécanismes de maintien de la connexité dans les réseaux au vu de leur impact considérable dans les résultats de simulation des différents protocoles. On pourrait envisager de limiter la mobilité à des instants précis, synchronisés localement ou globalement, afin d'avoir une liberté de mouvement plus grande. En effet, avec la téléportation, on vérifiait uniquement que la nouvelle position (finale) d'un nœud ne mettait pas en péril la connexité globale du réseau. Avec le modèle de déplacement continu, on vérifie régulièrement, pendant son déplacement, que le nœud maintient cette connexité. Si la connexité est perdue, il fait « marche arrière » jusqu'à ce qu'elle soit rétablie. Or, il est possible que cette perte ne soit que temporaire, et que la connexité soit rétablie dès lors que le nœud aurait atteint sa destination. On pourrait alors n'autoriser la mobilité du réseau que durant des intervalles de temps déterminés, pendant lesquels la connexité ne serait pas garantie, la connexité restant garantie en dehors de ces périodes.

Dans la même optique de maintien de la connexité, nous pensons que l'introduction de forces virtuelles dans les mouvements, le mouvement d'un nœud provoquant l'attraction de ses voisins, ou, pourquoi pas, leur répulsion, est une autre piste à explorer.

Une autre piste de recherche possible est le fait de provoquer la création de plusieurs chemins de routage topologiquement adaptés à chaque étape. En effet, les ordres de déplacement sont soit joints au paquet à router, soit inclus dans un beacon. Ces informations de déplacement sont reçues par la totalité du voisinage du nœud courant. Mais, dans l'approche actuelle, seul un nœud les exploite. Il s'agirait alors de profiter de ces communications pour provoquer le mouvement de *plusieurs* nœuds au lieu du prochain seulement.

Enfin, dans le cadre de réseaux toujours plus grands, les approches de *clustering* ont prouvé leur efficacité dans les réseaux statiques. Nous pensons que l'introduction de la mobilité contrôlée dans ce paradigme permettrait de nouvelles approches plus économes en énergie, et donc plus efficaces. Le rôle de *cluster-head* pourrait être dévolu aux actionneurs, qui, avec leur mobilité contrôlée, pourraient alors rejoindre des positions idéales au sein de chaque *cluster*. Les actionneurs restants peuvent alors servir à créer des chemins de routage topologiquement adaptés entre les différents *clusters*.

Sur le plus long terme ces réseaux ayant une topologie auto-adaptative vont trouver leur application, notamment dans la création de réseaux de secours en cas de catastrophe naturelle. On dispose, grâce à ces actionneurs, de la possibilité d'avoir des réseaux qui s'auto-déploient pour couvrir une zone précise, dont la topologie s'adapte au trafic du réseau pour augmenter sa durée de vie ou sa capacité, le tout en totale autonomie. Les contraintes énergétiques et d'auto-configuration seront donc fortes, avec des communications qui pourront n'être qu'intermittentes ou, au mieux, peu fiables. La mobilité pourra elle aussi être perturbée à cause d'un environnement instable, et devra,nous le

pensons faire l'objet de mouvements et de relocalisations en trois dimensions, et non plus seulement deux, pour fournir les meilleures performances.

Glossaire

Actionneur : un actionneur est un capteur capable d'agir sur son environnement. Dans le cadre de cette thèse un actionneur est un capteur doté d'un mécanisme de locomotion et dont le déplacement est contrôlable. On parle aussi de capteurs mobiles.

Ad-Hoc : réseau sans fil ne reposant pas sur des infrastructures fixes.

Algorithme glouton : qui suit le principe de faire, étape par étape, un choix optimum local, dans l'espoir d'obtenir un résultat optimum global.

Capteur : Système embarqué caractérisé par une puissance de calcul et une quantité de mémoire très faibles. Souvent alimentés par des batteries, ils sont associés dans des réseaux de grande échelle dans lesquels ils communiquent via le médium radio.

Connexité : issu de la théorie des graphes, le terme connexité désigne la possibilité dans un réseau de capteurs (assimilable à un graphe) pour n'importe quelle paire de capteurs du réseau de communiquer directement ou en utilisant des nœuds intermédiaires.

Nœud : les réseaux de capteurs étant assimilables à des graphes, on appellera indifféremment nœud un capteur. Voir capteur.

Protocole (de communication) : ensemble de conventions de communication partagées par les éléments d'un réseau pour l'échange et/ ou l'acheminement d'informations.

Réseaux véhiculaires : sous famille des réseaux ad-hoc. Ils sont caractérisés par leur topologie très chaotique, parfois dense, parfois très rapidement clairsemé, et une localisation souvent fiable. Ils ne sont généralement pas concernés par les contraintes énergétiques car alimentés par la batterie du véhicule.

Routage : mécanisme d'acheminement des données dans un réseau d'un expéditeur vers un ou plusieurs destinataires.

Schéma de déplacement : principe de relocalisation des voisins du nœud courant dans le cas idéal où le mécanisme de maintien de la connexité ne limite pas le mouve-

ment.

Table de routage : structure de données qui permet de mémoriser les choix précédents du routage. À une destination donnée, elle retourne le prochain nœud à qui retransmettre le paquet.

Topologie : configuration géographique globale du réseau reposant sur les positions individuelles des nœuds le composant.

Bibliographie

[BMJ+98] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*, MobiCom '98, pages 85–97, New York, NY, USA, 1998. ACM.

[BMSU99] Prosenjit Bose, Pat Morin, Ivan Stojmenović, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. In *Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications*, DIALM '99, pages 48–55, New York, NY, USA, 1999. ACM.

[BMSU01] Prosenjit Bose, Pat Morin, Ivan Stojmenović, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6) :609–616, 2001.

[CHH01] Srdjan Capkun, Maher Hamdi, and Jean-Pierre Hubaux. Gps-free positioning in mobile ad-hoc networks. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, HICSS '01, pages 9008–9018, Washington, DC, USA, 2001. IEEE Computer Society.

[CLN09] Carmelo Costanzo, Valeria Loscrì, and Enrico Natalizio. Distributed virtual-movement scheme for improving energy efficiency in wireless sensor networks. In *Proceedings of the 12th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, MSWiM '09, pages 297–304, New York, NY, USA, 2009. ACM.

[CSR04] J. Carle and D. Simplot-Ryl. Energy-efficient area monitoring for sensor networks. *Computer*, 37(2) :40–46, 2004.

[EMSR08] Essia Hamouda Elhafsi, Nathalie Mitton, and David Simplot-Ryl. End-to-end energy efficient geographic path discovery with guaranteed delivery in ad hoc and sensor networks. In *Procedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, PIMRC '08, pages 1–5. Springer, 2008.

[FCF07] Antoine Fraboulet, Guillaume Chelius, and Eric Fleury. Worldsens : development and prototyping tools for application specific wireless sensors networks. In *Proceedings of the 6th international conference on Information processing*

in sensor networks, IPSN '07, pages 176–185, New York, NY, USA, 2007. ACM.

[FGG06] Qing Fang, Jie Gao, and Leonidas J. Guibas. Locating and bypassing holes in sensor networks. *Mobile Network Applications*, 11(2) :187–200, 2006.

[Fin87] G. G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. Technical Report ISI/RR-87-180, Information Sciences Institute, 1987.

[GLM+04] David K. Goldenberg, Jie Lin, A. Stephen Morse, Brad E. Rosen, and Y. Richard Yang. Towards mobility as a network control primitive. In *Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '04, pages 163–174, New York, NY, USA, 2004. ACM.

[HL86] Ting-Chao Hou and V.O.K. Li. Transmission range control in multihop packet radio networks. *IEEE Transactions on Communications*, 34(1) :38–44, 1986.

[HMSR11] Essia Hamouda, Nathalie Mitton, and David Simplot-Ryl. Energy efficient mobile routing in actuator and sensor networks with connectivity preservation. In *Proceedigs of the 10th International Conference on Ad Hoc Networks and Wireless*, AdHocNow '11, pages 15–28. Springer, 2011.

[KH05] Elliott D Kaplan and Christopher J Hegarty. *Understanding GPS : principles and applications*. Artech house, 2005.

[KNS06] Johnson Kuruvila, Amiya Nayak, and Ivan Stojmenovic. Progress and location based localized power aware routing for ad hoc and sensor wireless networks. *International Journal of Distributed Sensor Networks*, 2(2) :147–159, 2006.

[KSU99] Evangelos Kranakis, Harvinder Singh, and Jorge Urrutia. Compass routing on geometric networks. In *Proceedings of the 11th canadian conference on computational geometry*, CCCG '99, pages 51–54, 1999.

[KV00] Young-Bae Ko and Nitin H Vaidya. Location-aided routing (lar) in mobile ad hoc networks. *Wireless Networks*, 6(4) :307–321, 2000.

[LLM06] Ben Leong, Barbara Liskov, and Robert Morris. Geographic routing without planarization. In *Proceedings of the 3rd conference on Networked Systems Design & Implementation*, volume 3 of *NSDI'06*, pages 25–25, Berkeley, CA, USA, 2006. USENIX Association.

[LNC10] Valeria Loscrí, Enrico Natalizio, and Carmelo Costanzo. Simulations of the impact of controlled mobility for routing protocols. *EURASIP Journal on Wireless Communications and Networking*, 2010 :7 :1–7 :12, 2010.

[LNS07] Hai Liu, Amiya Nayak, and Ivan Stojmenović. Localized mobility control routing in robotic sensor wireless networks. In Hongke Zhang, Stephan Olariu, Jiannong Cao, and David B. Johnson, editors, *Mobile Ad-Hoc and Sensor Networks*, volume 4864 of *Lecture Notes in Computer Science*, pages 19–31. Springer Berlin Heidelberg, 2007.

[LR00] Qun Li and Daniela Rus. Sending messages to mobile users in disconnected ad-hoc wireless networks. In *Proceeding of the 6th annual international conference on Mobile computing and networking*, MobiCom '00, pages 44–55, New York, NY, USA, 2000. ACM.

[MWH01] Martin Mauve, A Widmer, and Hannes Hartenstein. A survey on position-based routing in mobile ad hoc networks. *Network*, 15(6) :30–39, 2001.

[NK84] Randolph Nelson and Leonard Kleinrock. The spatial capacity of a slotted aloha multihop packet radio network with capture. *IEEE Transactions on Communications*, 32(6) :684–694, 1984.

[NN01] Dragos Niculescu and Badri Nath. Ad hoc positioning system (aps). In *Proceedings of the IEEE Global Telecommunications Conference*, GLOBECOM '01, pages 2926–2931. IEEE, 2001.

[RM99] Volkan Rodoplu and T. Meng. Minimizing energy mobile wireless networks. *IEEE Journal in Selected Areas in Communications*, 17(8) :1333–1344, 1999.

[RSR11] Tahiry Razafindralambo and David Simplot-Ryl. Connectivity preservation and coverage schemes for wireless sensor networks. *Transaction on Automatic Control*, 56(10) :2418–2428, 2011.

[SL01a] Ivan Stojmenovic and Xu Lin. Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 12(10) :1023–1032, 2001.

[SL01b] Ivan Stojmenovic and Xu Lin. Power-aware localized routing in wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 12(11) :1122–1133, 2001.

[TK84] Hideaki Takagi and Leonard Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Transactions on Communications*, 32(3) :246–257, 1984.

[Tou80] G. T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12 :261–268, 1980.

[WSC08] Wei Wang, Vikram Srinivasan, and Kee-Chaing Chua. Extending the lifetime of wireless sensor networks through mobile relays. *IEEE/ACM Transactions on Networking*, 16(5) :1108–1120, 2008.

[ZA03] Wenrui Zhao and Mostafa H. Ammar. Message ferrying : proactive routing in highly-partitioned wireless ad hoc networks. In *Proceedings of the Ninth IEEE Workshop on Future Trends of Distributed Computing Systems*, FTCDS'03, pages 308–314, Washington, DC, USA, 2003. IEEE Computer Society.

[ZAZ04] Wenrui Zhao, Mostafa H. Ammar, and Ellen Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '04, pages 187–198, New York, NY, USA, 2004. ACM.

[ZAZ05] Wenrui Zhao, Mostafa Ammar, and Ellen Zegura. Controlling the mobility of multiple data transport ferries in a delay-tolerant network. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, INFOCOM '05, pages 1407–1418. IEEE, 2005.