Exploitation de l'hétérogénéité des réseaux de capteurs et d'actionneurs dans la conception des protocoles d'auto-organisation et de routage

Bilel Romdhani

N d'ordre 2012-ISAL-XXX Année 2012

Thèse

présentée devant
L'Institut National des Sciences Appliquées de Lyon
pour l'obtention
du Grade de Docteur

Ecole doctorale : Informatique et Mathématiques de Lyon
présentée et soutenue publiquement
le 18 07 2012

par

Mr ROMDHANI Bilel

Exploitation de l'hétérogénéité des réseaux de capteurs et d'actionneurs dans la conception des protocoles d'auto-organisation et de routage

Sous la direction de :

Monsieur Fabrice VALOIS	Professeur INSA de Lyon
Monsieur Dominique BARTHEL	France Telecom R&D

Soutenue devant :

Madame Isabelle Guérin-Lassous (Présidente)	Professeure Université Lyon I
Monsieur André-Luc BEYLOT (Rapporteur)	Professeur Université de Toulouse
Monsieur Bernard TOURANCHEAU (Rapporteur)	Professeur Université de Grenoble
Monsieur Fabrice THEOLEYRE (Examinateur)	Chargé de Recherche CNRS
Monsieur Laurent TOUTAIN (Examinateur)	Maître de conférence Télécom Bretagne

Les travaux présentés dans ce mémoire ont été financés partiellement par le projet ANR verso ARESA2 2009-17.
Ces travaux ont été réalisés en collaboration entre le laboratoire CITI sous la direction du Mr. Fabrice VALOIS et Orange Labs sous la direction de Mr. Dominique BARTHEL

Résumé

Ces dernières années, nous avons assisté à l'apparition des réseaux sans fil multi-sauts. Avec des capacités distinctes, des caractéristiques différentes et des applications cibles variées, nous pouvons identifier les réseaux de capteurs (WSNs) et plus récemment les réseaux de capteurs et actionneurs (WSANs). Ces derniers sont des réseaux auto-organisés et ils sont constitués d'un grand nombre de nœuds capteurs autonomes à faible ressources (capacité de calcul, de mémoire, de puissance d'émission, etc...) et un nombre moins important de nœuds actionneurs disposant généralement d'une source d'énergie abondante. Les fonctions de calcul et de communication au niveau des actionneurs peuvent donc profiter de cette richesse en énergie : par conséquent les capacités de calcul et de stockage ainsi que la puissance de transmission au niveau des actionneurs sont plus importantes. Dans cette thèse, nous nous sommes intéressés aux réseaux urbains considérés par le projet ANR ARESA2 qui sont principalement des réseaux de capteurs et actionneurs hétérogènes : l'hétérogénéité est causée par la coexistence des nœuds capteurs à faibles ressources et des nœuds actionneurs riches en ressources. Ces derniers devraient être utilisés de manière différenciée par le réseau. C'est dans ce contexte que se déroule cette thèse dans laquelle nous avons étudié des algorithmes d'auto-organisations et de routage s'appuyant sur l'hétérogénéité.

Au début, nous nous sommes intéressés à l'auto-organisation dans un contexte hétérogène. Se basant sur l'idée que les ressources au niveau des nœuds actionneurs doivent être exploitées afin de réduire la charge de communication au niveau des nœuds capteurs, nous avons proposé un protocole d'auto-organisation appelée Far-Legos. Far-Legos permet de profiter de la puissance d'émission des actionneurs pour apporter une information de gradient au niveau des capteurs. Les actionneurs initient et construisent une topologie logique. Cette dernière sera utilisée pour faciliter la phase de collecte de données à partir des nœuds capteurs vers les nœuds actionneurs.

Ensuite, nous nous sommes intéressés aux liens asymétriques causés par la présence de différents types de nœuds avec différentes portées de transmission. Ces liens asymétriques, causés par l'hétérogénéité au niveau des nœuds constituant le réseau, peuvent détériorer les performances des protocoles de routage qui ne tiennent pas compte de ce type de liens. Pour éviter la dégradation de ces protocoles de routage, nous introduisons une nouvelle métrique de calcul de gradient ou de rang. Celle-ci sera utile pour détecter et éviter les liens asymétriques au niveau de la couche réseau pour le protocole de routage RPL. Nous présentons aussi une adaptation du protocole de collecte de données basé sur Legos pour détecter et éviter ces liens asymétriques.

Enfin, nous nous sommes intéressés à l'exploitation de ces liens asymétriques. Nous proposons ainsi un protocole de collecte de données dédiés aux réseaux hétérogènes contenant des liens asymétriques appelé AsymRP. AsymRP est un protocole de routage dédié au trafic de collecte de données basé sur une connaissance de voisinage à 2-sauts combinée avec l'utilisation des messages d'acquittements (ACKs) implicites et une technique de routage de messages ACKs explicites. Cette proposition tire profit des liens asymétriques afin d'assurer une collecte de données fiable.

Abstract

In recent years, we witnessed the appearance of multi-hop wireless networks. With distinct capabilities, properties, characteristics and target applications, we can identify Wireless Sensor Networks (WSNs) and Wireless Sensor and Actuator Networks (WSANs). The latter are self-organized networks composed of a large number of autonomous low power nodes called sensors and a number of actuators (relatively small compared to the number of sensor nodes). Actuator nodes are resource-rich devices with higher processing capabilities, transmission power and larger battery capacity. In this thesis, we focused on urban wireless networks considered by the ANR project ARESA2. The networks considered by this project are heterogeneous networks. This heterogeneity is caused by the coexistence of sensor nodes with limited resources and actuator nodes with higher resources. Actuators nodes should be used differentially by the network. Hence designed protocols for WSANs should exploit resource-rich devices to reduce the communication burden on low power nodes. It is in this context that this thesis takes place in which we studied self-organizing and routing algorithms based on the heterogeneity.

First, we are interested in self-organization protocols in a heterogeneous network. Based on the idea that resource-rich nodes must be exploited to reduce the communication load level on low-power nodes, we proposed self-organizing protocol called Far-Legos. Far-Legos uses the large transmit power of actuators to provide gradient information to sensor nodes. Actuators initiate and construct a logical topology. The nature of this logical topology is different inside and outside the transmission range of these resourceful nodes. This logical topology will be used to facilitate the data collection from sensor to actuator nodes.

Second, we investigated the asymmetric links caused by the presence of heterogeneous nodes with different transmission ranges. The apparition of asymmetric links can dramatically decrease the performance of routing protocols that are not designed to support them. To prevent performance degradation of these routing protocols, we introduce a new metric for rank calculation. This metric will be useful to detect and avoid asymmetric links for RPL routing protocol. We also present an adaptation of data collection protocol based on Legos to detect and avoid these asymmetric links.

Finally, we are interested in exploiting the asymmetric links present in the network. We proposed a new routing protocol for data collection in heterogeneous networks, called AsymRP. AsymRP, a convergecast routing protocol, assumes 2-hop neighborhood knowledge and uses implicit and explicit acknowledgment. It takes advantage of asymmetric links to ensure reliable data collection.

Remerciements

Le travail présenté dans ce mémoire de thèse a été réalisé au sein du projet ANR ARESA2 hébergé par Orange Labs à Meylan.

J'adresse mes premiers remerciements à France Télécom R&D et au laboratoire CITI pour avoir participé au financement de ces travaux et pour m'avoir accueillie dans leurs murs. Ces remerciements s'adresse particulièrement à Monsieur ***Dominique BARTHEL*** *et Monsieur Fabrice* ***VALOIS****.*

Je suis très reconnaissant envers Madame ***Isabelle Guérin-Lassous****, professeure à l'université de Lyon I, envers Monsieur* ***Fabrice THEOLEYRE****, chargé de recherche CNRS au sein du laboratoire LSIIT de l'université de Strasbourg, et envers Monsieur* ***Laurent TOUTAIN****, maître de conférence au sein du département réseaux, sécurité et Multimédia à l'ENST Bretagne, pour avoir accepté de juger ce travail.*

J'exprime ma gratitude aussi envers Monsieur le Professeur ***André-Luc BEYLOT****, professeur des universités au sein du laboratoire IRIT à l'université de Toulouse, et à Monsieur le Professeur* ***Bernard TOURANCHEAU****, professeur des universités au sein du laboratoire LIG à l'université de Grenoble, pour avoir accepté d'évaluer mon travail.*

Mes remerciements sont aussi adressés à toute l'équipe de l'unité de recherche TECH/MATIS/CITY chez Orange Labs pour leur aide si précieuse et leur soutien constant.

Je ne saurais oublier de remercier tous mes collègues ainsi qu'à tous les membres du laboratoire dans lequel j'ai eu l'opportunité et le plaisir d'y travailler.

Un grand merci à mes parents et à mes frères et sœurs pour leur amour inconditionnel et leur soutien si précieux durant ces dernières années.

Table des matières

Table des figures

Liste des tableaux

1 Introduction

Sommaire

1.1 Définitions

1.1.1 Les réseaux de capteurs

Les avancées récentes dans la micro-électronique et la communication sans fil ont permis le développement des nœuds capteurs de petite taille, à faible coût, contraints en énergie et communicants à courtes distances. Un réseau de capteurs [8], WSNs (*Wireless Sensor Networks*), consiste en un grand nombre de nœuds capteurs distribués qui s'auto-organisent en un réseau sans fil multi-sauts (Figure 1.2). L'objectif de ces nœuds capteurs est de récolter des grandeurs physiques à partir de l'environnement où ils sont déployés (température, pression, humidité, luminosité, etc.), de les traiter et enfin de les envoyer vers un (ou plusieurs) nœud(s) spécifique(s) dans le réseau appelé une entité de collecte ou Puits. Chaque nœud capteur est composé (voir figure 1.1) :

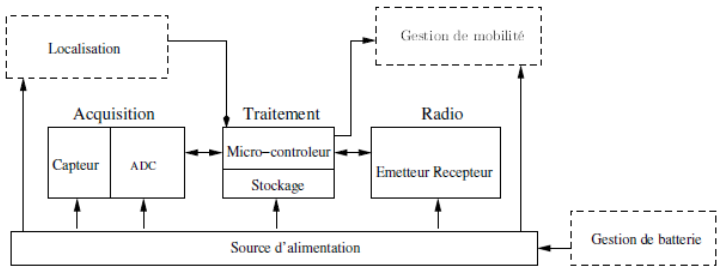


FIGURE 1.1: Architecture d'un nœud capteur [8]

- D'une unité de mesure chargée de transformer une grandeur physique récoltée dans l'environnement où il est déployé en une grandeur numérique.
- D'un micro-processeur avec une faible puissance de calcul et un faible espace de stockage par rapport aux ordinateurs ou aux Smartphones.
- D'un module de transmission sans fil avec une puissance radio limitée ne dépassant pas une centaine de mètres en extérieur et quelques dizaines de mètres en intérieur.
- D'une batterie limitée partagée par toutes les unités décrites ci-dessus.

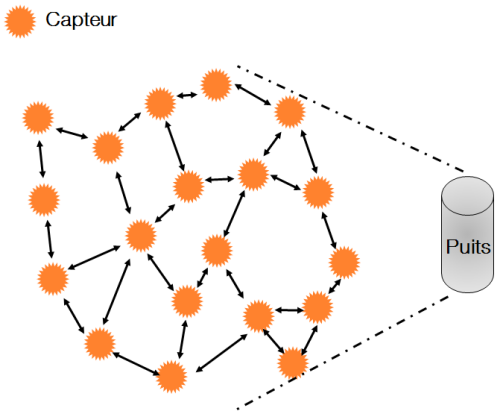


FIGURE 1.2: Exemple de réseau de capteurs (WSN)

Les principaux défis au niveau des WSNs sont [8] :

- Besoin de mécanismes pour préserver l'énergie au niveau des nœuds capteurs et ainsi maximiser

la durée de vie du réseau. Dans la littérature, plusieurs définitions de durée de vie existent et qui peuvent être classifiées en durée de vie basée sur le nombre des nœuds vivants, durée de vie basée sur la couverture et la connectivité, et durée de vie basée sur la qualité exigée par l'application [43].
- Besoin d'algorithmes distribués, localisés et collaboratifs pouvant être exécutés avec des micro-processeurs à faible puissance de calcul et une mémoire limitée.
- Besoin d'un routage multi-sauts sur une topologie dynamique pour acheminer les données à partir des nœuds capteurs vers le(s) nœud(s) puits.

1.1.2 Les réseaux de capteurs et actionneurs

Plus récemment, nous assistons à l'apparition des réseaux de capteurs et actionneurs WSANs (*Wireless Sensors and Actuators Networks*).

Ces WSANs sont composés d'un grand nombre de nœuds capteurs et un nombre moins conséquent de nœuds actionneurs autonomes comme indiqué dans la figure 1.3.

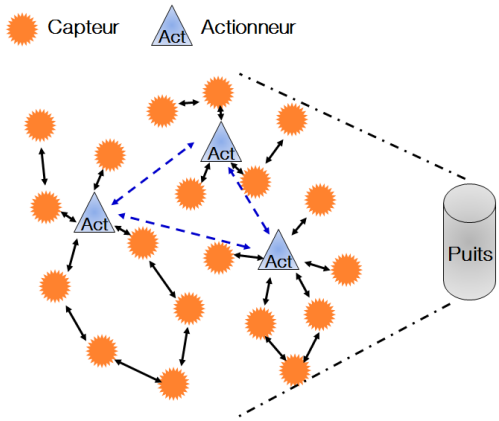


FIGURE 1.3: Exemple de réseau de capteurs et actionneurs (WSAN)

Les WSANs sont des réseaux émergents ayant une large gamme d'applications potentielles, y compris des applications civiles (comme la gestion de place de parking pour réduire les accidents et le taux de pollution [1]) ou des applications environnementales (comme la détection d'incendie de forêt [86]) ou des applications médicales (comme la surveillance de santé des malades [102]), etc.

Chaque nœud capteur a un micro-processeur avec une faible puissance de calcul, un espace de stockage limité, une faible puissance radio et une réserve d'énergie réduite. Un actionneur est un nœud dont l'objectif est de convertir un signal électrique de commande à une action physique en se basant sur les informations récoltées par les nœuds capteurs.

Comme l'action sur le monde physique exige une énergie souvent importante, les actionneurs disposent généralement d'une source d'énergie abondante. Les fonctions de calcul et de communication au niveau des actionneurs profitent donc de cette richesse en énergie : par conséquent les capacités de calcul et de stockage ainsi que la puissance de transmission au niveau des nœuds actionneurs sont plus importantes que celles au niveau des nœuds capteurs [6].

Dans cette thèse, nous ne nous intéressons pas à l'action physique des nœuds actionneurs ni à son impact mais plutôt à l'apport des nœuds actionneurs au niveau du réseau puisque ils exécutent aussi

1. SmartPark Project : http ://smartpark.epfl.ch/

les fonctions réseau à savoir recevoir, transmettre, traiter et relayer des données. Nous supposons aussi que les communications et les coordinations entre les nœuds actionneurs se font à travers un réseau dédié. Ainsi nous ne traitons pas les problématiques liées à la coordinations et à la communication entre ces nœuds.

Dans ce manuscrit, l'actionneur fait référence à un nœud communicant dans le réseau ayant des capacités de calcul, de stockage et une puissance de transmission supérieures à celles des nœuds capteurs.

Ainsi, la présence des nœuds actionneurs introduit une caractéristique importante des WSANs qui est l'hétérogénéité : entrainant par exemple l'apparition des liens asymétriques causés par la présence de différentes puissances de transmission dans le réseau. Cette hétérogénéité ouvre des nouvelles opportunités pour proposer des nouveaux algorithmes qui tiennent compte et aussi qui tirent profit de cette hétérogénéité.

1.2 Applications des WSANs

Ces dernières années, nous assistons à l'apparition de plus en plus d'applications déployées pour les WSNs et les WSANs. Ces applications couvrent plusieurs domaines tels que les réseaux urbains [45], domotiques [24], industriels [119] etc.

Dans le projet ANR ARESA2 [2], nous nous sommes intéressés aux réseaux urbains. Nous avons distingué principalement deux types d'applications :

- Les applications basées sur le mode événementiel où chaque nœud, se basant sur les informations récoltées, décide ou pas d'envoyer ces données vers le(s) nœud(s) puits ou actionneur(s).
- Les applications basées sur le mode périodique où les capteurs envoient périodiquement des informations sur l'environnement vers le(s) nœud(s) puits ou actionneur(s).

1.2.1 Quelques scénarii d'applications

Plusieurs scénarii ont été définit par le projet ARESA2, nous trouvons principalement :

- La télé-relève de compteurs d'eau, de gaz ou d'électricité où deux types de trafics sont utilisés :
  - La collecte de données où les nœuds capteurs envoient périodiquement des mesures vers le(s) nœud(s) puits ou actionneur(s).
  - La dissémination des messages de contrôles où un message de contrôle doit être disséminé du (des) nœud(s) puits ou actionneur(s) vers les nœuds capteurs.
- La surveillance de la pollution urbaine : ce scénario peut servir pour récolter des données périodiques sur l'environnement ou pour détecter des dépassements de seuil de pollution afin d'alerter les autorités. Le trafic dominant consiste en une collecte de données des nœuds capteurs vers le(s) nœud(s) puits ou actionneur(s).
- La gestion des déchets : le trafic dominant dans ces types d'applications est la collecte événementielle. En effet, quand les conteneurs sont presque pleins, les capteurs, détectant cet évènement, envoient une alerte au(x) nœud(s) puits ou actionneur(s).

Nous remarquons ainsi que chaque application exige des contraintes différentes, ce qui va influencer les protocoles qui vont être utilisés avec ces applications.

2. Projet ARESA2 : http ://aresa2.orange-labs.fr/

1.2.2 Caractéristiques des applications des WSANs

Malgré la diversité des contraintes exigées par chacune des applications, plusieurs paramètres sont partagés entre toutes ces applications. Nous pouvons citer :

- La taille du réseau : un WSAN est composé de quelques dizaines à quelques centaines de nœuds. La topologie obtenue est principalement une architecture multi-sauts.
- Déploiement de nœuds : une fois déployés, les nœuds sont généralement fixes. Il est à noter que même si le déploiement est statique, la topologie peut changer suite aux perturbations au niveau des liens qui sont généralement volatiles.
- Support de multiples types de trafics : le trafic dominant dans les WSNs et les WSANs est la collecte des données (à partir des nœuds capteurs vers le(s) nœud(s) puits ou actionneur(s)). Mais le réseau WSAN doit aussi supporter le trafic de dissémination (d'un nœud puits ou actionneur vers un ensemble de nœuds capteurs) et aussi le trafic entre différents nœuds dans le réseau (généralement d'un nœud capteur vers un nœud actionneur ou d'un nœud actionneur vers un autre nœud actionneur).
- Gestion d'énergie : les nœuds capteurs sont généralement alimentés par des batteries. Ainsi, des techniques économes en énergie sont nécessaires pour maximiser la durée de vie du réseau.
- Hétérogénéité : les réseaux WSANs sont constitués de plusieurs types de nœuds, à savoir des nœuds capteurs et des nœuds actionneurs. D'une part, les nœuds capteurs ont des capacités limitées (calcul, mémoire, batterie et puissance de transmission). Ces nœuds ne devraient pas normalement participer activement à la maintenance de la topologie par exemple sauf pour assurer une connectivité locale ou une couverture appropriée. D'autre part, les nœuds actionneurs sont des nœuds riches en ressources. Ainsi, le réseau doit être conçu et géré de telle sorte que les nœuds à faibles ressources (les nœuds capteurs) ne soient pas trop sollicités pour ne pas épuiser leurs batteries limitées au contraire des nœuds riches en ressources (les nœuds actionneurs) qui peuvent contribuer davantage dans le réseau.

1.3 Défis et motivations

Les réseaux urbains considérés dans par le projet ANR ARESA2 sont des réseaux hétérogènes et dynamiques. L'hétérogénéité est causée par la coexistence des nœuds capteurs à faibles ressources et des nœuds actionneurs riches en ressources. Ces derniers devraient être utilisés de manière différenciée par le réseau. Ainsi, nous devrons donc explorer des algorithmes s'appuyant sur l'hétérogénéité. La dynamique est causée par l'aspect volatile des liens radios. En effet, l'apparition et la disparition des liens radios dans le réseau implique une variabilité au niveau du voisinage

En plus des défis que nous trouvons dans les WSNs, les WSANs introduisent des nouveaux défis.

Parmi ces défis nous trouvons :

- Défis de coordination entre les actionneurs : Outre la communication classique capteur-actionneur, les actionneurs doivent se coordonner. Ces actionneurs sont généralement des nœuds riches en ressources avec une puissance de transmission élevée, la communication actionneur-actionneur peut être avantageusement à long distance contrairement à la communication capteur-actionneur et éventuellement via un réseau tiers [104].

- Défis de collaboration entre les actionneurs : Les actionneurs doivent collaborer afin de décider sur les problèmes d'affectation de tâche suite à un évènement reporté par les nœuds capteurs par exemple. A partir des informations reçues, l'actionneur doit décider si la tâche nécessite une seule commande (comment sélectionner cet actionneur unique) ou plus d'un actionneur (com-

ment décider sur le nombre optimal d'actionneurs pour exécuter l'action). De plus, lorsque les actionneurs ne peuvent pas communiquer directement entre eux, ils utilisent des nœuds capteurs comme des nœuds intermédiaires, ce qui signifie que la coordination actionneur-actionneur peut être réalisée par des nœuds capteurs [6].

- Défis d'hétérogénéité : La coexistence de deux types de nœuds limités en ressources et de nœuds riches en ressources, introduit des nouveaux défis qui ne se posent pas dans les WSNs. En effet, les travaux autours des réseaux ad-hoc ou les WSNs supposent que tous les nœuds sont homogènes (au niveau des ressources disponibles). Cette hétérogénéité, inévitable dans les WSANs, est une question difficile exigeant la conception de nouveaux mécanismes qui prennent en compte cette caractéristique [157].

- Défis énergétique : L'existence des nœuds à faibles ressources dans un WSAN impose la conception de protocoles économes en énergie. Comme dans les WSNs, il faut préserver l'énergie au niveau des nœuds capteurs. Les protocoles de communication utilisés doivent donc limiter l'utilisation de trafic de contrôle et doivent être localisés et distribués. Aussi les nœuds riches en ressources (actionneurs) peuvent être sollicités plus que les nœuds à ressources limitées (capteurs) [93].

- Défis de routage et d'auto-organisation : Les applications visées par le projet ANR ARESA2 et plus généralement par les WSANs sont diverses. Chaque application est susceptible d'exiger une solution de routage et de structuration différente. Ainsi ces protocoles doivent être adaptés aux besoins de l'application tout en respectant les contraintes des ressources dont disposent les différents nœuds du réseau. Les principaux défis que rencontrent ces protocoles de routage et d'auto-organisation dans les WSANs sont [154] :

  - Réseaux décentralisés : La nature décentralisée des WSANs comme les WSNs complique toute tentative de maintenir une table de routage centralisée. Ainsi les mécanismes localisés ou distribués doivent être invoqués.
  - Dynamiques des liens radios : Les liens radios entre les nœuds sont caractérisés par leurs apparitions et disparition au cours du temps. Ceci peut causer principalement des erreurs de paquets (à cause des effets du canal sans fil, des interférences provenant d'autres systèmes). Des mécanismes appropriés au niveau du réseau doivent être donc mis en place pour atténuer ce manque de fiabilité.
  - Caractéristiques du déploiement : Le nombre de nœuds et de leur densité jouent aussi un rôle central dans la conception des protocoles dédiés aux WSANs ou WSNs. Un grand nombre de nœuds avec une densité faible implique généralement un grand nombre de sauts avant d'atteindre la destination finale et une faible redondance de route. En revanche, une grande densité peut offrir une grande redondance de route et peut ainsi faire circuler plusieurs trafics sur le réseau en même temps mais en contre partie une contention plus forte.
  - Capacités limités des nœuds : En général, les capacités de traitement et de stockage sont faibles, cependant, la contrainte de conception la plus importante à prendre en compte est la capacité énergétique limitée des nœuds capteurs. Comme le montre la figure 1.4, la consommation d'énergie est dominée par la consommation de la radio du nœud. Il n'est donc pas seulement souhaitable d'envoyer peu de message mais aussi de minimiser le temps où la radio est allumée. Ainsi les nœuds devraient donc être mis en sommeil le plus longtemps possible sans compromettre les fonctionnalités du réseau.

Dans cette thèse, nous ne traitons pas les défis de collaboration et de coordination entre les ac-

tionneurs. Nous nous concentrons sur les défis d'énergie, d'hétérogénéité, de la dynamique, d'auto-organisation et de routage dans les WSANs. Notre objectif est d'étudier l'apport de cette hétérogénéité pour les protocoles de communication et principalement pour les protocoles d'auto-organisation et les protocoles de routage.

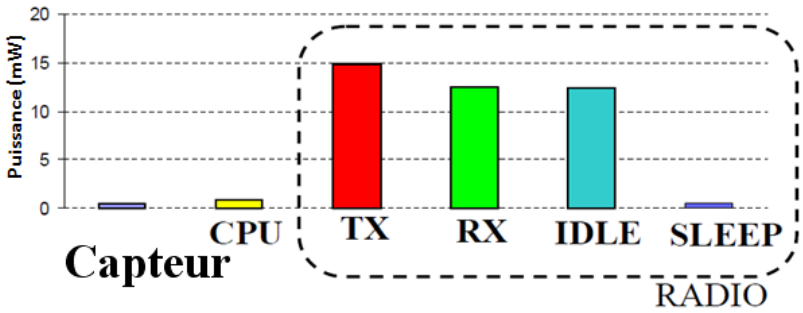


FIGURE 1.4: Principale source de consommation d'énergie d'un capteur avec une radio CC2500 et un micro contrôleur MSP430 [120]

1.4 Organisation de la thèse

Cette thèse est organisée en 7 chapitres. Dans le **chapitre II**, nous commencerons par la présentation des caractéristiques et les défis dans les réseaux homogènes et les réseaux hétérogènes tout en mettant l'accent sur les défis au niveau de la couche réseau dans les WSANs. A la suite, dans le **chapitre III**, nous introduirons les rôles et les objectifs des protocoles de routage et d'auto-organisation dans les WSANs. Nous achèverons ce chapitre par souligner la nécessité d'algorithme de communication (algorithmes d'auto-organisation et de routage) prenant en compte l'hétérogénéité inévitable dans les WSANs.

Dans le **chapitre IV**, nous présenterons une stratégie d'auto-organisation dans les WSANs en proposant la construction d'une topologie logique permettant de faciliter la collecte de données tout en tirant parti des ressources disponibles au niveau des nœuds actionneurs dans le réseau. Ce nouveau protocole d'auto-organisation et de collecte de données construit cette topologie logique à partir des nœuds actionneurs. La nature de cette topologie est différente à l'intérieur ou à l'extérieur de la portée de communication des nœuds actionneurs.

Nous traiterons par la suite la problématique de la présence de liens asymétriques causés par une hétérogénéité au niveau de la portée de transmission des nœuds du réseau. Nous supposons que les nœuds ont des portées de transmission hétérogènes ce qui cause l'apparition des liens asymétriques. Ces liens peuvent dégrader les performances de la plupart des protocoles qui n'ont pas été conçus pour supporter cette hétérogénéité.

Ainsi, nous commencerons dans le **chapitre V** par analyser le protocole de routage RPL [155], un standard au sein du groupe du travail ROLL [3] (*Routing Over Low power and Lossy networks*) de l'IETF [4] (*Internet Engineering Task Force*) dans un contexte hétérogène et dynamique. Nous introduisons dans ce chapitre une nouvelle métrique afin d'éviter les liens asymétriques dans le réseau. Nous décrivons aussi, une adaptation d'un autre protocole de collecte de données Legos [97] pour l'évaluer avec l'adaptation du protocole RPL.

3. ROLL IETF Working Group : http ://datatracker.ietf.org/wg/roll/charter/
4. Internet Engineering Task Force : http ://www.ietf.org

1.4 Organisation de la thèse

Dans le **chapitre VI**, nous proposerons un protocole de collecte de données dans un réseau hétérogène permettant non pas seulement de détecter les liens asymétriques mais aussi permettant de les exploiter durant la phase de collecte de données.

Nous conclurons, dans le **chapitre VII**, en résumant les principaux apports et en présentant les perspectives possibles de notre travail.

Cette thèse a été réalisée en collaboration entre Orange Labs à Meylan, sous la direction de Monsieur Dominique Barthel, et le laboratoire de recherche CITI de l'INSA Lyon sous la direction du Professeur Fabrice Valois. Elle a été financée partiellement par le projet de recherche ANR ARESA2 VERSO 2009-017 [5].

5. ARESA2 : http ://aresa2.orange-labs.fr

Des réseaux homogènes aux réseaux hétérogènes

2

Sommaire

2.1 Introduction

Les avancées technologiques de ces dernières années sont accompagnées de conception d'objets de plus en plus petits, à faible coût, à faible ressources (énergie, communication, calcul), intelligents et autonomes : les capteurs sans fil. Ces capteurs peuvent mesurer une distance, une direction, une vitesse, une vibration, etc. Ces capteurs disposent d'une antenne pour assurer la réception et la transmission des signaux. Ils disposent aussi d'un mini processeur et une capacité de stockage limitée (par rapport aux PCs ou aux *smartphones*) pour (dé)coder les signaux et pour exécuter des protocoles de communications. Enfin, ces capteurs sont alimentés généralement par une batterie.

Les capteurs sont généralement déployés dans un environnement et sont attachés à une station de base appelée puits. Ces capteurs doivent coopérer entre eux pour s'auto-organiser et s'auto configurer afin de constituer un réseau de capteurs (WSN *Wireless Sensor Network*). Ce WSN consiste généralement en un grand nombre de capteurs homogènes à faible ressources et un ou plusieurs nœuds puits.

Bien que les WSNs soient utilisés dans plusieurs applications, nous constatons l'apparition des nouvelles applications où nous avons besoin d'un nouveau composant du réseau appelé actionneur. Cette extension des réseaux de capteurs inclut les nœuds actionneurs qui ne sont pas considérés seulement comme responsables pour agir sur l'environnement mais d'un point de vue réseau, ils sont considérés comme des points de collecte locaux. Cette architecture est appelée réseau sans fil de capteurs et d'actionneurs (WSANs *Wireless Sensors and Actuators Networks*). La différence majeure entre les WSANs et les WSNs est que les WSANs sont des réseaux hétérogènes par nature.

Dans ce chapitre nous commençons par introduire les réseaux homogènes WSNs, leurs défis et leurs applications. Nous décrivons par la suite les réseaux hétérogènes WSANs, leurs architectures et applications. Enfin nous introduisons les défis pour les réseaux WSANs au niveau des différentes couches du modèles OSI et plus particulièrement les défis de la couche réseau.

2.2 Les réseaux homogènes

Les réseaux homogènes WSNs sont différents des réseaux sans fils classiques et des réseaux ad-hoc. Les nœuds dans un WSN sont généralement à faible ressources déployés dans une zone géographique donnée. Ces nœuds capteurs sont des nœuds homogènes qui une fois déployés, ils s'auto-organisent et s'auto-configurent pour constituer un réseau.

Vu que la capacité énergétique d'un capteur est limitée, la portée de transmission est également limitée. Ainsi, les réseaux de capteurs fonctionnent généralement avec un mode multi-sauts. Cette caractéristique est aussi parfois imposée par la topologie du réseau. Un nœud ayant des informations à envoyer vers le nœud puits doit les envoyer vers ses voisins pour que ces derniers relaient ces informations jusqu'à arriver à la destination finale.

La plupart des scénarii envisagés pour les WSNs contiennent un seul nœud puits et un grand nombre de nœuds capteurs homogènes. Le puits recueille des informations auprès des nœuds capteurs, les analyse puis les traite. Les capteurs sont supposés des nœuds homogènes ayant les mêmes caractéristiques matérielles (même capacité énergétique, capacité de calcul, portée de communication, etc.).

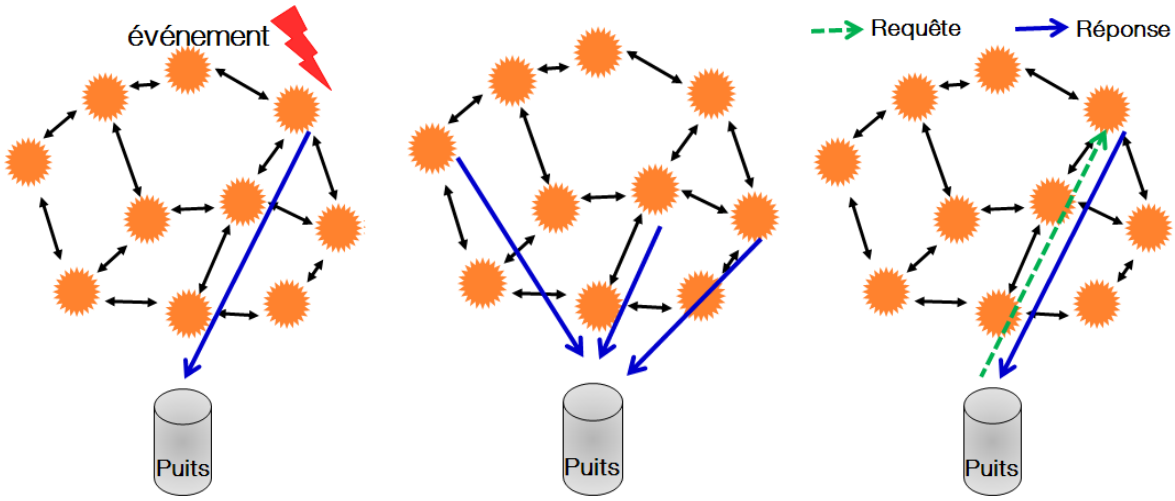
Nous pouvons distinguer deux principales familles d'applications pour les WSNs.

- Ces réseaux peuvent être utilisés pour assurer une récolte de données fiable.
- Les WSNs sont utilisés aussi pour assurer une couverture d'une zone géographique bien déterminée. Ces applications, généralement des applications de surveillance, visent à assurer une

couverture géographique de la zone où les capteurs sont déployés.

Il existe trois modes de communication pour les réseaux de capteurs :

- Le mode événementiel : les capteurs envoient vers le puits des données dès qu'un événement précis a été détecté (voir figure 2.1(a)). Par exemple, pour une application de gestion de déchets et de conteneurs de verres [1]. Quand les capteurs détectent un dépassement d'un seuil de remplissage d'un conteneur, ils envoient cet évènement détecté vers le nœud puits. Ce dernier va informer les autorités de la commune pour envoyer un camion pour décharger ce conteneur.
- Le mode périodique : les capteurs envoient périodiquement des informations de l'environnement vers le nœud puits (voir figure 2.1(b)). Un exemple d'application dans le monde agricole [138]. Les capteurs sont déployés dans un champ pour surveiller l'état du sol. Ils envoient périodiquement des informations sur l'état du sol pour optimiser les apports d'eau et de nutriments.
- Le mode à la demande : le puits envoie une requête à un ou un ensemble de nœuds demandant de lui envoyer des données (voir figure 2.1(c)). Un exemple typique de ces applications est la télérelève de compteur d'eau [2]. Dans ces applications des capteurs sont déployés avec les compteurs d'eau pour surveiller la consommation des clients. Le nœud puits peut être présenté comme un technicien ayant un appareil de collecte de donnée. Une fois devant l'immeuble où il doit collecter des informations, le technicien envoie avec son nœud puits une demande de collecte d'information. Quand la demande est reçu par le(s) nœud(s) destination, ce(s) dernier(s) va (vont) répondre en envoyant les informations requises par le nœud puits.



(a) Mode de communication événementiel

(b) Mode de communication périodique

(c) Mode de communication à la demande

FIGURE 2.1: Les différents modes de communications pour les WSNs

2.3 Les réseaux hétérogènes

Bien que les WSNs soient employés dans des nombreuses applications, il y a un nombre croissant d'applications qui nécessite l'utilisation d'un autre type de nœud dans le réseau, ce dernier est le

1. Ijinus : http ://www.ijinus.com/fr/applications
2. m2oCity : http ://www.m2ocity.com/

nœud actionneur. Ainsi, un WSAN est un réseau hétérogène par nature puisqu'il est constitué au moins de deux types de nœuds : les nœuds capteurs qui collectent les données de l'environnement où ils sont déployés, et les nœuds actionneurs qui ne sont pas responsables seulement d'agir sur l'environnement physique mais aussi, d'un point de vue réseau, sont responsables de recueillir les données récoltées par les nœuds capteurs. Nous nous intéressons à ce rôle joué par les nœuds actionneurs qui est le rôle de points de collecte locaux.

Ainsi des communications sans fil entre les nœuds capteurs et les nœuds actionneurs se produisent. Un WSAN doit être un réseau auto-organisé et autonome pour répondre aux exigences des applications visées par ce réseau. Dans les applications typiques des WSANs, les différents nœuds du réseau (capteurs et actionneurs) sont statiques. Cependant, il y a des applications qui supposent une mobilité au niveau des nœuds actionneurs. Comparés aux nœuds capteurs, les actionneurs sont des nœuds riches en ressources, ayant une grande capacité de calcul, de mémoire et une puissance de transmission importante.

Cette coexistence entre des nœuds à faibles ressources (capteurs) et les nœuds riches en ressources (actionneurs) introduit des nouveaux défis en liaison avec l'hétérogénéité.

2.3.1 Caractéristiques des réseaux hétérogènes WSANs

Un exemple typique des WSANs est le cas d'extinction automatique des incendies : les capteurs qui détectent un déclenchement de feu dans leurs environnements font remonter cette information aux systèmes d'extinction. L'information devrait rapidement arriver à (aux) l'actionneur(s) qui devrai(en)t prendre la décision afin que le feu puisse être facilement éteint avant qu'il ne devienne incontrôlable.

Ainsi, il y a des besoins de communication et de collaboration entre les nœuds capteurs et les nœuds actionneurs. Comme les actionneurs ont un rôle important dans le réseau et sont généralement des nœuds riches en ressource, ils peuvent contenir des antennes de meilleure qualité avec une puissance de transmission élevée. Ainsi les communications entre les actionneurs peuvent être à longue distance, contrairement aux communications entre capteurs ou entre les capteurs et les actionneurs [104]. C'est ainsi que dans cette thèse nous nous intéressons principalement aux communications entre les nœuds capteurs et les nœuds actionneurs.

Cette communication entre des nœuds hétérogènes ouvre une opportunité pour de nouveaux algorithmes et des protocoles prenant en compte et tirant profit de cette hétérogénéité.

2.3.2 Architectures de communication WSANs

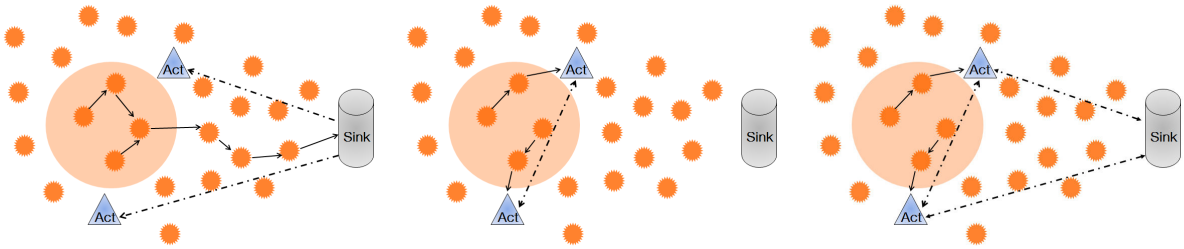
Il y a trois architectures de communications dans les réseaux WSANs. Deux architectures de base [6], une architecture appelée semi-automatique et une architecture automatique, et plus récemment une troisième appelée architecture coopérative [141].

Dans la première architecture les capteurs envoient leurs données vers le nœud puits, ce dernier choisit lui même l'actionneur le plus approprié à qui envoyer ces données (Figure 2.2(a)).

Dans la seconde architecture, les capteurs envoient les données directement vers les actionneurs qui traitent toutes les données entrantes, initient les actions appropriées et/ou vont faire suivre ces données vers le nœud puits (Figure 2.2(b)).

Dans la troisième architecture, les capteurs transmettent les données vers les nœuds actionneurs via un ou plusieurs sauts. Les actionneurs analysent les données et peuvent consulter le(s) nœud(s) puits avant de prendre toute action. Les actionneurs peuvent utiliser leur réseau point-à-point pour

prendre des décisions et prendre des mesures d'action, ou peuvent simplement informer le puits et attendre des nouvelles instructions de la part de ce dernier (Figure 2.2(c)).



(a) Architecture semi-automatique (b) Architecture automatique (c) Architecture coopérative

FIGURE 2.2: Les architectures de communication dans les WSANs

Nous nous concentrons dans nos travaux sur les deux architectures de base et plus particulièrement sur l'architecture automatique. En effet, en comparant les deux architectures de base, l'architecture automatique a deux avantages principaux :

- **Faible temps de latence** : Les informations détectées sont transmises par des capteurs directement aux actionneurs vu qu'ils peuvent être proches géographiquement les uns des autres.
- **Durée de vie du réseau plus longue** : Pour l'architecture semi-automatique, les capteurs autour du puits sont plus susceptibles de consommer plus d'énergie que les autres nœuds dans le réseau, car ils peuvent avoir une plus grande charge pour relayer les messages à destination du puits.

Cependant, cette architecture automatique nécessite que les nœuds capteurs soient au courant de la présence de ces nœuds actionneurs ce qui exige une phase d'auto-configuration et d'auto-organisation comme nous verrons dans le chapitre 4. De plus, la communication entre ces deux types de nœuds doit éviter les problèmes causés par la différence de portée de transmission. En effet, vu que les nœuds actionneurs ont généralement une grande portée de transmission par rapport à celle des nœuds capteurs, ces derniers doivent par exemple éviter les problèmes d'utilisation de liens unidirectionnels qui se présentent dans le réseau.

2.4 Défis des couches PHY/MAC pour les WSANs

Certains travaux, comme dans [93], considèrent que les WSANs sont l'union des WSNs et des réseaux Ad-hoc (les actionneurs), donc les problèmes et les défis pour les réseaux WSNs et Ad-hoc existent aussi dans les réseaux WSANs. De plus, l'hétérogénéité dans les WSANs introduit aussi des nouveaux problèmes et défis. En effet, les communications entre les nœuds actionneurs qui sont à longues distances peuvent provoquer des interférences au niveau des communications entre les nœuds capteurs. Nous ne considérons pas ce cas, puisque nous supposons que les nœuds actionneurs utilisent un autre réseau tiers pour communiquer entre eux.

Au niveau de la couche Physique, il est nécessaire de traiter des signaux, faire face à la défaillance matérielle des nœuds, de gérer une largeur de bande et une puissance limitée, etc. De plus la communication sans fil implique du bruit, des pertes de l'évanouissement et de l'atténuation des signaux [17].

Egalement, les nœuds sont soumis à un risque physique, car ils peuvent être défectueux, perdu, endommagé ou compromis. La communication sans fil implique une bande passante limitée et dans la plupart des cas également une capacité énergétique limitée (à moins que la batterie soit rechargeable [10], ou le capteur fonctionne avec de l'énergie solaire [74] ou d'autres alternatives d'approvisionnement énergétique [129]). La communication sans fil implique également la communication en diffusion où les messages envoyés par un nœud sont simultanément reçus par tous les voisins dans son voisinage radio (propriété naturelle du medium radio).

En ce qui concerne les défis au niveau de la couche MAC pour les WSANs, ils ne diffèrent pas de ceux pour les WSNs. En effet, la couche MAC vise essentiellement à assurer une efficacité énergétique au niveau des nœuds capteurs (puisqu'ils sont contraints en énergie) et à éviter les collisions.

De nos jours, diverses technologies normalisées à courte portée sont disponibles, dont certaines sont adaptées aux WSANs et d'autres qui ne le sont pas.

Les technologies qui ne sont pas adaptées sont : l'identification par radio-fréquence (RFID) [3] ainsi que le Bluetooth [4] (en s'appuyant sur la norme IEEE 802.15.1) parce que ces deux technologies ne supportent pas la communication multi-sauts, et la technologie Wifi [5] (en s'appuyant sur la norme IEEE 802.11 x) parce qu'elle consomme beaucoup d'énergie et elle n'est pas donc adapté pour les nœuds à faible ressource.

Parmi les technologies qui sont adaptées aux WSNs et aux WSANs, nous pouvons citer Zigbee [6] et Wavenis [7].

L'alliance ZigBee vise à développer des protocoles qui sont adaptés aux applications embarquées nécessitant un faible débit de données et une faible consommation d'énergie. C'est aujourd'hui le plus grand groupe de normalisation pour les WSNs et les WSANs. Les couches PHY et MAC se basent sur la norme IEEE 802.15.4 [137] et les nœuds ZigBee fonctionnent dans les bandes ISM. L'utilisation de cette bande ISM expose le réseau à des interférences, parce que ces bandes ISM sont aujourd'hui très chargées avec les réseaux WiFi.

Wavenis a été développée par Coronis [8] dans le cadre de leur technologie facilitant la télé-relève des compteurs (d'eau, d'électricité et de gaz). Wavenis définit seulement les couches basses du modèle OSI (les couches Physique, Liaison de données et Réseau). Wavenis opère dans les bandes de fréquences ISM. Vu les applications visées, Wavenis a été conçue pour assurer la transmission des petits volumes de données, de quelques octets à quelques centaines d'octets avec de l'ordre de 19,2 kilobits par seconde. Grâce à ce faible débit, Wavenis permet d'assurer une portée de communication qui peut atteindre quelques centaines de mètres avec une consommation d'énergie très faible. Coronis a annoncé en 2008, l'ouverture de leur Wavenis Open Standard Alliance, qui pourrait aboutir à normaliser des solutions de communication ultra faible puissance.

Dans le tableau 2.1, nous comparons les deux principales technologies dédiées pour les WSANs à savoir Wavenis et Zigbee. La technologie Wavenis permet d'avoir une une portée de communication importante comparée à celle assurée par Zigbee, de plus elle permet d'avoir une durée de vie supérieure à celle de Zigbee.

3. RFID : http ://www.rfidfr.org
4. Bluetooth : https ://www.bluetooth.org
5. Wifi Alliance : http ://www.wi-fi.org
6. ZigBee Alliance : https ://www.zigbee.org
7. Wavenis Open Standard Alliance : http ://www.wavenis-osa.org
8. Coronis : http ://www.coronis.com

Paramètre	Wavenis	Zigbee
Bandes de fréquence (Europe)	868 MHz	868 MHz
Débit effectif	19,2 Kbps	250 Kbps
Autonomie de la pile (typique)	10 ans	3 ans
Portée	200 m à l'intérieur 1 km à l'extérieur	20 m

TABLE 2.1: Comparaison Wavenis et Zigbee

2.5 Défis de la couche réseau pour les WSANs

Nous pouvons classer les défis au niveau de la couche réseau en deux catégories : les défis de contrôle de topologie et les défis du routage.

2.5.1 Contrôle de topologie

Définition

Comme dans les WSNs, la densité et la redondance des nœuds, dans les WSANs, est importante. Chaque nœud peut détecter donc un ensemble conséquent de nœud voisin. La prise de connaissance de tous les nœuds dans sa zone de communication et le maintien des informations concernant ces voisins s'avèrent coûteux en énergie et aussi en espace mémoire.

D'un point de vue macroscopique, un protocole de contrôle de topologie est utilisé conjointement avec un protocole de routage dans l'objectif d'améliorer les performances de ce dernier.

D'un point de vue microscopique, un protocole de contrôle de topologie a pour but de gérer les relations de voisinage entre un nœud et tous les nœuds dans sa zone de communication.

Défis de contrôle de topologies dans les réseaux hétérogènes

Le but des protocoles de contrôle de topologie consiste à gérer les relations de voisinage en éliminant des liens considérés comme redondants ou en ignorant des nœuds voisins considérés comme « inutiles ».

Les principales techniques utilisées sont :

- L'établissement d'une hiérarchie où un nœud est désigné comme responsable d'un ensemble de nœuds dans son voisinage. Ce nœud assurera la gestion des communications de ses voisins avec les autres nœuds du réseau.
- L'ajustement des portées de communication des nœuds, afin de réduire les zones de communication et par la suite ignorer un ensemble de nœuds dans le voisinage.

1. **Mise en place d'une hiérarchie**

   La mise en place d'une hiérarchie comprend l'établissement de clusters ou d'un ensemble dominants.

   La technique de création des clusters [1] consiste à répartir les nœuds du réseau en des ensembles disjoints appelés cluster. Le Leader de chaque cluster est appelé un Cluster-Head (CH). Ce CH peut être élu par les nœuds du cluster ou pré-attribué par le concepteur du réseau (dans certains cas, les nœuds actionneurs, riches en ressources, sont désignés à jouer

le rôle de CH dans les WSANs). Nous allons discuter les techniques de choix de CHs dans le chapitre suivant.

D'autres algorithmes de contrôle de topologie proposent de sélectionner un certain nombre de nœuds du réseau, appelé nœuds dominants, pour créer une structure connectée en épine dorsale [20].

Ces deux techniques sont utilisées pour réduire la consommation énergétique [161] [11] et améliorer l'efficacité des protocoles de communication et plus particulièrement les protocoles de routage.

Dans les réseaux hétérogènes, tels que les WSANs, plusieurs travaux portent sur la conception d'algorithmes de création de clusters ou des ensembles de nœuds dominants permettant de profiter de cette hétérogénéité afin d'augmenter la durée de vie du réseau [80].

La technique de création d'une hiérarchie a plusieurs avantages, en effet :

- Elle réduit la taille de la table de routage stockée au niveau des nœuds [4].
- Elle peut économiser la bande passante de communication : en effet, pour la technique basée sur la création de clusters, le champ de communication des nœuds dans le même cluster [161].
- Elle peut prolonger la durée de vie des nœuds à faibles ressources et ainsi la durée de vie du réseau [161] [96].
- Elle réduit le surcoût de maintenance de la topologie du réseau : puisqu'un nœud surveillera seulement sa connexion avec son CH [73] ou avec le nœud dominant de l'épine dorsale.
- Elle permet de réduire le nombre de paquets redondant en utilisant des techniques d'agrégations des données au niveau des nœuds dominants ou au niveau des CHs [41].

2. **Ajustement de la puissance de transmission**

La deuxième famille de protocole de contrôle de topologie est la technique de contrôle de puissance d'émission. Cette technique a pour but de rendre les liens existant dans le réseau plus efficaces et moins couteux en termes de consommation énergétique.

Le contrôle de la puissance d'émission peut se faire à quatre niveaux : au niveau du réseau, du nœud, de chaque voisin ou de chaque paquet.

- Au niveau du réseau, le but est de calculer une portée de transmission adéquate pour assurer une connectivité prédéfini du réseau.
- Quand l'ajustement de la portée de transmission se fait au niveau du nœud, l'objectif est de calculer pour chaque nœud du réseau la portée optimale afin de communiquer avec ses voisins.
- Le choix d'une portée pour chaque voisin permet au nœud de choisir une portée de transmission différente selon le voisin avec lequel il souhaite communiquer.
- Enfin, la puissance d'émission peut être ajustée pour chaque paquet envoyé.
  Ces derniers points ont pour objectifs de minimiser les interférences et/ou la consommation énergétique tout en préservant les caractéristiques de connectivité prédéfinies par le réseau.

Les avantages de la technique d'ajustement des portées de transmission, nous pouvons citer :

- Réduire des interférences et augmentations de la capacités du réseau [61].
- Economiser la consommation énergétique [38] : en effet en appliquant un ajustement de la portée de transmission au niveau du nœud, il y aura une conservation de l'énergie au niveau du nœud émetteur (moins d'énergie nécessaire pour envoyer un paquet) et au niveau du nœud récepteur (suite à l'ajustement de la portée, des nœuds seront hors de portée de l'émetteur et ne traiteront plus des paquets qui ne leur sont pas destinés).

Dans le chapitre suivant, nous présentons les principaux protocoles de contrôle de topologie dédiés aux réseaux homogènes et hétérogènes.

2.5.2 Routage

Définition

Le routage est l'une des questions cruciales dans presque n'importe quel type de réseau. L'objectif d'un protocole de routage est de trouver une route entre deux nœuds du réseau.

Pour les réseaux hétérogènes, ils exigent aussi, en même temps, la minimisation des surcoûts de communication, de la consommation d'énergie et la prise en compte de la diversité des capacités des nœuds constituants le réseau.

Comme dans les réseaux de communications sans fil classiques, l'objectif principal du routage, du routage avec QoS, du multicast et du broadcast est d'assurer l'acheminement des paquets avec succès entre les nœuds du réseau. Pour le routage avec QoS, les chemins choisis par l'algorithme de routage doivent répondre aux critères de qualité de service (tels que le délai, débit, etc.). Comme dans les réseaux sans fil classiques, le multicast et le broadcast (inondation) représentent respectivement le fait qu'un même message doit être acheminé à partir d'un nœud source à un nombre fixe de k destinations connues ou à tous les nœuds du réseau.

Défis du routage dans les réseaux hétérogènes

La diversité des applications visées par les réseaux hétérogènes implique des exigences au niveau de routage qui sont différentes. Ainsi un protocole de routage doit répondre aux exigences des applications tout en respectant les contraintes de ressources au niveau des nœuds.

De plus, la nature décentralisée des WSANs complique toute tentative de maintenir une table de routage centralisée. Des mécanismes décentralisés doivent donc être utilisés. Cependant, pour tirer bénéfice des nœuds actionneurs riches en ressource, nous pouvons profiter de leur capacité de stockage à sauvegarder des tables de routage [63].

Comme dans les réseaux sans fil, les liens entre les nœuds sont volatiles ce qui implique une dynamique dans le réseau (apparition et disparition des nœuds). Ainsi les protocoles de routage doivent utiliser des mécanismes appropriés pour atténuer ce manque de fiabilité.

En outre, la capacité énergétique des nœuds à faibles ressources introduit des défis de conception des protocoles de routage dans les réseaux hétérogènes. En effet, les protocoles de routage doivent conserver de l'énergie au niveau des nœuds à faibles ressources et impliquer plus les nœuds riches en ressources dans le processus de routage.

Une autre particularité des réseaux hétérogènes WSANs est qu'ils nécessitent une coordination non seulement entre les capteurs ou entre les actionneurs, mais aussi entre les capteurs et les actionneurs. Pour faciliter la coordination, le premier problème est de parvenir à la sélection de l'actionneur approprié. Les nœuds capteurs ont besoin de savoir où et comment envoyer les informations au plus « proche »actionneur. L'utilisation du terme « proche »ici peut couvrir la proximité géographique ou électromagnétique. Les actionneurs peuvent inonder le réseau avec des messages annonçant leur position ou simplement leurs identifiants. Les capteurs recevant ces messages pourront éventuellement les rediffuser ou les ignorer si, par exemple, un actionneur plus proche a été déjà identifié. Ces algorithmes d'inondation de plusieurs actionneurs dans le réseau ont été discutés dans [76].

Ainsi les capteurs transmettent leurs informations vers l'actionneur approprié via des transmissions multi-sauts. Plusieurs scénarii supposent des actionneurs mobiles. Ces actionneurs mobiles,

des robots par exemple, peuvent se déplacer pour recueillir périodiquement des informations et pour répartir la consommation énergétique des nœuds capteurs [5] [50]. Nous ne détaillons pas les travaux qui concernent la mobilité dans les réseaux WSANs car c'est hors du périmètre de nos travaux.

Dans le chapitre suivant, nous détaillons les principales classes des protocoles de routage dans les réseaux WSANs.

2.6 Conclusion

Dans ce chapitre, nous avons pu voir les caractéristiques des réseaux homogènes, les WSNs comme exemple, et des réseaux hétérogènes, les WSANs comme exemple. Nous avons décrit les applications des réseaux WSNs. Ensuite, nous avons identifié les spécificités des WSANs, leurs architectures et leurs applications. Nous avons introduit en plus les défis pour les réseaux WSANs au niveau des différentes couches. Nous nous concentrons sur la couche réseau et principalement sur les défis de contrôle de topologie et les défis de routage. Dans le chapitre suivant nous faisons un état de l'art sur les principaux protocoles dédiés à la structuration et au routage dans les réseaux WSANs.

L'auto-organisation et le routage dans les réseaux WSANs

3

Sommaire

3.1 Introduction

Comme nous venons de voir dans le chapitre précédent, les WSNs et les WSANs sont des réseaux autonomes, spontanés et multi-sauts, où les nœuds collaborent afin d'assurer le bon fonctionnement du réseau.

La propagation des signaux radio dans ces réseaux est affectée par plusieurs facteurs qui contribuent à la dégradation de sa qualité. Par conséquent, les liaisons radio dans les WSNs et les WSANs sont souvent imprévisibles et volatiles. En fait, la qualité des liens varie au fil du temps [30] [139] et de l'espace [167] [165]. Ainsi, en raison de la volatilité des liens radios, la topologie physique du réseau change avec l'apparition et la disparition de ces liens radios. Ainsi, il y a un besoin de cacher cette dynamique au niveau des nœuds du réseau en créant une vue stable du voisinage radio. En outre, vu que les nœuds capteurs sont des nœuds contraints en énergie, une répartition de la charge de communication entre les nœuds est nécessaire. Pour les WSANs, les nœuds actionneurs, riches en ressources, doivent être impliqués d'avantage pour assurer le bon fonctionnement du réseau. Ainsi, créer une topologie logique permet de cacher les changements de la topologie (en créant une vue stable au niveau de chaque nœud), prendre en compte l'hétérogénéité (distribution de charge entre les nœuds riches en ressources et les nœuds à faibles ressources) et faciliter le déploiement des algorithmes de communication et les applications au dessus de cette topologie logique [143].

Dans ce chapitre, nous nous intéressons à la structuration du réseau pour faciliter le déploiement des algorithmes de routage. En effet, un protocole de routage joue un rôle clé dans réseau de communication. Son rôle est d'assurer l'acheminement des messages entre une source et une destination. Les caractéristiques spécifiques des WSNs et des WSANs sont différentes des caractéristiques des autres types de réseaux (que ce soit des réseaux filaires ou sans fil). En effet, la nature aléatoire des liens sans fil impacte fortement la fiabilité de la communication dans les réseaux WSNs et WSANs. De plus, l'absence de contrôle centralisé dans ces réseaux et les capacités limitées au niveau des nœuds capteurs compliquent la tâche du routage. Il est à noter également que dans les WSANs, la co-existence entre les nœuds capteurs, à faibles ressources, et les nœuds actionneurs, riches en ressources, a un impact sur les performances des protocoles de routage. En effet, l'hétérogénéité au niveau des portées de transmission des nœuds implique l'existence des liens asymétriques. Ces liens peuvent dégrader les performances des protocoles de routage qui ne prennent pas en considération ce type de liens.

Le problème de routage est donc un problème délicat. Le domaine de routage dans les réseaux WSNs et WSANs a bien attiré l'attention du monde scientifique pendant la dernière décennie jusqu'à atteindre un état de maturité qui a permi au groupe de travail ROLL [1] (*Routing Over Low power and Lossy networks*) de proposer un protocole en cours de standardisation. En effet, le groupe de travail ROLL a été créé pour normaliser un protocole de routage pour ces types de réseaux appelé RPL (*Routing Protocol for LLNs*) [155]. La principale raison qui a été derrière la création du groupe ROLL est que les exigences des réseaux LLNs (*Low Power and Lossy Networks*) ont évolué à un point où des solutions standardisées pour les réseaux ad-hoc ne s'appliquent plus [90]. Dans [90], les auteurs montrent qu'aucun protocole de routage standard au sein de l'IETF (à savoir OSPF [108], OLSRv2 [37], RIP [98], AODV [118], DYMO [31] et DSR [77]) ne peut répondre aux exigences des réseaux LLNs. Il est à noter que ces réseaux LLNs couvrent bien les WSNs et les WSANs.

Principalement, les protocoles de routages dans les LLNs doivent fonctionner sous un ensemble de contraintes que les protocoles de routage dédiés aux réseaux ad-hoc ne prennent généralement pas en compte. En effet, dans les LLNs, les nœuds sont contraints en ressources (capacité de calcul

1. ROLL IETF Working Group : http ://datatracker.ietf.org/wg/roll/charter/

et de stockage), les applications et les modèles de trafics sont aussi différents des réseaux ad-hoc et filaire [7]. De plus, l'énergie est un enjeu fondamental pour les LLNs : en effet, assurer une faible consommation d'énergie est une préoccupation majeure afin de permettre aux nœuds à faibles ressources de fonctionner pendant des mois et des années sans interruption. Et comme ces protocoles existants n'ont pas été conçus avec l'ensemble de ces contraintes à l'esprit, il s'avère qu'ils ne peuvent pas être utilisés dans les LLNs.

Dans ce chapitre, nous commençons par introduire et définir la notion d'auto-organisation. Ensuite, nous décrivons quelques techniques d'auto-organisation se basant sur la création d'une topologie logique au dessus sur une topologie physique. Nous décrivons par la suite les défis de routage dans les WSANs pour introduire à la fin les principaux protocoles de routage.

3.2 L'auto-organisation, rôle et objectifs

3.2.1 Définition

Le terme d'auto-organisation est utilisé dans de nombreuses branches de la science [13], [140], etc.

Un système constitué de plusieurs entités, est dit organisé s'il possède une structure et un ensemble de fonctions [140]. La structure vise à créer une relation entre les entités en les disposant d'une certaine manière particulière et à faciliter la communication entre ces entités. L'ensemble de fonctions a pour rôle de maintenir la structure et l'utilisation de celle-ci pour répondre à des besoins bien déterminés [121].

Un système est dit auto-organisé, s'il est organisé sans aucune entité externe et sans contrôle centralisé. Ainsi, les entités, qui constituent le système, doivent interagir localement avec d'autres entités d'une façon distribuée.

La notion d'auto-organisation a aussi trouvé sa place dans le domaine de la communication, des réseaux informatique et des réseaux WSNs et WSANs. Les nœuds capteurs et les nœuds actionneurs interagissent directement les uns avec les autres d'une manière distribuée avec un objectif commun. Ainsi, l'auto-organisation dans les WSNs et les WSANs est un processus duquel émerge une structure globale provenant seulement des interactions locales entre les nœuds du réseau [49].

3.2.2 Finalités de l'auto-organisation

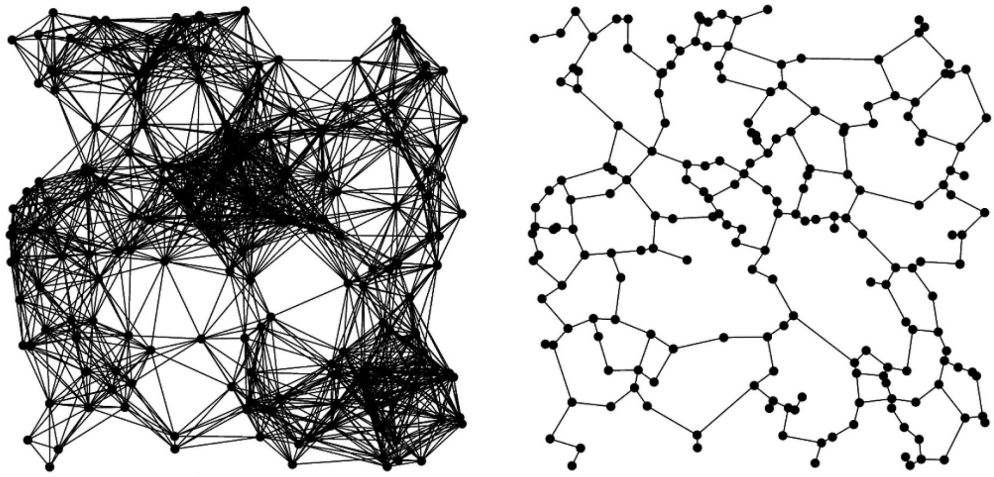
Un réseau auto-organisé crée une topologie logique au dessus de la topologie physique (voir Figure 3.1). Nous trouvons plusieurs types de topologies que nous détaillons dans la section 3.3.1.

La topologie logique est construite non pas seulement pour « cacher »la dynamique locale au niveau de chaque entité appartenant à la topologie (la dynamique couvre aussi bien l'apparition/disparition des liens et des nœuds) mais aussi pour faciliter le déploiement des protocoles de communication au dessus de cette topologie logique. Ainsi la topologie s'adapte aux changements de la topologie et se reconstruit dynamiquement et localement (dans le cas idéal). De plus, cette topologie logique doit être capable de supporter le passage à l'échelle qui est une caractéristique partagée entre les WSNs et les WSANs. Les principaux objectifs de la construction d'une topologie logique sont :

– **Minimiser la consommation énergétique au niveau du réseau** : Certaines topologies logiques permettent la mise en place d'un mécanisme d'endormissement de certains nœuds dans le réseau [75]. La mise en état d'endormissement de certains nœuds permet de conserver

de l'énergie dans le réseau. D'autres propositions d'auto-organisation permettent de définir une puissance de transmission pour chacun des nœuds afin de réduire le coût énergétique total du réseau [84], [125] et [113].

- **Améliorer les performances du réseau** : La création d'une topologie logique peut se baser sur des métriques de réputation de nœuds pour créer une topologie sécurisée [57]. Cette topologie logique peut aussi réduire les interférences et par la suite augmenter le débit du réseau [107].
- **Partager et gérer les ressources disponibles dans le réseau** : Un protocole d'auto-organisation doit faciliter le déploiement des protocoles de communication et principalement les protocoles de routage auxquels nous nous intéressons dans ce chapitre. En effet, un protocole de routage doit être plus efficace quand il est déployé dans un réseau structuré que lorsqu'il déployé sur un réseau à plat. En effet, une topologie logique permet de répartir les informations de routage entre les nœuds tout en prenant compte de leurs capacités de stockage limitées [40] et [144].



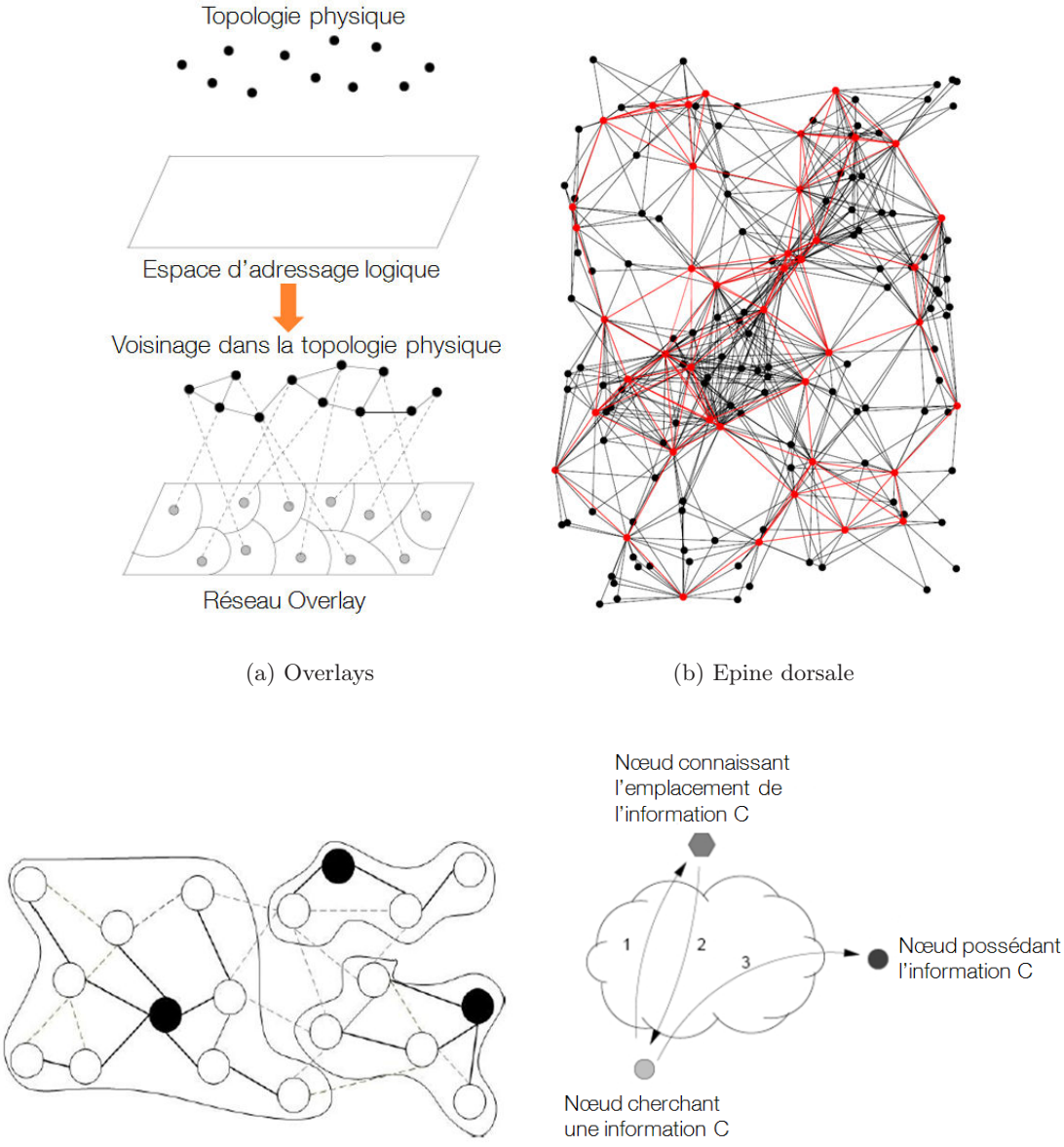
(a) Topologie physique (b) Topologie logique

FIGURE 3.1: Principe de création d'une topologie logique au dessus d'une topologie physique.

3.3 Techniques d'auto-organisation

Plusieurs types de topologies logiques existent parmi lesquelles nous trouvons les topologies basées sur les épines dorsales virtuelles [15], [25], [27], sur la création d'*overlays* [26], sur les techniques de *clusters* [15] et sur les tables de hachage distribuées [151] (voir figure 3.2).

Dans ce chapitre, nous ne nous intéressons pas à la structuration en *clusters* consistant à découper le réseau en régions organisées autour de nœuds spécifiques appelés *Cluster-Heads* (CHs).En effet, les problématiques rencontrées dans les topologies basées sur les *clusters* et les *overlays* se reproduisent dans les topologies basées sur les épines dorsales virtuelles. De plus, les tables de hachage distribuées utilisées dans les réseaux filaires pair-à-pair ne peuvent pas être utilisées pour des réseaux sans fil étendus et contraints en ressources. En effet, le coût de distribution et d'interrogation des tables de



(a) Overlays

(b) Epine dorsale

(c) Cluster

(d) Tables de hachage distribuées

FIGURE 3.2: Les 4 stratégies principales d'auto-organisation.

hachage sur l'ensemble d'un réseau de capteurs et actionneurs est élevé pour un tel réseau contraint en énergie.

Ainsi dans cette section, nous nous intéressons aux protocoles d'auto-organisation qui construisent des topologies basées sur les épines dorsales.

3.3.1 Les principales topologies logiques

Nous nous concentrons dans cette section sur deux grandes familles de protocoles d'auto-organisation pour la construction des épines dorsales virtuelles : les protocoles basés sur la mise en place d'une hiérarchie et les protocoles basés sur l'élagage des liens en s'appuyant sur l'ajustement de la puissance de transmission.

Les objectifs des différentes topologies logiques sont multiples, mais l'idée directrice est la même : créer une topologie logique maitrisée (figure 3.1(b)) au dessus d'une topologie physique imposée (figure 3.1(a)).

1. Topologies basées sur la mise en place d'une hiérarchie

La technique de mise en place d'une hiérarchie se base sur l'affectation des états au niveau des nœuds du réseau : Nous trouvons ainsi des nœuds dominants et des nœuds dominés. L'ensemble des nœuds dominants est de sorte que tous les nœuds du réseau sont, soit dans cet ensemble, soit voisin d'un nœud dominant de cet ensemble. L'ensemble de nœuds dominants est dit connecté si et seulement s'il existe un chemin entre chaque paire de nœuds composant cet ensemble. Les principaux protocoles d'auto-organisation basés sur la mise en place d'une hiérarchie sont : CDS-règle k [156] (*Connected Dominating Set-règle k*), CDS-IDS [117] (*Connected Dominating Set-Independent Dominating Set*), Legos [97] (*Low Energy Self-OrGanisatiOn Scheme*), MPR [122] (*Multi Point Relay*) et MPR-DS [2] (*Multi Point Relay-Dominating Set*).

Connected Dominating Set-règle k (CDS règle-k)

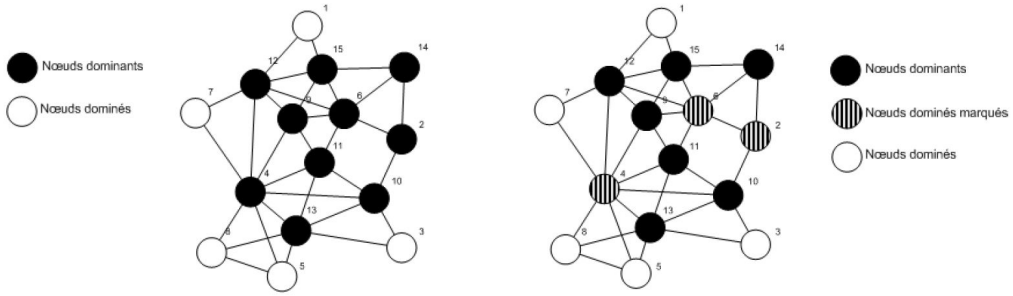
Soit S, un ensemble connecté dominant (CDS) [19]. S est défini comme un sous-ensemble de l'ensemble des sommets V, qui satisfait les trois points suivants :

- Chaque nœud de l'ensemble V est soit dominant (dans l'ensemble S), soit dominé.
- Un nœud dominé a un voisin à un saut appartenant à l'ensemble S.
- L'ensemble S est connecté.

Le CDS-règle k [156] permet la construction localisée d'un CDS en deux phases : une phase de marquage et une phase d'élagage de la règle k.

- Durant la première phase de marquage, chaque nœud se marque quand il possède deux voisins non-connectés entre eux (voir figure 3.3(a)).
- Après cette première phase, le processus d'élagage de la règle k est appliqué par chaque nœud marqué. Si l'ensemble des voisins d'un nœud dominant sont couverts par un ensemble dominant de k nœuds voisins, et que ce nœud a le plus petit identifiant, alors ce dernier devient un nœud dominé (voir figure 3.3(b)).

L'algorithme de CDS-règle k est complètement localisé. Il définit un ensemble connecté dominant de plus fort poids dans le voisinage (voir figure 3.3). Chaque nœud appliquant cet algorithme a besoin des informations sur son voisinage à deux sauts. Dans cette optique, le recours aux paquets « *Hello* »est indispensable.



(a) Phase de marquage (b) Phase d'élagage

FIGURE 3.3: Principe du protocole CDS-règle *k*.

Connected Dominating Set-Independent Dominating Set (CDS-IDS)

Un ensemble dominant indépendant (IDS) est un ensemble dominant où chaque nœud n'est pas adjacent à aucun autre nœud de l'ensemble. Ainsi un nœud appartenant à un IDS, est éloigné d'au moins deux sauts et d'au plus trois sauts d'un autre nœud dominant (voir figure 3.4).

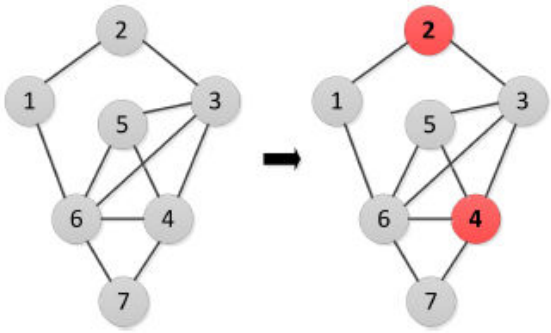


FIGURE 3.4: Exemple d'un IDS.

Dans [117], les auteurs proposent une version distribuée de construction d'un CDS. L'algorithme dans [117] est un algorithme distribué composé de trois étapes, il construit :

1. un arbre couvrant.
2. un ensemble indépendant dominant (IDS).
3. un arbre dominant.

L'IDS construit par l'algorithme dans [117] est de sorte que chaque nœud de l'IDS est à exactement deux sauts d'un autre nœud de l'IDS. Les auteurs utilisent pour cela une notion de rang utilisée pour faire un ordonnancement total parmi les nœuds. Ce rang est calculé en se basant sur le nombre de sauts séparant chaque nœud à la racine de l'arbre et sur l'identifiant unique de chaque nœud.

L'IDS est construit à partir du nœud racine. Ce dernier se déclare « noir »et envoie un message « *BLACK* ». Les nœuds recevant ce message « *BLACK* »se déclarent « gris »et envoient un message

« *GREY* ». Un nœud recevant un message « *GREY* »de l'ensemble de ses voisins de rang inférieur, se déclare « noir »et envoie un message « *BLACK* ».

Les nœuds déclarés « noirs »appartiennent à l'IDS. Le CDS est construit aussi à partir du nœud racine, les nœuds « noirs »désignent un parent de rang inférieur. Les parents de tous les nœuds de l'IDS seront considérés comme des nœuds dominants : ils forment avec les nœuds de l'IDS l'ensemble dominant connecté.

Low Energy Self-OrGanisatiOn Scheme (Legos)

Legos (*Low Energy Self-OrGanisatiOn Scheme*) [97] est un protocole d'auto-organisation visant à générer une topologie logique en épine dorsale non-orienté sous l'hypothèse d'un déploiement progressif ou sous l'hypothèse que les nœuds ne se réveillent pas simultanément pour rejoindre la topologie logique.

Avec Legos, les nœuds sont dans l'un des trois états suivants : *Leader* (L), *Gateway* (G) ou *Member* (M). Les nœuds *Members* sont des nœuds dominés et les nœuds *Leaders* et *Gateways* sont des nœuds dominants. Chaque nœud *Member* est relié à un seul *Leader*. Ce dernier est en charge de toutes les communications dans son voisinage à 1 saut. Les *Leaders* sont espacés de 2 sauts. Un nœud *Gateway* est en charge de l'interconnexion des deux ou plusieurs nœuds *Leaders* (voir figure 3.5). Les *Leaders* diffusent périodiquement des messages pour annoncer leurs présences. Un nœud, détectant un *Leader* dans son voisinage, s'attache à lui. Une fois attaché à un *Leader*, les communications entrantes et sortantes de ce nœud sont gérées par ce *Leader*.

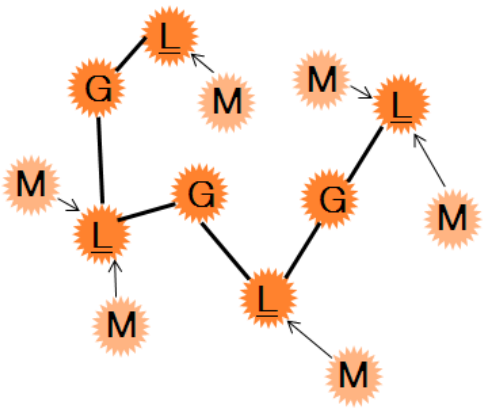


FIGURE 3.5: Topologie Legos.

Multi Point Relay (MPR)

Dans [122], les auteurs développent un algorithme de calcul d'un ensemble dominant connecté afin de réduire le nombre de retransmissions lors d'une inondation appelé l'ensemble MPR. Un nœud source sélectionne dans son voisinage direct un sous-ensemble de nœuds couvrant l'ensemble de ses voisins à 2 sauts, ce sont ses voisins MPR (voir figure 3.6)

[37] et [112] proposent deux algorithmes différents pour calculer l'ensemble MPR d'un nœud : MPR glouton et MPR Min-id.

L'algorithme MPR glouton [37] se compose de deux étapes. Pour un nœud donné, la première étape consiste à chercher dans le voisinage à deux sauts les nœuds qui ne sont connectés que par un seul voisin direct et intégrer dans le sous-ensemble MPR ces voisins connecteurs. Ensuite,

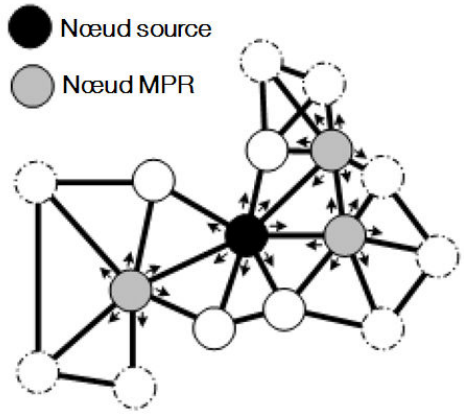


FIGURE 3.6: Mécanisme de désignation des voisins MPRs.

la deuxième étape consiste à intégrer dans le sous-ensemble MPR en priorité les voisins directs couvrants le plus de voisins à 2 sauts non encore couverts.

Pour l'algorithme MPR Min-id [112], les voisins sont explorés et une décision est prise quant à leurs ajouts dans l'ensemble MPR suivant l'ordre croissant de leurs identifiants. L'avantage de ce dernier algorithme est qu'un nœud peut savoir s'il appartient à l'ensemble MPR d'un voisin. Cependant, [2] montre que le calcul de l'ensemble MPR pour le MPR Min-id est d'une complexité cubique (en $O(m^3)$), contre une complexité quadratique (en $O(m^2)$) pour le MPR glouton (où m représente le degré maximal d'un nœud). Les auteurs de [2] montrent que l'algorithme MPR Min-id est bien moins efficace que le MPR glouton en termes de cardinalité de l'ensemble MPR.

Pour construire un ensemble qui soit connecté et dominant, chaque nœud de l'ensemble MPR désigne à son tour ses nœuds MPR dans son propre voisinage. La désignation des nœuds MPR est localisée puisqu'un nœud a besoin d'une connaissance du voisinage à deux sauts pour choisir ses voisins MPR. Par contre, la construction de l'ensemble dominant est distribuée puisqu'elle dépend du nœud source qui initie la construction.

Cette topologie est utilisée dans le protocole de routage OLSR (*Optimized Link State Routing*) [37] standardisé par l'IETF.

Cette technique de construction de MPR peut être utilisée dans les WSNs et les WSANs pour assurer une dissémination de donnée dans le réseau. Dans le chapitre suivant, nous évaluons la consommation énergétique de notre proposition avec celle d'une structuration avec MPR.

Multi Point Relay-Dominating Set (MPR-DS)

Le *Multi Point Relay-Dominating Set* (MPR-DS) [2] utilise le même mode d'élection décrit ci-dessus. Cependant, cet algorithme est totalement localisé et non-orienté source. En effet, l'élection des nœuds se base sur les identifiants des nœuds. Un nœud va rejoindre l'ensemble dominant si son identifiant est le plus petit que tous ses voisins directs ou si ce nœud est désigné dominant par le voisin de plus petit identifiant.

2. Topologies basées sur l'élagage des liens par ajustement de la puissance de transmission

La deuxième famille de protocole d'auto-organisation est la famille des protocoles basés sur l'élagage des liens en s'appuyant sur l'ajustement de la puissance de transmission. Ces protocoles visent à éliminer les liens redondants et inutiles dans le réseau (voir figure 3.10). Les principaux protocoles d'auto-organisation basés sur l'élagage des liens sont : RNG [147] (*Relative Neighborhood Graph*), GG [56] (*Gabriel Graph*) et LMST [92] (*Local Minimum Spanning Tree*).

Relative Neighborhood Graph (RNG)

Relative Neighborhood Graph (RNG) [147] est un une famille de graphe proposée par Toussaint en 1980. L'idée est d'élaguer l'arête la plus longue dans chaque triangle dans le graphe. Comme le montre la figure 3.7, l'arête (U, V) n'est pas dans le sous-graphe RNG car c'est l'arête la plus longue du triangle (U, V, W). L'idée de construction d'une topologie logique basée sur le graphe RNG est d'économiser les liens les plus longs en les élaguant et dons en émmettant à plus faible puissance.

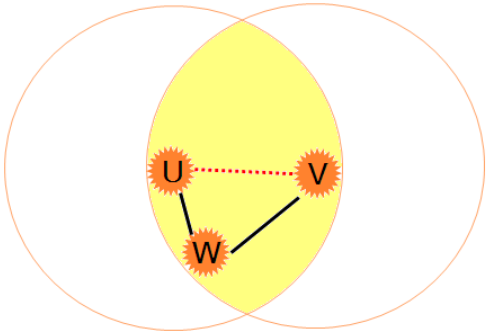


FIGURE 3.7: Élagage des liens avec RNG.

La construction du graphe RNG se repose sur la connaissance du voisinage direct. Ainsi cette construction est basée sur un algorithme localisé. Celui-ci respecte les propriétés de connexité du graphe initial. En d'autres termes si le graphe initial est connexe alors le graphe RNG l'est également.

Graphe de Gabriel (GG)

Le graphe de Gabriel est introduit dans [56]. En se référant à la figure 3.8, le lien (U, V) entre le nœud U et le nœud V est éliminé si dans le disque de diamètre (U, V) il y a un autre nœud W connecté aux deux nœuds U et V. La zone d'exclusion d'un GG est incluse dans la zone d'exclusion d'un RNG (voir figure 3.9), ainsi une topologie RNG est un sous-graphe de la topologie GG. La topologie GG conserve donc, comme RNG, les propriétés de connexité du réseau.

Local Minimum Spanning Tree (LMST)

Les algorithmes du *Minimum Spanning Tree* (MST) ont été largement utilisés pour la diffusion dans le réseau et pour le routage. Cependant, la détermination du MST nécessite une connaissance complète de la topologie du réseau. Cette connaissance ne peut pas être envisagée dans les réseaux WSNs ou les WSANs. [92] propose une version de MST locale (*Local MST* ou LMST) dans laquelle chaque nœud détermine localement un ST sur son voisinage à deux sauts. Pour cela, les nœuds doivent s'échanger des paquets « *Hello* »contenant la liste des voisins directs. Si deux nœuds sont voisins dans leurs topologies logiques de MST respectives, alors l'arête entre les deux nœuds est ajoutée dans le sous-graphe du LMST. Il est à souligner que le sous-graphe résultant n'est pas un arbre. Il existe des boucles à cause de l'exécution localisée de l'algorithme. Il a été rappelé dans [28]

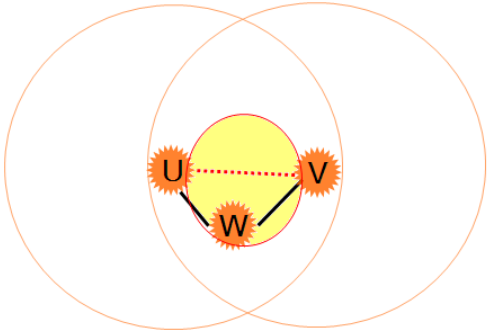
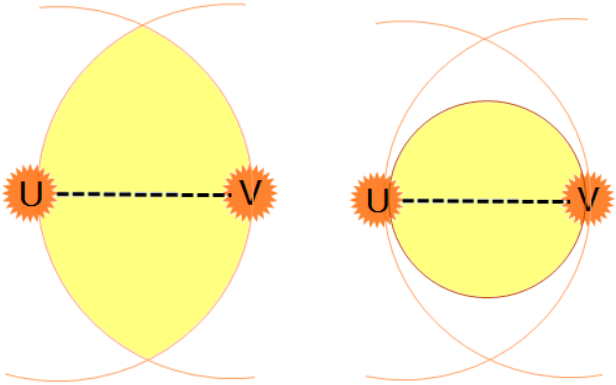


FIGURE 3.8: Élagage du lien pour le Graphe de Gabriel.



(a) RNG (b) GG

FIGURE 3.9: Comparaison de la zone d'exclusion des topologies RNG et GG.

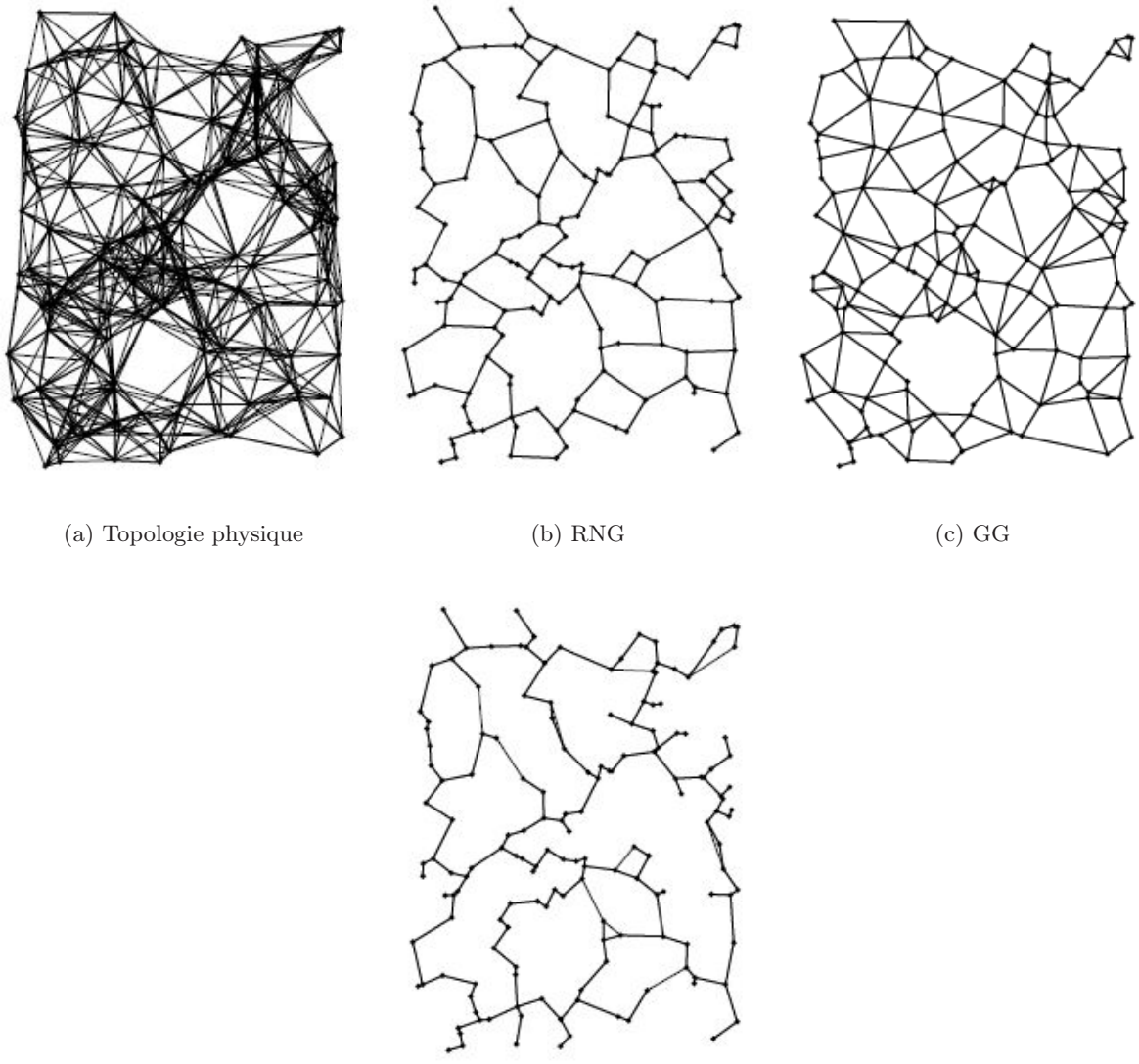
que LMST est un sous-graphe de RNG. La topologie LMST préserve la connexité du graphe initial aussi.

3.3.2 Besoin de protocole de contrôle de topologie pour les réseaux hétérogènes

Un algorithme de contrôle de topologie ne permet pas seulement de maintenir la connectivité du réseau en économisant de l'énergie pour prolonger la durée de vie du réseau, mais aussi d'améliorer la réutilisation spatiale et donc la capacité du réseau [65]. Ainsi, plusieurs algorithmes de contrôle de topologie basés sur l'ajustement des puissances de transmission et s'appuyant sur la mise en hiérarchie ont été proposés afin de créer des topologies logiques ou virtuelles dans les réseaux sans fil multi-sauts.

Cependant, la plupart de ces protocoles de topologie supposent que tous les nœuds sont homogènes avec des capacités uniformes. L'hypothèse des nœuds homogènes n'est pas toujours valable. D'une part, les WSANs sont hétérogènes de nature comme nous venons de voir dans le chapitre précédent. D'autre part, même si les nœuds constituants le réseau sont homogènes, le déploiement et la topologie physique peuvent influencer sur les capacités des nœuds (sur la portée de communication par exemple).

Récemment, nous recensons des travaux qui s'intéressent aux réseaux hétérogènes [91], [105],



(a) Topologie physique (b) RNG (c) GG

(d) LMST

FIGURE 3.10: Aperçu sur la topologie physique (a), le graphe RNG (b), le graphe GG (c) et le graphe LMST (d).

[159], [80]. D'une part nous trouvons les travaux qui s'intéressent à l'impact de l'hétérogénéité sur les protocoles de contrôle de topologie [91]. Les auteurs du [91] montrent que la plupart des algorithmes de contrôle de topologie ne peuvent pas garantir la connexité du réseau dans un contexte hétérogène.

D'autre part, d'autres travaux étudient l'impact de cette hétérogénéité sur la durée de vie du réseau [159] et sur le coût global du réseau [105]. Dans [159], les auteurs évaluent l'impact du nombre et du placement des nœuds hétérogènes sur les performances dans des réseaux de différentes tailles et densités. Il est montré que le déploiement des nœuds sans contraintes énergétique a une influence sur les performances du réseau. Un déploiement optimal améliore le taux de livraison et augmente la durée de vie du réseau. Dans [105], une étude comparative des réseaux homogènes et hétérogènes en termes de coût global du réseau. Ce coût est défini comme étant la somme du coût de l'énergie et le coût du matériel. Les auteurs analysent des réseaux organisés en clusters où les nœuds sont à 1-saut et en multi-sauts des CHs. Ils concluent que l'utilisation d'une organisation en clusters où les capteurs sont à 1-saut des CHs n'est pas efficace. Les auteurs proposent dans [105] une version multi-sauts M-LEACH du protocole LEACH [68].

La plupart des travaux autour des protocoles de contrôle de topologie traitent seulement l'hétérogénéité au niveau des capacités énergétiques des nœuds [80]. Or l'hétérogénéité englobe aussi l'hétérogénéité des puissances de transmission et des capacités de calcul. Les protocoles de contrôle de topologie pour les réseaux hétérogènes ne doivent pas seulement équilibrer la consommation énergétique entre les nœuds riches en ressources et les nœuds pauvres en ressources. Ils doivent aussi s'intéresser aux autres types d'hétérogénéités pour profiter de ces ressources. De notre point de vue, les capacités de transmission et de calcul peuvent offrir des opportunités pour les protocoles de contrôle de topologie. Nous défendons cette idée dans nos contributions présentées dans les chapitres suivants.

3.4 Les challenges du routage dans les WSANs

3.4.1 Le routage dans les WSANs : rôle et défis

Le protocole de routage est central pour le bon fonctionnement de tout système de communication multi-sauts. Le premier objectif d'un protocole de routage est d'établir une route entre une source et une destination, de maintenir une trace sur l'existence de cette route et de faciliter le bon acheminement des données le long de la route retenue.

Nous pouvons distinguer plusieurs défis que rencontre un protocole de routage dans les réseaux sans fil et plus particulièrement dans les WSANs.

D'abord, la tâche du routage est influencée par le fait que le canal radio est partagé. En plus ce canal est fluctuant et volatile au cours du temps : un lien radio peut apparaître et disparaitre.

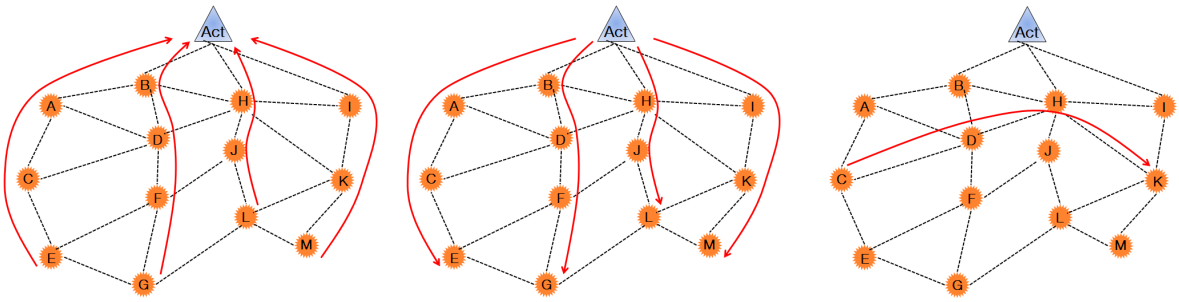
Ensuite, la plupart des nœuds constituant les WSANs (principalement les nœuds capteurs) souffrent de ressources limitées en termes d'énergie, de mémoire, de capacité de calcul et de puissance de transmission. Ainsi un protocole de routage ne doit pas être complexe algorithmiquement. En effet, ces protocoles vont être intégrés dans des microprocesseurs à faible coût et ils doivent être économes en énergie pour étendre la durée de vie du réseau sans compromettre les communications fiables et efficaces entre les nœuds.

D'autres défis auxquels un protocole de routage dans les WSANs sera confronté sont en liaison avec la particularité architecturale de ces réseaux. En effet, contrairement aux réseaux sans fil classiques, les applications visées par les WSNs et les WSANs sont diverses allant des applications de surveillances à petites échelles jusqu'aux applications sensibles au délai par exemple. Ainsi, des

solutions basées sur le maintien de tables de routage centralisées n'est pas envisageable dans ces réseaux.

Enfin, nous trouvons différents types de trafics dans les WSNs et les WSANs. En effet, trois types de trafic peuvent exister comme le montre la figure 3.11 :

- **Trafic Multi-Points à Point ou *convergecast* (MP2P)** : C'est le trafic de base dans les WSNs et les WSANs (Figure 3.11(a)). Ici, les données sont récoltées par des nœuds capteurs et sont acheminées vers le(s) nœud(s) puits ou le(s) nœud(s) actionneur(s).
- **Trafic Point à Multi-Points ou *divergecast* (P2MP)** : Nous trouvons ce type de trafic dans les WSNs aussi. Ici, le(s) nœud(s) puits ou actionneur(s) envoie(nt) un message vers tous les nœuds constituant le réseau. Ce type de trafic est appelé aussi dissémination (Figure 3.11(b)).
- **Trafic Point à Point (P2P)** : C'est le trafic de données échangé entre deux nœuds du réseau (Figure 3.11(c)). Il se peut qu'un nœud capteur envoie des données vers un actionneur ou vice versa ou bien deux actionneurs qui s'envoient des données.



(a) Trafic MP2P (b) Trafic P2MP (c) Trafic P2P

FIGURE 3.11: Différents types de trafic dans les réseaux WSANs

3.4.2 Caractéristiques des métriques de routage

Les protocoles de routage se concentrent traditionnellement sur la recherche de chemins avec le nombre de sauts minimum. Toutefois, ces chemins peuvent inclure des liens lents ou avec perte, conduisant à un débit faible [46] et un manque de fiabilité [47]. Un algorithme de routage permet de sélectionner le « meilleur »chemin (en se basant sur des contraintes et des métriques bien déterminées) et de prendre explicitement en compte la qualité des liens. Dans cette section, nous survolons les principales caractéristiques des métriques de routage.

En effet, une métrique de routage doit prendre en compte la qualité des liens du réseau et permettre le choix du (des) chemin(s) qui satisfai(en)t les contraintes de routage. Les éléments clés pouvant être utilisés pour composer une métrique de routage pour les WSNs ou les WSANs sont : le nombre de sauts, la capacité des liens, la qualité des liens, la diversité du canal, l'efficacité énergétique et la fiabilité. Les caractéristiques souhaitables d'une bonne métrique de routage pour les WSNs ou les WSANs sont [64] :

1. Utilisation des informations locales et non-locales : Certaines métriques nécessitent des informations ou des paramètres observés sur les autres nœuds du réseau (telles que les canaux utilisés par les nœuds du saut précédent dans un chemin, le taux de livraison, etc.). Ces infor-

mations peuvent faire partie de la métrique de routage pour rendre plus optimale les décisions de routage [64].

2. Equilibrage de la charge : Une bonne métrique de routage doit avoir la capacité de pouvoir équilibrer la charge entre les nœuds du réseau. Cette métrique doit fournir une utilisation des ressources disponibles d'une manière équitable entre les nœuds. Il s'agit d'une considération très importante surtout quand il y a une concentration de trafic autour du (des) nœud(s) puits et/ou actionneurs. De plus, pour équilibrer la charge, cette métrique doit prendre en considération l'état des liens et/ou des nœuds voisins. Comme dans [94] par exemple, la métrique du routage utilisée prend en compte la qualité du lien, l'interférence entre les liens et la charge de trafic sur ces liens. Les auteurs de [94] montrent qu'en utilisant une telle métrique, l'équilibrage de charge entre les canaux ainsi qu'entre les nœuds peut être atteint.
3. Agilité : L'agilité d'une métrique se réfère à sa capacité à répondre rapidement et efficacement à l'évolution du réseau. L'évolution du réseau peut couvrir un changement au niveau de la topologie ou une variation de la charge de trafic. Une métrique est dite agile, si la vitesse à laquelle les mesures sont prises est plus élevée que la vitesse des changements dans le réseau. Si la vitesse des changements dépasse la vitesse de la mesure, alors la métrique ne fournit plus une image fidèle de l'état du réseau et n'est donc plus précise. Dans [47] par exemple, les auteurs montrent que la métrique basée sur le nombre de sauts est une métrique agile dans des scénarii de réseau très dynamique (avec une grande mobilité).

3.4.3 Quelques métriques pour le routage

Dans cette section, nous survolons les principales métriques utilisées par les protocoles de routage pour évaluer les qualités des liens constituant un chemin.

Nous trouvons ETX (*Expected Transmission count*) [46] qui a été parmi les premiers travaux sur les métriques. ETX est calculée comme le nombre de transmissions nécessaires pour réaliser l'envoi d'un paquet avec succès. ETT (*Expected transmission time*) [48] est une extension de la métrique ETX. ETT représente le temps prévu pour réussir à transmettre un paquet à la couche MAC. Le surcoût de ces métriques est faible et elles prennent explicitement en compte le taux de pertes. Cependant, ce taux de perte de paquets retransmis n'est pas le même que le taux de perte de paquets de données. Parce que les paquets de contrôle sont généralement plus petits que les paquets de données. En plus, ces métriques ne prennent pas aussi en compte le taux d'occupation ou la surcharge des liens.

RTT (*Per-hop Round Trip Time*) [3] est une métrique qui mesure le délai en aller-retour observé pour les messages envoyés en unicast entre les nœuds voisins. Chaque nœud envoie un paquet portant un horodatage (*timestamp*) à chacun de ses voisins. A la réception, chaque voisin répond avec un accusé de réception. Ainsi le nœud émetteur peut calculer le temps aller-retour avec chacun de ses voisins et il choisit les voisins légitimes pour lesquels la valeur de RTT est inférieure à un seuil. RTT est facile à implémenter, mais cette métrique est coûteuse et non évolutive : elle est coûteuse si le temps aller-retour est estimé périodiquement avec un paquet de contrôle et elle est non évolutive si l'estimation du temps aller-retour est faite seulement une seule fois.

Parmi les propositions qui ne nécessitent pas spécialement l'utilisation des paquets de contrôle, nous trouvons par exemple la mesure basée sur les RSSI [146] (*Received Signal Strength Indicator*). A la réception d'un paquet (de donnée ou de contrôle), le nœud mesure la puissance du paquet reçu : si cette valeur dépasse un seuil, le nœud considère la source comme un voisin légitime. Le meilleur chemin entre deux nœuds est constitué de liens dont la somme des métriques du chemin est inférieure à un seuil donné. L'inconvénient de l'utilisation des métriques basées sur la valeur du RSSI

est que d'abord, les liens utilisés peuvent être asymétriques. Ces liens causent une dégradation des performances des protocoles qui ne prennent pas en compte ce type de liens. En plus, la valeur du RSSI, généralement stable à long terme, elle est très fluctuante et instable à court terme. Finalement, il a été montré dans [71] que l'utilisation de la métrique RSSI pour estimer la proximité n'est pas efficace.

3.5 Les protocoles de routage dans les WSANs

Plusieurs papiers dans la littérature ont étudié les protocoles de routage pour les WSNs et les WSANs. Dans [90], une classification classique des protocoles de routage a été faite : les protocoles à état de liens ou à vecteur de distance. Dans [4] et [9], les auteurs présentent un état de l'art des protocoles de routage dédiés aux WSNs mais il est de nos jours obsolète. Les auteurs de [154] présentent un état de l'art récent sur les protocoles de routage pour les LLNs (couvrant les WSNs et les WSANs). Dans [154], une présentation suivant l'évolution historique des protocoles de routage est faite.

En effet, les protocoles de routage initialement développé pour les réseaux ad-hoc ont été adaptés aux nouveaux besoins des réseaux de capteurs. Cela a conduit à l'apparition des protocoles basés sur les inondations et sur la création des hiérarchies. Ensuite, les protocoles de routage géographique sont apparus. Avec ces protocoles, les nœuds, connaissant leurs positions géographiques, utilisent ces informations pour acheminer les données dans le réseau. Récemment, nous trouvons des protocoles de routage s'appuyant sur des topologies d'auto-organisation et se libérant des informations géographique.

Dans cette section nous suivons cette taxonomie comme nous la représentons sur la figure 3.12.

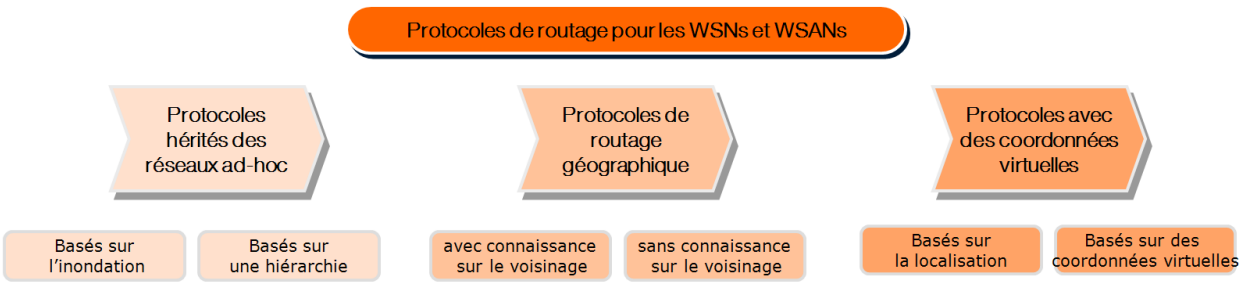


FIGURE 3.12: Classification des protocoles de routage pour les WSNs et les WSANs.

3.5.1 Les protocoles de routage hérités des réseaux Ad-hoc

Les protocoles de routages à plat

Les premiers travaux s'intéressent aux réseaux à plat et portent sur l'inondation. Cette technique primordiale permet d'assurer les trois types de trafics décrits précédemment (section 3.4.1). En effet, ayant une information à envoyer, chaque nœud diffuse ce message à tous ses voisins. Chacun de ces voisins, relaie à son tour ce message jusqu'à ce que tous les nœuds du réseau aient reçu ce message y compris la destination ou jusqu'à ce que le nombre de saut maximum autorisé soit atteint. Cette technique qui ne nécessite pas un algorithme de routage souffre principalement d'une consommation énergétique non négligeable [148].

Ensuite, plusieurs travaux ont porté sur la réduction des coûts de l'inondation. Des travaux proposent la limitation du nombre de relais avec des approches probabilistes [164] [66], ou la sélection d'un ensemble de nœuds minimal suffisant pour livrer le message (par exemple la sélection de relais multipoints dans [37]) ou en sélectionnant un ensemble de nœuds connectés dominant ou en utilisant des techniques comme le NES (*Neighbor elimination scheme*) pour éliminer les messages redondants [116] [142].

D'autres travaux essayent de limiter la surface géographique concernée par l'inondation, en utilisant des informations de localisations des nœuds ou sur une structuration hiérarchique dans laquelle un sous-ensemble de nœuds utilise les inondations seulement au sein du *cluster* [68].

Nous trouvons aussi des travaux qui proposent de réduire le nombre de relais en se basant sur l'énergie résiduelle au niveau des nœuds [135].

Une autre approche consiste à inonder les paquets de contrôle avant la transmission de données : les paquets de contrôle sont diffusés pour trouver un chemin vers la destination. Une fois cette route est trouvée les données suivent ce chemin. Les protocoles les plus populaires dans le monde ad-hoc mais qui ne sont pas adaptés auw WSNs ou les WSANs sont DSR (*Dynamic Source Routing*) [77], AODV (*Ad-hoc On-demand Distance Vector*) [118] et OLSR (*Optimized Link State Routing Protocol*) [37].

Les protocoles de routage hiérarchiques

Les protocoles de routage hiérarchiques sont généralement des protocoles basés sur les *clusters*. Les nœuds sont regroupés en *clusters*, avec un nœud chef de *cluster* (CH *clusterhead*) élu pour chacun d'eux. La transmission de données va généralement des membres du *cluster* au nœud CH, avant d'aller d'un CH à un autre jusqu'à atteindre la destination finale. Vu que les CHs effectuent des tâches nécessitant plus de traitement et de capacité de transmission, ils sont généralement des nœuds ayant une capacité énergétique plus élevée [168], [163] ou alors un algorithme d'ordonnancement d'activité pour changer les CHs est utilisé [68]. Dans [1], les auteurs présentent une taxonomie et une classification des protocoles de création de *clusters*.

LEACH (*Low-Energy Adaptive Clustering Hierarchy*) [68] est l'une des premières approches dans la littérature traitant le routage hiérarchique pour les WSNs. Selon une probabilité prédéfinie, les nœuds s'élisent en tant que CHs ; les autres nœuds rejoignent le CH le plus proche. Chaque CH crée et diffuse une décomposition en slots de temps pour coordonner la communication *intra-cluster* (dans le même *cluster*). Pour les communications *inter-cluster* (entre les différents *clusters*), LEACH utilise l'accès basé sur CDMA. LEACH exploite la rotation aléatoire du rôle des CHs pour répartir équitablement la charge de l'énergie. L'inconvénient de LEACH est qu'il n'assure pas la connectivité entre tous les nœuds du réseau.

Avec HEED (*Hybrid Energy-efficient Distributed clustering protocol*) [162], la sélection des CHs se base sur une combinaison entre l'énergie résiduelle ainsi que le degré des nœuds. Comme LEACH, la formation des *clusters* est complètement distribuée. L'inconvénient de HEED est qu'il suppose que tous les nœuds peuvent ajuster leurs portées de communication grâce à la puissance de transmission. Ainsi les communications *intra-cluster* sont assurées avec une faible puissance de transmission et les communications *inter-cluster* sont faites avec des puissances plus élevées. Dans le cas d'un déploiement réel, où il y a de l'interférence, de l'affaiblissement et du multi-chemins, il est difficile de prédire la portée de communication à partir de la puissance de transmission. L'utilisation des messages d'acquittement peut faire une correspondance entre la puissance du signal et la portée de communication des nœuds, mais l'utilisation de ces messages de contrôle peut s'avérer coûteuse en termes de consommation énergétique.

TEEN (*Threshold sensitive Energy Efficient sensor Network protocol*) [99] et APTEEN (*Adaptive threshold sensitive Energy Efficient sensor Network protocol*) [100] sont conçus pour des applications sensibles aux délais. Contrairement à LEACH, TEEN et APTEEN construisent une hiérarchie multi-niveaux (voir figure 3.13). TEEN [99] est conçu pour les applications se basant sur le mode de communication événementiel (les applications dédiées pour détecter les changements soudains). L'architecture du réseau de capteurs est basée sur un groupement hiérarchique où les nœuds les plus proches forment des (*clusters*). Ensuite, les CHs du premier niveau forment un *cluster* de deuxième niveau et ainsi de suite jusqu'à ce que le nœud puits soit atteint (voir figure 3.13). APTEEN [100] est une extension de TEEN dédié à la fois aux applications se basant sur le mode de communication événementiel et à celles se basant sur le mode de communication périodique.

Les principaux inconvénients de TEEN et APTEEN est le surcoût et la complexité associés à la formation des *clusters* à plusieurs niveaux.

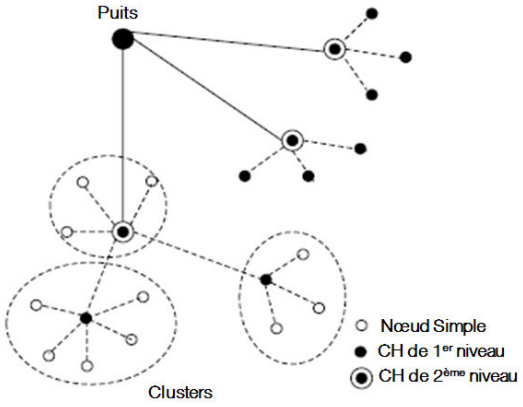


FIGURE 3.13: Construction d'une hiérarchie multi-niveaux avec TEEN et APTEEN

Les protocoles à base de *clusters* se diffèrent selon plusieurs critères comme la manière de construction des *clusters*, les contraintes utilisés pour l'élection des CHs, les rôles et les propriétés des CHs, la manière dont ils communiquent avec les membres du *cluster*, avec les autres CHs et avec le puits. Cette différence porte aussi sur la définition même d'un *cluster* : le diamètre, la cardinalité, avec ou sans recouvrement, etc.

L'utilisation de *cluster* a l'avantage de limiter la zone d'inondation, d'augmenter la durée de vie du réseau en étant efficace en énergie [100], [23]. Plusieurs autres applications peuvent profiter de cette topologie. En effet, les CHs peuvent assurer la tâche d'agrégation de données par exemple [59]. Une même donnée reportée par plusieurs capteurs, qui sont dans une même zone géographique, peut être agrégée au niveau du CH. Ainsi on réduit le nombre de données à destination du puits et on conserve de l'énergie globale au niveau du réseau.

3.5.2 Les protocoles de routage à base de coordonnées géographique

Beaucoup d'applications des WSNs et des WSANs exigent que tous les nœuds connaissent leurs localisations physiques. Cela peut être réalisé grâce au GPS par exemple. Cette information de localisation est utilisée pour assurer une communication fiable dans le réseau. En effet, un protocole de routage géographique utilise la connaissance de la position géographique du nœud et/ou les positions de ses voisins, pour élire le prochain saut et acheminer les données vers le nœud puits.

La forme la plus simple du routage géographique est le routage *Greedy* [54]. Le principe du routage *Greedy* est que chaque nœud envoie les données vers le voisin le plus proche géographiquement de la destination finale. Prenons l'exemple de la figure 3.14 et supposons que le nœud S est le nœud courant dans la route et D est le nœud destination. Les voisins du nœud S sont ceux présents dans le cercle centré à S. Le nœud B est le voisin du nœud S le plus proche géographiquement du nœud destination D. Ainsi avec *Greedy*, le nœud B sera choisit comme prochain saut vers la destination. Le routage *Greedy* est simple et localisé, il est adéquat aux réseaux larges échelles et denses.

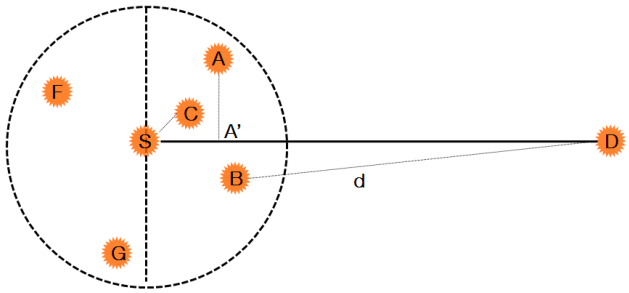


FIGURE 3.14: Différentes façons de définir la distance vers une destination.

Des améliorations portent sur la métrique de choix du prochain saut. Il est à noter que considérer seulement l'avancement géographique vers la destination n'est pas une bonne stratégie [134] [128]. D'une part, transmettre le message à un voisin le plus proche de la destination implique que ce voisins est susceptible d'être à une grande distance à partir du nœud de transfert, et donc le lien entre le nœud actuel et ce voisin n'est pas stable [134]. D'autre part, pour acheminer des données vers un nœud puits fixe dans le réseau, le choix du prochain saut sera statique. Cela peut causer un épuisement de batterie des nœuds capteurs. Des travaux portent sur l'amélioration du choix du prochain saut en combinant l'avancement géographique avec le coût énergétique au niveau de chaque nœud par exemple [128].

L'inconvénient majeur du routage *Greedy* est qu'il n'assure la livraison des données que dans les réseaux denses. En effet, quand un nœud ne trouve pas de voisins plus proche vers la destination, le routage *Greedy* échoue [36]. Dans la figure 3.15, considérons le nœud S voulant envoyer un message vers la destination D. Le message sera envoyé à partir du nœud S vers A, son voisin le plus proche géographiquement de la destination D. Une fois arrivé au nœud A, le message ne peut plus avancer puisque A n'a pas de voisin plus proche que lui-même de la destination D. Le nœud A se retrouve dans une zone de vide appelée aussi un minimum local. Cette zone de vide apparaît quand un nœud n'a pas de voisin plus proche que lui-même de la destination.

Pour remédier à ce problème, plusieurs protocoles de routage géographique ont été proposés avec une garantie de livraison avec des liens et des nœuds fiables. L'idée est de combiner entre deux modes : le routage *Greedy* et le routage *Face*. Le routage *Greedy* est utilisé par défaut, si ce mode échoue à cause d'un zone de vide dans le réseau, le second mode est utilisé pour contourner cette zone. Une fois contournée, le mode de routage *Greedy* est repris (voir figure 3.16).

Le protocole GFG (*Greedy-Face-Greedy*) [22] introduit ce principe. En mode *Face*, GFG utilise des graphes planaires pour contourner la zone de vide. Prenons l'exemple de la figure 3.16 où le nœud S envoie des données vers la destination D. Le message de donnée arrive dans une zone de vide (où le nœud A n'a pas un voisin plus proche que lui-même de la destination D). Dans ce cas, GFG passe du mode *Greedy* au mode *Face*.Le mode *Face* est utilisé pour contourner la zone de vide. Quand le nœud actuel (le nœud B) est plus proche de la destination que le nœud qui a initié le mode

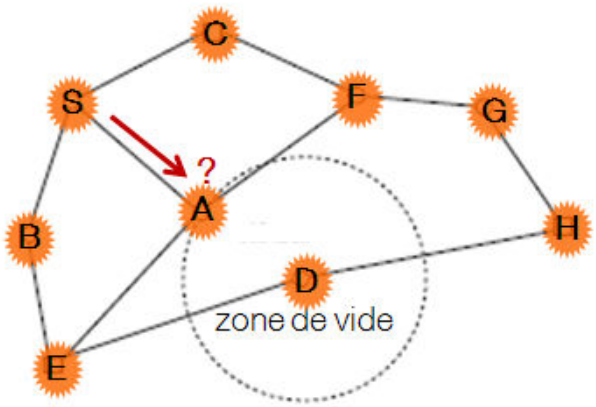


FIGURE 3.15: Problème de routage géographique en présence de zone de vide.

Face (le nœud A), GFG revient au mode *Greedy*. Dans le mode *Face*, un nœud ne considère que les liens entre lui et ses voisins appartenant au graphe planaire de Gabriel [56]. Parmi ces voisins, il choisit le prochain saut en utilisant la règle de la main droite.

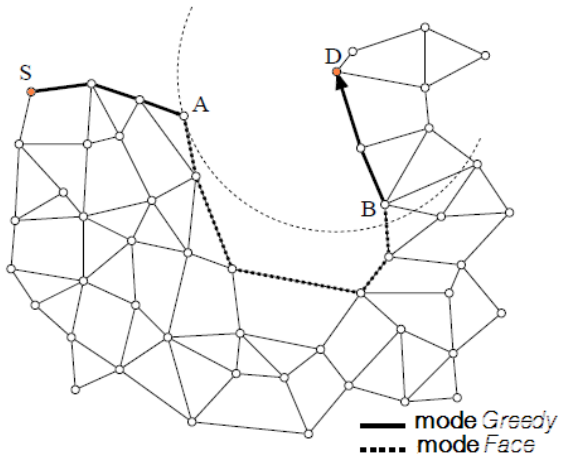


FIGURE 3.16: GFG : Combinaison du mode *Greedy* et *Face* pour contourner les zones de vide.

GFG a été revisité par Karp et Kung et a été appelé GPSR (*Greedy Perimeter Stateless Routing*) [79].

Les deux algorithmes *Greedy* et *Face* ont été optimisés afin de réduire la consommation d'énergie. Plusieurs autres travaux, comme dans [52], optimisent la métrique du choix du prochain saut dans les deux modes *Greedy* et *Face*. L'idée principale est qu'un nœud choisit le prochain saut qui minimise les coûts sur le progrès. Le coût est défini comme l'énergie dépensée quand ce voisin est choisi comme prochain saut. Le progrès qu'offre un voisin est défini comme la réduction de la distance euclidienne vers la destination si ce voisin est choisi comme prochain saut.

Limites des protocoles de routage géographique

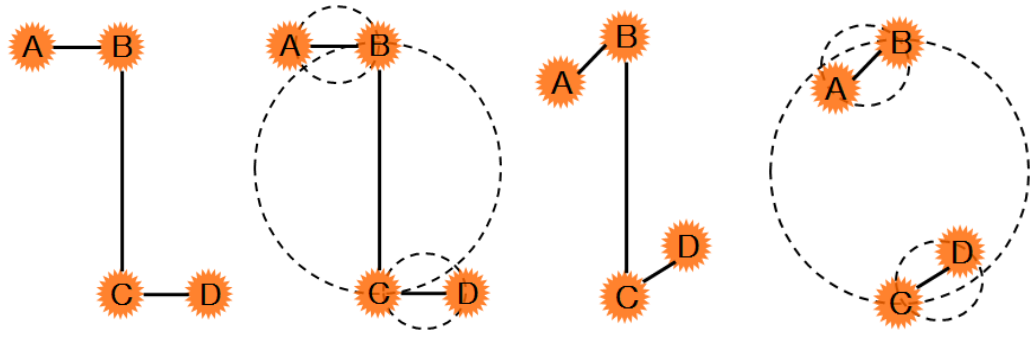
Pour éviter les zones de vide, les protocoles de routage géographique utilisent plusieurs techniques [36]. La technique la plus connue est la technique de planarisation. La planarisation d'un graphe est le fait de supprimer les liens redondants comme dans [147] et [56].

Les solutions basées sur des techniques de planarisation de graphes reposent sur deux hypothèses : (1) les nœuds connaissent parfaitement leurs positions géographique et (2) le graphe de connectivité est un graphe à disque unitaire UDG. En conséquence, lorsque ces hypothèses ne sont plus satisfaites (dans un déploiement réel par exemple), les performances de ces protocoles de routage se dégradent fortement [154]. Quand les nœuds ne connaissent pas parfaitement leurs positions, comme illustré sur la figure 3.17, la technique de planarisation peut provoquer la déconnexion du réseau. En effet, dans [152], les auteurs font une évaluation quantitative de ces phénomènes et étudient leurs influences sur les performances des protocoles de routage.

Dans [152], les auteurs proposent un protocole de routage géographique, appelé *3rule*, utilisant la séquence de nœuds déjà parcouru pour aider à la décision de transmission saut par saut. Chaque nœud traversé par un message ajoute à l'entête de ce message son identifiant supposé unique. Ainsi, à la réception d'un message un nœud peut vérifier s'il a déjà participé à la transmission de ce même message. Le nœud courant applique 3 règles simples pour filtrer la liste de ses voisins et transmettre ce message à celui qui convient. En favorisant les voisins qui sont géographiquement plus proche de la destination, les auteurs montrent que le protocole de routage *3rule* trouve des chemins qui ont la même longueur que ceux trouvés par GFG, tout en garantissant la livraison sur des graphes stables.

Les protocoles de routage géographique supposent que chaque nœud connaît ses coordonnées géographiques, ses voisins et leurs emplacements. Traditionnellement, la découverte du voisinage se fait en échangeant périodiquement des messages « *Hello* »pour maintenir des tables de voisinage d'une manière proactive.

Cette découverte du voisinage peut s'avérer couteuse pour les réseaux à larges échelles et à forte densité. C'est ainsi qu'une famille de protocole de routage géographique sans connaissance de voisinage est apparue.



(a) Une connaissance exacte de la position géographique conserve la connectivité du réseau après planarisation

(b) La technique de planarisation peut provoquer la déconnexion du réseau dans le cas où les nœuds ont une connaissance approximative des positions géographique

FIGURE 3.17: Nécessité d'une connaissance géographique parfaite pour la technique de planarisation

Routage géographique sans connaissance de voisinage

D'autres approches proposent d'utiliser le principe du routage géographique sans recours à la connaissance de voisinage. Ce sont des protocoles de routage sans connaissance préalable de voisinage [12] [21] [34] [35] [69] [55] [88] [131] [152] [158]. En mode *Greedy*, le nœud courant, ayant un message vers une destination, au lieu de l'envoyer vers le voisin le plus proche de la destination, il diffuse ce message dans son voisinage radio. Le prochain saut est sélectionné d'une manière distribuée à partir d'un ensemble de nœuds candidats se trouvant dans une zone appelée une zone de progrès. Chaque nœud candidat calcule un délai avant de participer à l'acheminement du message de donnée reçu. Ce délai est utilisé pour privilégier les nœuds les plus prometteurs (offrant un meilleur avancement vers le nœud destination) : Plus le progrès est important, plus le délai est court.

IGF (*Implicit Geographic Forwarding*) [21], SIF (*State-Free Implicit Forwarding*) [34], CBF (*Contention-based Forwarding*) [55] et OGF (*On-demand Geographic Forwarding*) [35] proposent une intégration Routage/MAC. Les protocoles BLR (*Beacon Less Routing*) [69], BOSS (*Beacon-less On Demand Strategy for Geographic Routing*) [131], PSGR (*Priority-based Stateless Geo-Routing*) [158], PF (*Pizza-Forwarding*) [12] et TRIF (*Tier-based RoutIng Framework*) [88] sont au niveau routage.

Certains de ces protocoles de bases ne fonctionnent que dans le réseau très dense, car ou bien ils n'utilisent pas une stratégie de contournement de la zone de vide (IGF, CBF) ou bien ils proposent des techniques de contournement de trous pas toujours applicables. Par exemples, SIF suppose que les nœuds peuvent augmenter leurs portées de transmission jusqu'à contourner les zones de vide. De plus la plupart de ces protocoles supposent un modèle de propagation basé sur des graphes UDG, ce qui n'est pas toujours vrai pour un déploiement réel.

Nous allons nous intéresser aux protocoles PF et TRIF parce qu'ils supposent des modèles de propagation plus réalistes.

PF (*Pizza-Forwarding*) [12] est un protocole de routage géographique sans connaissance de voisinage. PF se compose de deux stratégies de transmission : la transmission *Greedy* basée sur la technique de contention et la stratégie de récupération optimisée pour contourner les zones de vide. Au lieu de retransmettre le paquet immédiatement, chaque nœud recevant un message calcule un délai d'attente et arme un temporisateur. Le délai d'attente dépendra de la position du nœud dans les quatre secteurs 1, 2, 3 et 4 (voir figure 3.18). Les nœuds dans les secteurs 1 et 2 sont plus prioritaires, donc le délai d'attente est plus court. Lorsque ce temporisateur expire, le nœud gagnant retransmet le message. Pour éviter les retransmissions multiples d'un même message, les nœuds qui sont en phase de contention pour ce message, en écoutant le message retransmis par un autre nœud, vont supprimer ce message et vont arrêter la phase de contention. Les nœuds, qui n'entendent pas ce message retransmis, vont être informés par la source qui envoie un message de notification de retransmission quand elle entend son message relayé. Dans le cas où il n'y a pas de nœud relayeur dans les secteurs 1, 2, 3 et 4, les nœuds des secteurs 3' et 4' vont lancer une découverte de voisinage à 2-sauts. L'objectif est de découvrir un nœud permettant d'avancer vers la destination finale. Après cette phase de découverte, les nœuds des secteurs 3' et 4' vont envoyer le meilleur nœud permettant d'avancer vers la destination finale au nœud source. Ce dernier envoie ainsi son message vers le nœud intermédiaire, annonçant le meilleur voisin à deux sauts. Ce processus continu jusqu'à atteindre la destination finale.

TRIF (*Tier-based RoutIng Framework*) [88] est un protocole de routage sans connaissance de voisinage utilisé conjointement avec le mécanisme d'envoi de RREQ/RREP. Il repose sur l'idée qu'un nœud source (ou un nœud relais qui transmet des données) envoie ses RREQ avec différents niveaux de puissance de transmission. TRIF suppose que la puissance de transmission est réglable, et qu'il y a une relation entre cette puissance de transmission et la portée radio. Un nœud cherchant une

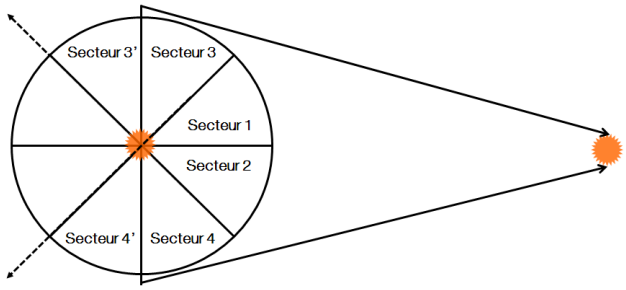


FIGURE 3.18: Principe de décomposition en secteur pour le protocole PF.

route vers une destination envoie successivement des RREQ avec des puissances de transmission décrémentées à chaque fois. Le nœud source ajoute dans l'en-tête de la RREQ le niveau de la puissance de transmission utilisé lors de l'envoi de cette requête. Un nœud récepteur traite les RREQ si le niveau indiqué dans l'entête du message est inférieur ou égal à son niveau de puissance de transmission. Si le niveau de puissance de transmission utilisé pour envoyer la RREQ est plus élevé que le niveau de puissance de transmission disponible au niveau du nœud en réception, alors cette requête est ignorée : le nœud en réception conclut qu'il a reçu peut être cette demande via un lien asymétrique (il peut écouter les messages provenant de cette source, mais sa réponse peut ne pas atteindre ce nœud). Un nœud de réception relai la RREQ en utilisant son niveau de puissance de transmission en ajoutant dans l'entête les informations lui concernant et en se basant sur le même principe (plusieurs RREQ avec des puissances de transmission décrémentées à chaque envoi). Ce processus continue jusqu'à atteindre la destination.

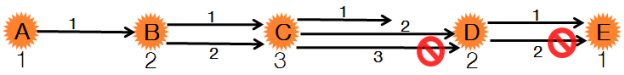


FIGURE 3.19: Principe du protocole de routage TRIF.

Limites des protocoles de routage sans connaissance de voisinage

Dans [130], d'une part, les auteurs montrent par simulation et par expérimentation que le problème des protocoles de routage géographique sans connaissance de voisinage est le taux élevé de paquets dupliqués reçus par la destination finale. D'autre part, les auteurs montrent aussi que ces protocoles sans connaissance de voisinage sont plus performants (au niveau du taux de livraison et au niveau du surcoût protocolaire) pour un déploiement réel que les protocoles de routage basés exclusivement sur le nombre de sauts.

Les protocoles de routage géographique sans connaissance de voisinage sont des bons candidats pour les réseaux à forte dynamique. Cependant, il y a encore quelques questions ouvertes. Il s'agit notamment d'étudier le compromis entre l'efficacité énergétique d'utilisation d'une connaissance préalable du voisinage quand le réseau est stable. De plus, ces protocoles de routage doivent prendre en considération l'état de fonctionnement cyclique des nœuds (mode actif, mode sommeil).

3.5.3 Les protocoles de routage à base de coordonnées virtuelles

Avoir une connaissance de localisation géographique a un coût non-négligeable. De plus, cette localisation basée sur les coordonnées GPS n'est pas toujours évidente à mettre en place. Deux

principales familles de protocole de routage proposant de ne pas utiliser ces coordonnées géographiques.

- La première famille de protocole consiste à utiliser des techniques pour faire une approximation des coordonnées géographiques réelles.
- La deuxième famille propose de se libérer de la topologie physique réelle et opte pour l'utilisation des coordonnées virtuelles et d'utiliser les techniques du routage géographique en se basant sur ces coordonnées [126].

Approximation des coordonnées géographiques réelles

Cette famille de protocoles de routage suppose qu'il y a un ensemble de nœuds dans le réseau qui connaissent leurs positions géographiques réelles. Généralement, ces nœuds sont les nœuds puits et/ou quelques nœuds actionneurs. Ces nœuds disposant de leurs localisations géographique se comportent comme des nœuds ancres [44] [115].

Les nœuds capteurs et/ou actionneurs, ne disposant pas de leurs coordonnées géographiques réelles utilisent des mesures locales et des techniques de localisation pour déduire leurs emplacements. L'objectif d'un nœud est de faire une approximation aussi proche que possible des coordonnées géographiques réelles. Après cette approximation, le protocole de routage utilise ces coordonnées calculées pour acheminer les données entre les nœuds source et destination.

En utilisant des mesures locales et des techniques de localisation, chaque nœud calcule la distance qui le sépare de chaque nœud ancre (comme dans le système GPS représenté par la figure 3.20(a)). En se basant sur ces approximations de distances le séparant de chacun des nœuds ancres, chaque nœud détermine sa position comme l'intersection des cercles centrés à chaque nœud ancre et de rayon la distance à ce dernier.

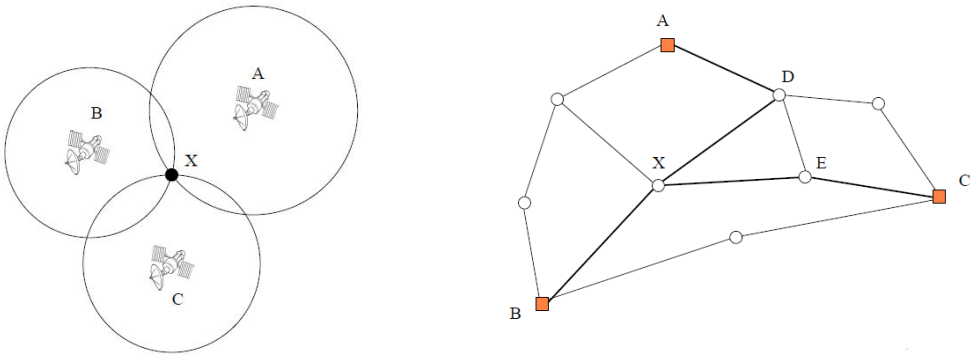
Comme les WSNs et les WSANs sont multi-sauts, une première approximation de la distance à un nœud ancre est la somme des distances des liens individuels constituant le chemin le plus court (voir la figure 3.20(b)).

En mesurant la distance le séparant des nœuds A, B et C (connaissant leur emplacement), le nœud X peut faire une approximation de sa localisation. Cette technique est appliquée pour les systèmes GPS et les réseaux multi-sauts.

Il y a un certain nombre de techniques de mesure de ces distances à un saut [114]. Nous trouvons la puissance du signal reçu RSS (*Received Signal Stregth*) et le temps d'arrivée TOA (*Time Of Arrival*) [114]. D'autres techniques existent aussi comme la mesure de l'angle d'arrivée AOA (*Angle of Arrival*) [111] ou le temps de vol TOF (*Time Of Flight*) [42] mais qui exigent des composants matériels en plus sur les nœuds du réseau.

Avec ces algorithmes de localisation, la précision de localisation dépend du nombre d'ancres, leurs positions relatives [29], de la précision des mesures utilisées et de la densité du réseau [18].

L'inconvénient de l'utilisation des coordonnées géographiques réelles ou des coordonnées de localisation approximatives pour le routage est que la proximité géographique n'est pas synonyme de proximité électromagnétique. En effet, dans le monde réel de déploiement de réseau les nœuds proches géographiquement ne peuvent pas toujours communiquer, et les nœuds qui peuvent communiquer ne sont pas toujours géographiquement proches [83].



(a) Le nœud X mesure la distance qui le sépare des trois satellites.

(b) Le nœud X mesure la distance qui le sépare des trois ancres en multi-sauts en sommant les approximations saut par saut vers les ancres.

FIGURE 3.20: En mesurant la distance le séparant des nœuds A, B et C (connaissant leur emplacement), le nœud X peut faire une approximation de sa localisation. Cette technique est appliquée pour les systèmes GPS (a) et les réseaux multi-sauts (b).

Calcul des coordonnées virtuelles

Les coordonnées virtuelles d'un nœud V sont définis comme un vecteur (V1, V2,. . . , VN) où Vi est la distance logique du nœud courant pour atteindre le nœud ancre i (N est le nombre de nœuds ancre). Une façon simple d'attribuer ces coordonnées est que chaque nœud ancre diffuse périodiquement un message contenant un compteur. Ce message sera relayé par les nœuds non-ancre et le compteur sera incrémenté à chaque saut quand il se propage à travers le réseau. Ainsi, les coordonnées virtuelles ne sont pas liées aux coordonnées réelles voir figure 3.21. Ces coordonnées virtuelles vont être utilisées par l'algorithme *Greedy* du routage géographique. En effet, pendant le processus de routage d'un message vers une destination, le nœud courant choisit le prochain saut qui a une distance virtuelle la plus petite vers le nœud destination [123]. Avec ce principe de coordonnées virtuelles, les auteurs du [123] montrent qu'il y'aura moins de trous dans le réseau et par la suite il y'aura une amélioration de performance en terme de taux de livraison et de conservation d'énergie. L'inconvénient de cette technique est qu'elle suppose que tous les liens sont symétriques. Cette hypothèse n'est pas toujours valable dans les réseaux hétérogènes WSANs.

BVR (*Beacon Vector Routing*) [53], VCap ((*Virtual Coordinate Assignement Protocol*) [29] et VCost [51] sont trois exemples de protocoles de routage basés sur les coordonnées virtuelles. Avec BVR, les nœuds ancres sont choisis aléatoirement dans le réseau. VCap est conçu pour élire un nombre prédéfini de nœuds ancre, répartis uniformément autour de la périphérie du réseau. Les deux protocoles utilisent le routage *Greedy* pour acheminer les données vers les nœuds ancres en se basant sur les vecteurs construits lors de la phase de l'affectation de coordonnées (voir figure 3.21). VCost est une variante des deux protocoles BVR et VCap. Avec VCost, un nœud choisit comme prochain saut son voisin qui minimise le rapport entre le coût énergétique et le progrès réalisé (la diminution de la distance virtuelle).

RTP (*Routing with Position Trees*) [32] attribue un *label* hiérarchique à chaque nœud, à partir d'un nœud racine (généralement un puits ou un actionneur). Le nœud racine donne des étiquettes

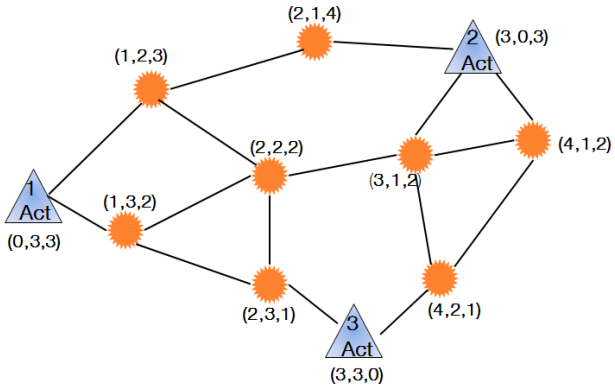


FIGURE 3.21: Principe de routage à base de coordonnées virtuelles

1, 2, 3, etc à ses voisins qui a leurs tours attribuent à leurs voisins n'ayant pas encore des *labels*. A la fin de cette phase d'étiquetage, chaque nœud aura un *label* formé par des chiffres consécutifs : la longueur de l'étiquette représente la distance de ce nœud vers le nœud racine. Le routage est alors effectué de façon hiérarchique. Les messages en provenance des nœuds du réseau vers la racine suivent la hiérarchie logique. Les messages envoyés entre deux nœuds de la topologie passent par leur plus proche ancêtre commun dans l'arborescence (voir figure 3.22). Sur la figure 3.22, le nœud « 3.1.1 »a des données à envoyer vers le nœud « 3.2.1 ». Le message suivra la hiérarchie jusqu'à atteindre le plus proche ancêtre commun (le nœud « 3 »). Une fois arrivé au nœud « 3 », le message descend jusqu'à atteindre la destination finale.

HECTOR (*Energy effiCient Tree-based Optimized Routing protocol*) [106] combine RTP et Vcost pour profiter de la garantit de livraison de RTP et de la conservation d'énergie de Vcost.

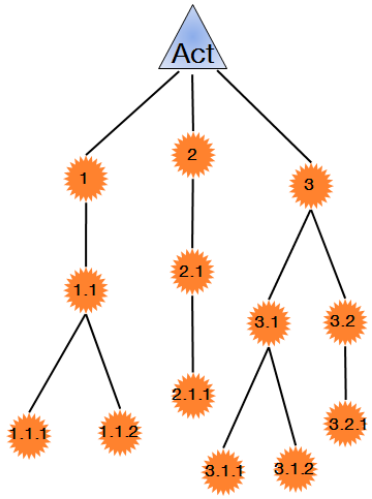


FIGURE 3.22: Principe du protocole RTP.

Dans [71], les auteurs proposent un protocole de routage basé sur des coordonnées virtuelles ne nécessitant pas des nœuds ancres dans le réseau. Ce protocole est basé sur un algorithme de localisation appelé QLoP [70]. L'objectif de QLoP est de permettre de localiser les voisins d'un nœud. Il classe ces voisins en trois classes de voisinage logique : 1-voisinage logique, 2-voisinage logique et 3-voisinage logique. QLoP permet à chaque nœud de déterminer approximativement la

proximité de ses voisins en utilisant une connaissance du 1-voisinage ouvert. Cette connaissance du 1-voisinage ouvert est réalisée avec des échanges périodiques des messages « Hello ». Chaque nœud diffuse dans son message « Hello »la liste de ses voisins. QLoP ne nécessite pas d'informations de localisation géographique ni de synchronisation entre les nœuds.

Les auteurs du [71] appliquent cet algorithme de localisation pour construire une topologie logique QLoP-RNG. Le graphe RNG est construit en utilisant l'indice de proximité calculé avec QLoP. Ensuite, cette topologie logique est utilisée par un protocole de routage pseudo-DSR adapté aux réseaux avec des liens radios dynamiques. Ce protocole favorise les chemins utilisant les nœuds du 1-voisinage logique comme relais. La topologie QLoP-RNG réduit le coût de la diffusion des requêtes utilisées par le protocole de routage DSR et permet aussi de construire des chemins plus robustes aux bruits et aux interférences.

L'utilisation des coordonnées virtuelles pour le routage dans les réseaux WSNs et WSANs est une approche très prometteuse. Puisque ces coordonnées virtuelles sont liées à la topologie du réseau et non pas à l'emplacement physique des nœuds.

Cependant les protocoles de routage basés sur ces coordonnées virtuelles souffrent souvent d'un allongement de chemin par rapport au chemin optimal. En plus, un service assurant la connaissance de ces coordonnées virtuelles est nécessaire pour permettre aux nœuds de communiquer.

Protocoles de routage par gradient

Le routage par gradient est simple et facile à mettre en oeuvre pour les déploiements dans le monde réel, il est surtout utile pour le trafic MP2P.

La notion de gradient est particulièrement utile pour les trafics MP2P que nous trouvons aussi bien dans les WSNs que dans les WSANs. Un scénario simple de trafic MP2P quand tout le trafic est envoyé vers un seul puits et/ou un actionneur. Dans ce cas, un gradient unique - dont la racine est le nœud puits et/ou actionneur- est construit et maintenu dans le réseau. Le gradient peut être simplement le nombre de saut vers ce puits et/ou cet actionneur. Il est appelé hauteur, rang ou simplement gradient. Un nœud courant ayant un message vers le puits et/ou l'actionneur, envoie vers un voisin ayant un rang plus petit que le sien.

GBR (*Gradient-Based Routing*) [133] est l'exemple typique du routage par gradient. Le rang d'un nœud reflète le nombre de sauts le séparant du puits ou de l'actionneur. Un nœud peut augmenter son rang lorsque son énergie descend en dessous d'un certain seuil pour qu'il ne participe pas trop à la phase de récolte de données.

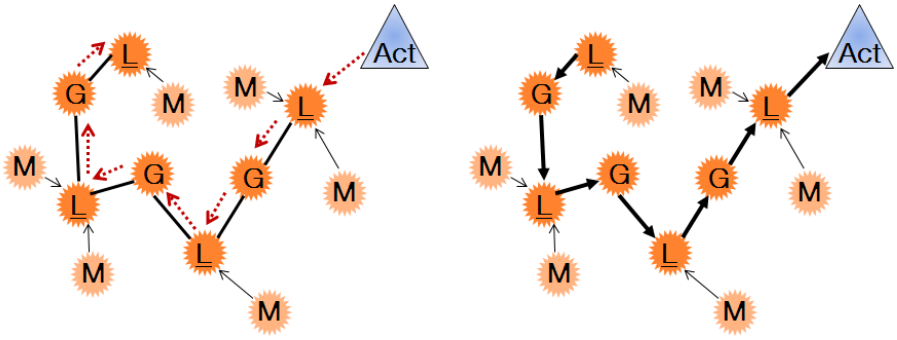
GRAB (*GRAdient Broadcast*) [160] renforce la fiabilité de la livraison de données à travers une diversité de chemins. Pour assurer la collecte de données, le puits ou l'actionneur construit d'abord un gradient en diffusant des messages d'avertissement (ADV) dans le réseau. Le rang d'un nœud (surnommé « coût »dans GRAB) est la quantité d'énergie minimale nécessaire pour transmettre un message à partir de ce nœud vers le nœud puits ou le nœud actionneur. GRAB suppose que chaque nœud a les moyens d'estimer le coût d'envoi de données vers les nœuds dans son voisinage. Ainsi chaque nœud maintient le coût nécessaire pour la transmission des paquets vers le puits ou l'actionneur. Dans la phase de routage, seulement les nœuds offrant un coût plus faibles sont candidats pour relayer le message vers la destination finale.

Le protocole BBDD (*Backbone Based Data Dissemination*) [96] vise à la fois à réduire l'énergie nécessaire pour le trafic P2MP (diffusion de l'information du nœud puits ou actionneur à tous les nœuds du réseau) et à faciliter le trafic MP2P (la collecte de données provenant de capteurs vers le puits ou l'actionneur). BBDD utilise une approche en deux étapes : Une première étape

de construction de topologie logique ; Une deuxième étape, permettant d'orienter la topologie pour faciliter le trafic de collecte de données.

Dans la première étape, BBDD construit une topologie basée sur Legos [97], qui est un protocole d'auto-organisation visant à générer une topologie logique en épine dorsale non-orienté (voir section 3.3.1).

Pendant la deuxième étape, le puits ou l'actionneur envoie une requête qui va parcourir la topologie en épine dorsale construite pendant la première étape. Cette demande se propage à travers la topologie logique (figure 3.23(a)) et oriente l'épine dorsale. Quand un nœud a des données a envoyé vers la racine, il les envoie sur la topologie logique orientée. Ces données vont parcourir l'épine dorsale orientée jusqu'à atteindre la destination finale (figure 3.23(b)).



(a) Propagation de la requête à travers la topologie logique

(b) Orientation de la topologie logique

FIGURE 3.23: Principe du protocole BBDD.

Le protocole RBF (*RSSI based Forwarding Protocol*) [14] qui est un protocole combinant les fonctions de routage avec les fonctions MAC qui fusionne les opérations d'accès au canal et de routage. RBF utilise la valeur de RSSI comme gradient qui servira pour le routage, il peut être divisé en deux phases : la première phase consiste à la diffusion d'un message par le nœud puits ou l'actionneur. RBF suppose que ce puits peut atteindre tous les nœuds du réseau par un seul message de diffusion. En recevant ce message, les nœuds capteurs enregistrent la valeur de la puissance du signal reçu (RSSI). La deuxième phase consiste à la collecte saut par saut des messages de données à partir des nœuds capteurs vers la destination finale. RBF utilise le mécanisme RTS-CTS-DATA-ACK, qu'il exploite aussi pour le routage. Quand un nœud a des données à transmettre (par exemple le nœud 1 dans la figure 3.24), il diffuse une trame RTS dans son voisinage. Dans la trame RTS, le capteur envoie la valeur du RSSI stockée au cours de la première phase (à savoir le niveau de puissance du signal reçu du message diffusé par le nœud puits ou le nœud actionneur). Les nœuds, recevant cette trame RTS (les nœuds 2, 3, 4 et 41 dans la figure 3.24), lisent la valeur de RSSI contenue dans cette trame RTS et la comparent à leurs propres valeurs de RSSI en réception du puits. Si leurs valeurs du RSSI stockées sont plus élevés que la valeur du RSSI contenue dans la trame RTS, alors ils déduisent qu'ils sont plus près de la destination finale (les nœuds 2, 3 et 4 de la figure 3.24), et vont participer ainsi à un processus de contention. Le nœud ayant tiré le plus petit délai d'attente répond par une trame CTS, pour s'annoncer comme étant le prochain saut (le nœud 4 dans l'exemple de la figure 3.24). Les autres nœuds arrêtent la contention dès qu'ils entendent la trame CTS. Enfin, un échange DATA et ACK suit entre les deux nœuds spécifiques. Le processus est répété au niveau de chaque saut jusqu'à ce que les données soient livrées à la destination finale

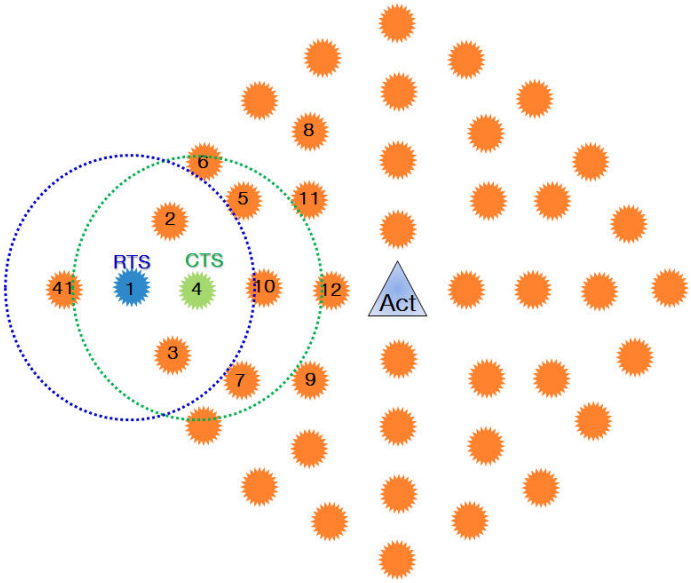


FIGURE 3.24: Principe du protocole du routage RBF

CTP (*Collection Tree Protocol*) [60] utilise ETX (*Expected Transmission Count*) [46] comme une métrique de lien pour la mise en place du gradient. Ainsi, le rang d'un nœud indique combien de fois un message à l'origine de ce nœud est transmis avant qu'il n'atteigne la destination finale. Chaque nœud avec CTP diffuse des messages contenant des informations concernant son rang calculé. CTP utilise l'algorithme Trickle [89] pour régler l'intervalle de l'envoi de ses messages d'annonces. En l'absence de changements de la topologie, cet intervalle est régulièrement doublé jusqu'à ce qu'il atteigne une valeur maximale. Suite à des changements de topologie, le délai est réduit à une valeur minimale pour permettre la convergence rapide du gradient et la valeur de l'intervalle de diffusion est ramenée à sa valeur minimale.

Le groupe de travail ROLL [2] au sein de l'IETF [3] a identifié le routage par gradient comme particulièrement adapté aux réseaux LLNs. Le protocole RPL [155] est en cours de standardisation au sein de ce groupe. Ce protocole exploite la plupart des idées présentées précédemment.

RPL est un protocole de routage comprenant un algorithme de structuration du réseau. La topologie créée par RPL est un graphe orienté acyclique DAG (*Directed Acyclic Graph*) [16]. Un DAG maintient son caractère acyclique en exigeant que chaque nœud du DAG doive avoir un rang plus grand que tous ses parents dans le DAG. Le DAG est construit afin d'être utilisé pour acheminer le trafic dominant dans le réseau qui est le trafic MP2P.

L'objectif de cette topologie est d'offrir un routage efficace et fiable de n'importe quel point du réseau vers la racine du DAG. Comme les liens radio sont fluctuants et volatiles, RPL maintient plusieurs chemins actifs à partir de chaque nœud vers la destination. RPL construit donc un DAG enraciné au niveau du (des) puits ou actionneur(s) appelé DODAG (*Destination Oriented DAG*). Chaque nœud appartenant à la topologie en DAG transmet des messages DIO (DODAG Information Objet). Ces messages contiennent une valeur appelée rang. Ce rang est analogue à la profondeur dans une topologie arborescente, sauf qu'il n'est pas limité à des augmentations de 1 à chaque saut. La racine du DAG s'attribue un rang de 0. La transmission de ces messages DIO est

2. ROLL IETF Working Group : http ://datatracker.ietf.org/wg/roll/charter/
3. Internet Engineering Task Force : http ://www.ietf.org

réglementée par l'algorithme Trickle [89] pour supprimer les messages redondants. En effet comme décrit précédemment, l'intervalle de diffusion des messages de contrôle est doublé tant qu'il n'y a pas des changements dans la topologie. Ainsi, tant que le réseau est stable, le nombre de messages de contrôle envoyés est réduit par Trickle.

Lors de la construction de la topologie en DAG, chaque nœud choisit parmi ses voisins un parent préféré, c'est celui qui offre le « meilleur »rang en se basant sur les métriques et les contraintes utilisées pour la construction du DAG. Ce nœud choisit un parent préféré, en tenant compte des informations de routage de ses voisins et les coûts liées à ses voisins. Il s'attribut un rang égal au rang de son parent préféré incrémenté par le coût pour atteindre ce parent préféré (voir figure 3.25).

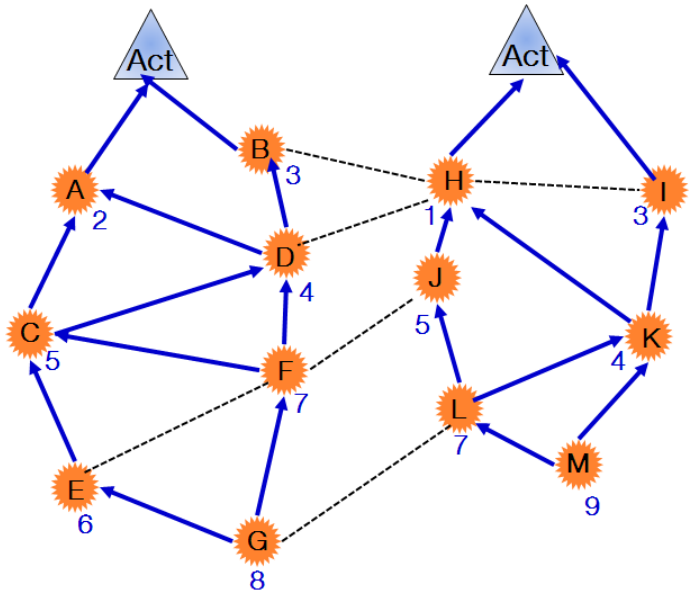


FIGURE 3.25: Principe du protocole RPL

Ce DAG est utilisé pour faciliter l'envoi des données vers la racine du DAG. En ce qui concerne le routage P2P avec RPL, les auteurs du [62] distinguent plusieurs cas de figure : d'abord, si la source et la destination sont à portée de communication, alors la source envoie directement vers la destination. Si la source et la destination ne sont pas à portée de communication, deux modes sont possibles. Le « *storing mode* »où des nœuds sauvegardent des routes vers des nœuds qui se sont annoncés comme des nœuds destinations. Le routage vers ces destinations suit un chemin vers « le haut »en direction du la racine du DODAG jusqu'à atteindre un nœud ancêtre qui connait une route vers cette destination. Cet ancêtre envoie vers « le bas »en direction de la destination. Pour le deuxième mode appelé « *non-storing mode* », la source envoie son message vers « le haut »jusqu'à atteindre la racine du DODAG. La racine du DODAG envoie ce message vers « le bas »en direction de la destination finale. Finalement, pour le trafic P2P, [62] propose de construire un DAG temporaire enraciné au niveau du nœud source, ce DAG temporaire sera construit jusqu'à atteindre et couvrir le nœud destination.

3.5.4 Le routage et l'hétérogénéité

Malgré sa longue histoire, les problématiques et les défis des protocoles de routage pour les WSNs et les WSANs sont loins d'être clos. D'un point de vue industriel, il est intéressant de suivre les activités de l'IETF, et plus particulièrement le groupe du travail ROLL. En effet, le protocole de

routage RPL, en cours de standardisation au sein de ROLL à l'IETF, ne propose pas une solution optimale pour tous les scénarii, mais il permet plutôt d'utiliser différents paramètres pour obtenir une performance acceptable.

D'un point de vue académique, les questions liées à la consommation énergétique, à la mobilité, au passage à l'échelle et à l'hétérogénéité dominent la conception des prochains protocoles de communication et plus particulièrement les protocoles de routage.

De notre point de vue, nous apportant une attention particulière à la caractéristique d'hétérogénéité dans ces réseaux sans fil. Nous essayerons à trouver un point d'intersection entre le monde industriel avec son protocole en cours de standardisation et le monde académique en se concentrant sur la problématique d'hétérogénéité.

3.6 Conclusion

Dans ce chapitre, nous avons introduit la notion d'auto-organisation et son rôle de création de topologie logiquepour faciliter le déploiement des protocoles de routage. Nous avons aussi introduit les principaux protocoles de routage qui se basent principalement sur une topologie logique pour assurer un routage efficace. Cependant, nous remarquons que pratiquement la plupart de ces protocoles n'ont pas été conçus pour tirer bénéfice de la principale caractéristique des réseaux WSANs qui est l'hétérogénéité. Ces protocoles d'auto-organisation et de routage ne sont pas conçus au départ pour exploiter les ressources des nœuds riches (actionneurs) afin de réduire la charge de communication sur les nœuds de faibles ressources (capteurs). Par conséquent, ils ne sont pas les mieux adaptés à un réseau hétérogène. De cette conclusion, nous avons jugé qu'il y a une nécessité de proposer des protocoles qui prennent en compte et tirent bénéfice de l'hétérogénéité dans les WSANs. C'est ce que nous présentons dans les chapitres qui suivent.

3.6 Conclusion

Stratégie d'auto-organisation dans les réseaux hétérogènes de capteurs et actionneurs

4

Sommaire

4.1 Introduction

Comme nous venons de le voir dans le chapitre précédent, les protocoles de contrôle de topologie, MAC et routage proposés pour les WSNs n'ont pas été généralement conçus pour tirer bénéfice de l'hétérogénéité que nous trouvons dans les WSANs [150]. Ils ne sont généralement pas conçus pour exploiter les ressources des nœuds riches (actionneurs) afin de réduire la charge de communication au niveau des nœuds à faibles ressources (capteurs). Par conséquent, ils ne sont pas les mieux adaptés à un réseau hétérogène.

Se basant sur cette idée et sur l'observation que, dans les WSANs, généralement les trafics de données se produisent des nœuds capteurs vers les nœuds actionneurs, nous proposons dans ce chapitre un protocole d'auto-organisation pour les réseaux hétérogènes WSANs profitant des ressources disponibles au niveau des nœuds actionneurs. Dans cette nouvelle proposition, appelée Far-Legos, les nœuds riches en ressources initient et construisent une topologie logique. La nature de cette topologie est différente dans les zones couvertes et dans les zones non-couvertes par ces nœuds riches en ressources. Ainsi, notre proposition sera une combinaison de deux protocoles d'auto-organisation : le premier est appliqué à l'intérieur des zones couvertes par les nœuds riches en ressources. Le second protocole d'auto-organisation est appliqué dans les zones non-couvertes. Dans chaque zone non-couverte, les nœuds à faibles ressources doivent construire une topologie logique permettant d'interconnecter cette zone non-couverte vers une ou plusieurs autres zones couvertes.

En dehors de faciliter l'acheminement du trafic de collecte de données des nœuds à faibles ressources vers les nœuds riches en ressources, la topologie logique dans les zones couvertes et les zones non-couvertes doit conserver l'énergie au niveau des nœuds à faible ressources et doit aussi préserver la propriété de connectivité du réseau.

Nous présentons aussi dans ce chapitre deux variantes de notre proposition appelées respectivement *Oriented-Far* et *Clustered-Far*. L'objectif de ces deux variantes est d'améliorer les performances de notre proposition de base Far-Legos dans certains cas de figures que nous présentons dans ce chapitre.

Nous commençons ce chapitre par une description de la proposition de base Far-Legos. Ensuite, nous évaluerons les performances de cette proposition. Dans la section 4.4, nous présenterons les deux variantes *Oriented-Far* et *Clustered-Far*, leurs rôles et objectifs. Enfin, nous évaluerons les performances de ces deux variantes.

4.2 Far-Legos : Une topologie logique pour un routage convergecast dans les WSANs

4.2.1 Vue d'ensemble

Dans ce chapitre, nous proposons un protocole d'auto-organisation qui profite des capacités des nœuds riches en ressources pour créer une topologie logique afin de faciliter la collecte de données dans les réseaux WSANs. L'objectif de notre proposition, Far-Legos, est de fournir une efficacité énergétique dans les zones couvertes par les nœuds actionneurs et dans les zones non-couvertes, de réduire le délai de bout en bout et d'assurer un taux de livraison élevé. Notre proposition peut être subdivisée en quatre phases.

Durant la première phase un nœud actionneur affecte des rangs aux nœuds capteurs qui se retrouvent dans sa portée de communication (voir la figure 4.1). Ces rangs seront utilisés comme un gradient quand il y a des données à envoyer vers le nœud actionneur. Les capteurs dans la

zone couverte par le nœud actionneur utiliseront ces rangs comme des coordonnées virtuelles pour acheminer les données vers le nœud actionneur.

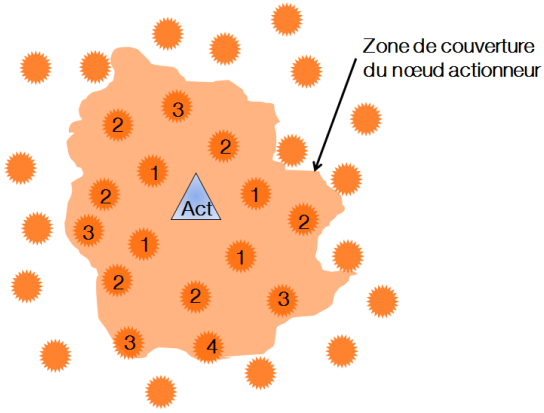


FIGURE 4.1: Phase 1 : affectation de rang dans la zone couverte par l'actionneur

Ensuite, les nœuds capteurs lancent une phase de découverte de voisinage (voir figure 4.2). L'objectif de cette phase de découverte de voisinage est de permettre aux nœuds capteurs de connaitre les nœuds candidats pour être les prochains sauts durant la phase de collecte de données.

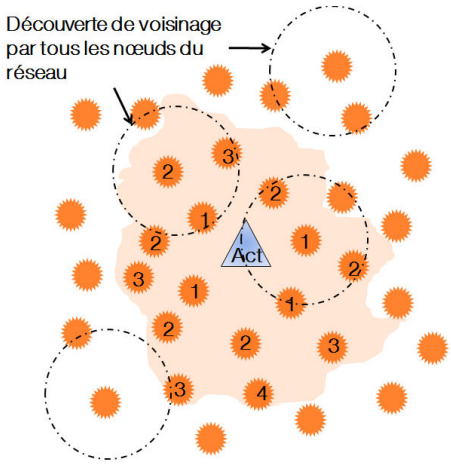


FIGURE 4.2: Phase 2 : phase de découverte de voisinage

Dans une troisième phase, les capteurs dans la zone non-couverte créent une topologie logique en épine dorsale dérivée du protocole d'auto-organisation Legos [97] (voir figure 4.3). Le choix de la topologie en épine dorsale construite avec Legos est principalement basé sur les avantages de la topologie Legos par rapport aux autres topologies [95] [11]. Legos créé une topologie logique d'une manière efficace en énergie et en utilisant des décisions locales pour faire face aux changements dans l'environnement et à la dynamique du réseau (apparition/disparition des liens et arrivée/départ des nœuds dans le réseau). Dans [97], les auteurs montrent que Legos consomme 20% moins d'énergie que les autres protocoles basés sur la diffusion des messages *Hello* périodiques. En effet, Legos offre une cardinalité bornée des nœuds dominants [97]. Ainsi le nombre de nœuds qui diffusent périodiquement des messages *Hello* est limité et ainsi la durée de vie maximale du réseau est prolongée. De plus, en comparant une variante du protocole Legos avec plusieurs autres protocoles d'auto-organisation, les

auteurs du [11] concluent que généralement les protocoles d'auto-organisation basés sur les nœuds dominants sont plus efficaces en termes de débits, de surcoût protocolaire et de consommation énergétique. Plus particulièrement, les auteurs montrent que cette variante du protocole Legos surpasse tous les autres protocoles en termes d'énergie dissipée pendant la phase d'auto-organisation. Ainsi, nous nous proposons d'utiliser ce protocole d'auto-organisation dans les zones non-couvertes pour créer une topologie en épine dorsale dans ces zones. Cette épine dorsale vise à trouver un chemin vers une zone couverte et de préserver l'énergie au niveau des nœuds capteurs à faibles ressources (voir figure 4.3).

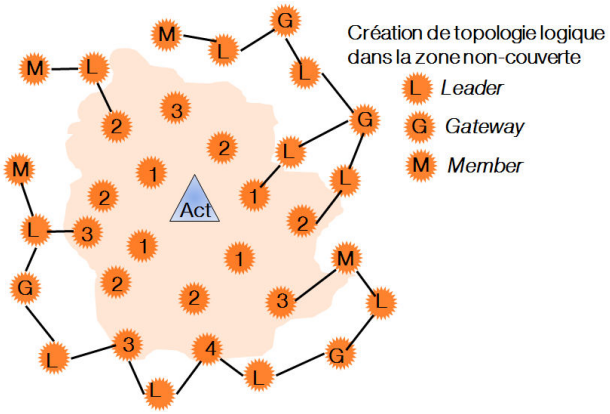


FIGURE 4.3: Phase 3 : création de topologie logique dans la zone non-couverte

Enfin, la quatrième phase est la phase de collecte de données (voir figure 4.4). Dans cette dernière phase, un capteur, ayant des données à transmettre vers le nœud actionneur, cherche parmi ses voisins découverts durant la deuxième phase le nœud le plus approprié pour être le prochain saut auquel il peut envoyer son message de donnée et qui va le faire suivre à son tour en direction du nœud actionneur de destination.

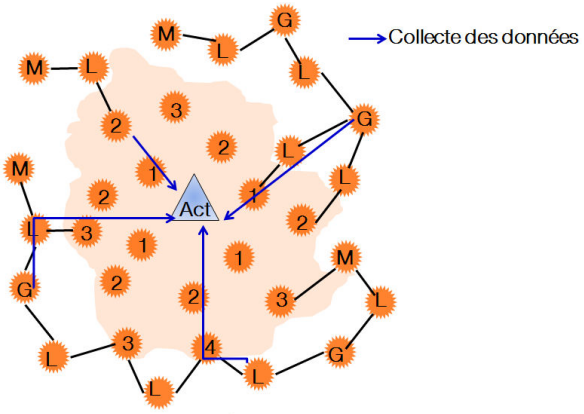


FIGURE 4.4: Phase 4 : phase de collecte de données

4.2.2 Hypothèses

Nous considérons un WSAN constitué d'un grand nombre de nœuds capteurs et un nombre beaucoup moins conséquent de nœuds actionneurs. Nous supposons initialement que tous les nœuds capteurs et actionneurs sont déployés, c'est à dire que tous les nœuds sont placés dans leurs emplacements dans le réseau et sont allumés.

Nous admettons que la puissance de transmission des nœuds actionneurs est plus grande que celles des nœuds capteurs et qu'elle est réglable. Nous supposons aussi qu'il y a une zone non-couverte dans le réseau, c'est à dire qu'il y a des capteurs qui ne sont couverts par aucun actionneur. Les nœuds ne sont pas synchronisés et le modèle de trafic dans le réseau est la collecte de donnée ou convergecast. Nous supposons que les capteurs sont homogènes, c'est-à-dire, qu'ils ont les mêmes capacités de calcul, de mémoire et de transmission. Nous admettons que les nœuds actionneurs connaissent la portée de transmission des nœuds capteurs. Nous rediscutons cette dernière hypothèse dans la section 4.4.

Aucune information géographique n'est disponible pour l'ensemble des nœuds du réseau. Nous supposons aussi que le réseau est stable et statique : pas de mobilité et pas de redéploiement de nouveaux nœuds dans le réseau. Ces dernières hypothèses seront discutées plus loin dans ce chapitre. Nous supposons aussi que les nœuds actionneurs peuvent communiquer directement puisqu'ils ont des portées de communication plus grandes par rapport à la portée des nœuds capteurs.

4.2.3 Description de Far-Legos

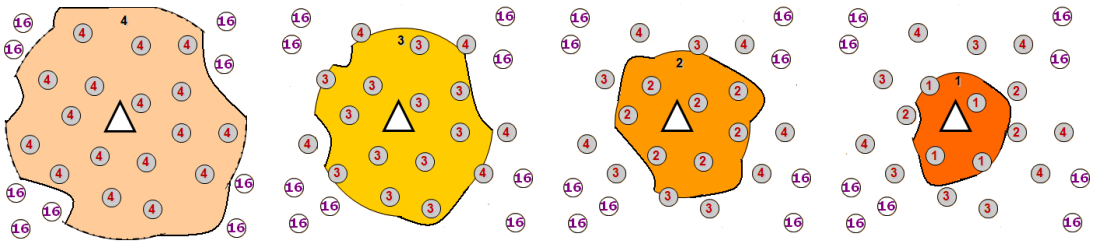
L'idée de base de notre proposition Far-Legos est de profiter de la capacité d'énergie et de la puissance de transmission importante des nœuds actionneurs. Nous proposons d'utiliser cette puissance de transmission réglable pour affecter des rangs qui serviront comme gradient pour la structuration du réseau en vue d'une remontée des données à partir des nœuds capteurs vers les nœuds actionneurs. Ainsi, nous pouvons décomposer notre proposition en quatre phases : une phase d'affectation de rang, une phase de découverte de voisinage, une phase de structuration dans les zones non-couvertes et une dernière phase d'acheminement des données vers les nœuds actionneurs :

Phase d'affectation de rang

Nous proposons que chaque nœud actionneur affecte des rangs aux nœuds capteurs dans son voisinage en diffusant des messages contenant ces rangs. L'affectation de ces rangs sera assurée par une diffusion avec des puissances successives décrémentées. Ainsi les capteurs plus proches du nœud actionneur auront un plus petit rang.

A chaque étape, l'actionneur attribue un rang aux nœuds d'une couronne autour de lui. L'épaisseur de cette couronne est égale par défaut à la portée radio des nœuds capteurs. Ainsi, un capteur appartenant à la couronne N peut communiquer avec les capteurs appartenant aux couronnes $(N-1)$ ou $(N+1)$. En outre, nous proposons d'utiliser une affectation de rang basée sur des diffusions avec des puissances d'émission décrémentées parce qu'elle est plus économe en énergie que celle basée sur des diffusions de message d'affectation avec des puissances d'émission incrémentées. En effet, intuitivement lorsque les nœuds qui sont plus loin de l'actionneur ont reçu un rang, ils peuvent passer en mode d'économie d'énergie (éteindre leur radio par exemple).

La Figure 4.5 représente un exemple d'affectation de rang par un actionneur. Ce dernier commence par diffuser dans cet exemple le rang 4 avec sa puissance maximale P4. Les capteurs recevant ce message et qui n'ont pas un encore un rang vont lire cette valeur de rang contenu dans ce message



(a) Puissance P4 (b) Puissance P3 (c) Puissance P2 (d) Puissance P1

FIGURE 4.5: Exemple d'affectation de 4 rangs aux capteurs autour d'un nœud actionneur

et vont se l'affecter (Figure 4.5(a)). Ensuite le nœud actionneur diffuse un second message avec une puissance P3 ($< P4$). Les capteurs recevant ce message ayant un rang supérieur à celui diffusé vont mettre à jour leur rang (Figure 4.5(b)). Ce processus va être répété jusqu'à ce que le nœud actionneur atteigne sa valeur de puissance de transmission minimale P1 (Figure 4.5(d)). Cette valeur minimale est la puissance de transmission des nœuds capteurs présents dans le réseau. Les nœuds qui n'ont pas reçu de message d'affection de rang, gardent la valeur par défaut de rang (16 dans cet exemple de la Figure 4.5).

Phase de découverte de voisinage

Après la phase d'affectation rang, chaque nœud capteur envoie un message *Hello* pour découvrir son voisinage. Chaque actionneur envoie un message de découverte de voisinage en utilisant sa puissance de transmission minimale (qui est égale à la puissance de transmission des nœuds capteurs). Chaque nœud, avant d'envoyer ce message de découverte de voisinage, calcule un temps d'attente aléatoire pour éviter les collisions entre les messages *Hello* échangés.

Dans ce message de découverte, chaque nœud met son identité et son rang (égale à 0 pour les nœuds actionneurs et la valeur par défaut pour les nœuds capteurs dans les zones non-couvertes). À la fin de cette phase, chaque nœud aura une liste de ses voisins et leurs rangs. Deux cas se présentent :

- Tout d'abord, dans la zone couverte, chaque nœud capteur peut classer ses voisins généralement en trois types : les capteurs les plus proches du nœud actionneur (ayant un rang plus petit), les capteurs les plus loin du nœud actionneur (ayant un rang plus élevé) et les capteurs dans la même couronne (ayant le même rang).
- Deuxièmement, dans la zone non-couverte, les capteurs qui détectent des nœuds avec un rang différents du rang par défaut concluent qu'ils sont reliés à une zone couverte. Ils mémorisent ce fait pour être les premiers nœuds qui lancent la troisième phase détaillée ci-dessous.

Phase de construction de topologie dans les zones non-couvertes

Dans cette phase, une topologie logique est construite dans les zones non-couvertes par aucun actionneur. La topologie construite dans les zones non-couvertes se base sur le protocole d'auto-organisation Legos [97]. En effet, Legos se base sur la construction d'une épine dorsale. Chaque nœud non-couvert commence la construction de la topologie à $t = T_{start\_Legos}$, calculé comme dans l'équation 4.1 ci-dessous.

Comme décrit dans la formule, chaque nœud capteur attend un $Timeout_{neighbor\_discovery}$ qui correspond à la durée pendant laquelle un nœud capteur peut recevoir des messages *Hello* de la part des nœuds dans son voisinage. Puis, il attend un temps aléatoire qui dépendra de la situation du nœud dans la zone non-couverte. Par conséquent, les nœuds à proximité de la zone couverte ont une chance d'être les premiers à commencer la construction de la topologie.

$$T_{start\_Legos} = Timeout_{neighbor\_discovery} + Rand(X) + Timeout_{Leader\_detection} \tag{4.1}$$

Les autres nœuds non-couverts commencent la construction et rejoignent la topologie plus tard. Nous utilisons ce temps aléatoire pour introduire une priorité entre les nœuds et aussi pour éviter que plusieurs nœuds démarrent la construction de la topologie en même temps. Il est à noter que même si plusieurs nœuds démarrent la construction de plusieurs topologies en même temps ça ne pose pas de problème, puisque Legos proposent des techniques de fusion de topologies [97]. Enfin, chaque nœud non-couvert devrait attendre un $Timeout_{Leader\_detection}$ qui correspond à la période séparant deux messages diffusés par les *Leaders* de la topologie Legos dans la zone non-couverte. Ces *Timers* sont dimensionnés par défaut comme dans [97].

Phase de collecte de données

Chaque capteur disposant de données à envoyer vers un nœud actionneur applique l'algorithme de collecte de données décrit ci-après. Cette phase dépend de l'emplacement du nœud capteur (dans une zone couverte ou une zone non-couverte).

- **Dans une zone couverte**, un nœud capteur devant envoyer des données vers un actionneur choisit le prochain saut à partir de sa liste des voisins : si la destination (l'actionneur) n'est pas un voisin direct, le prochain saut sera un voisin ayant un rang plus petit que celui du nœud actuel et ayant le plus petit rang parmi tout le voisinage. Par conséquent, les messages de données parcourent les nœuds jusqu'à atteindre le nœud destination. Si le nœud actuel ne possède pas un voisin avec un plus petit rang, il comprend qu'il y a un trou dans le réseau et transmet son paquet de données à un voisin ayant le même rang. Si ce dernier n'existe pas, le nœud courant choisit un nœud ayant un rang plus élevé, pour tenter de contourner ce trou. Le recours à un acheminement via des voisins ayant un même rang ou un rang plus grand augmente le taux de livraison comme c'est montré dans [87]. Cependant, cela peut créer aussi un risque de boucles. Nous discutons ce cas de figure dans la section 4.4.
  Dans le cas où un nœud envoie vers un nœud ayant un rang supérieur ou égal, il se souvient de l'identifiant de ce message (son numéro de séquence et l'identifiant du nœud source par exemple). Ainsi, quand ce nœud reçoit un message, il vérifie d'abord si ce message est passé par lui. Si c'est le cas, il refuse de participer à l'acheminement de ce message et demande explicitement à l'expéditeur de trouver un autre prochain saut.

- **Dans la zone non-couverte**, un capteur ayant une donnée destinée à un actionneur vérifie d'abord s'il est directement relié à une zone couverte : si c'est le cas, il envoie les données vers le nœud couvert le plus proche indépendamment de son rôle dans la topologie Legos. Si ce n'est pas le cas, un nœud, qu'il soit un *Member* ou un nœud *Gateway* envoie les données vers son nœud *Leader*. Ce dernier vérifie d'abord s'il a un nœud (*Member* ou *Gateway*) qui soit connecté à une zone couverte. Si ce *Leader* en a un, il envoie les données vers ce nœud qui les transmettra

vers la zone couverte. Si le *Leader* n'a pas de nœud connecté à une zone couverte alors il envoie les données sur l'épine dorsale jusqu'à ce qu'un nœud *Leader* de cette épine dorsale trouve une connexion à une zone couverte.

4.2.4 Présence de plusieurs actionneurs dans le réseau

Lorsqu'il y a plusieurs actionneurs dans le réseau, chacun d'entre eux tire dès le déploiement un temps aléatoire avant de lancer la phase d'affectation de rang. Ce temps aléatoire avant les diffusions est utilisé pour éviter les collisions des messages d'affectation de rang au niveau des nœuds capteurs. Dans les messages d'affectation de rang, les actionneurs peuvent spécifier le type de donnée les concernant. Chaque nœud capteur dans le réseau garde alors une liste des actionneurs avec lesquels il a un rang et, durant la phase de découverte de voisinage, chaque capteur envoie sa liste des actionneurs dans le message *Hello*. Ainsi lorsqu'un capteur détecte un évènement à reporter vers un actionneur, il consulte sa liste d'actionneurs et choisit l'actionneur approprié. Sans perdre de généralité, nous supposons que chaque capteur envoie les évènements détectés vers l'actionneur le plus « proche ». Comme nous supposons que les actionneurs communiquent directement entre eux, les informations envoyées par les capteurs vont arriver chez l'actionneur le plus « proche »du nœud source et c'est cet actionneur qui se charge d'informer ou non les autres actionneurs du réseau de l'évènement détecté.

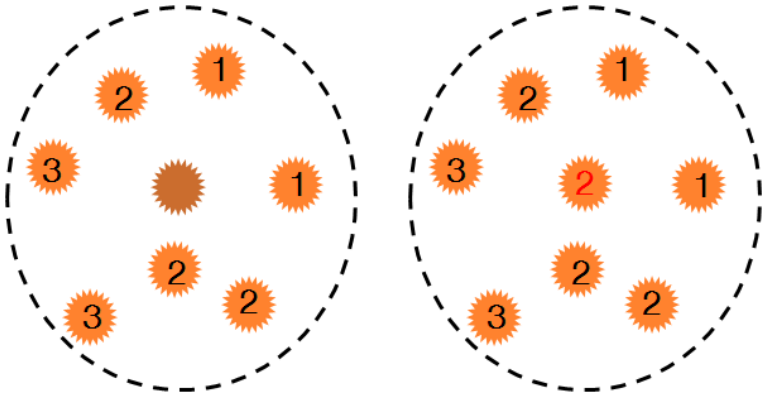
4.2.5 Déploiement de nouveaux nœuds dans le réseau

Notre proposition part du principe que le réseau est stable et statique, mais comme Legos, notre proposition Far-Legos fournit des mécanismes pour tenir compte de la dynamique provoquée par le déploiement de nouveaux nœuds capteurs ou actionneurs dans le réseau.

D'une part, quand un nouvel actionneur est déployé dans le réseau, il commence par inviter les nœuds capteurs dans son voisinage à une phase d'affectation de rang. Ainsi, le nouvel actionneur diffuse des messages d'affectation de rang comme décrit dans 4.2.4. Par conséquent, les nœuds capteurs créent une nouvelle entrée dans la liste de leurs actionneurs. Par la suite, la phase de découverte de voisinage est relancée.

D'autre part, quand un nouveau capteur est déployé dans le réseau, il découvre d'abord son voisinage en envoyant un paquet *Hello*. Les nœuds déjà présents dans le réseau et recevant ce message *Hello* répondent en envoyant leurs rangs. Trois cas se présentent :

- **D'abord**, lorsque ce nœud récemment déployé ne détecte que des nœuds dans une zone non-couverte, il rejoint la topologie Legos et conserve la valeur par défaut pour son rang.
- **Deuxième cas**, quand ce nouveau nœud ne détecte que des nœuds dans une la zone couverte (ayant un rang différent de celui par défaut), il calcule son rang qui correspond au rang les plus significatif parmi les rangs reçus. Après le calcul de son rang, le nœud nouvellement déployé envoie un deuxième message *Hello* pour informer ses voisins de son nouveau rang. Les voisins ajoutent ce capteur comme une nouvelle entrée dans leur table de voisinage. Ainsi, ce nœud est intégré dans la topologie de la zone couverte sans que cela l'affecte. Prenons l'exemple de la figure 4.6, le nouveau nœud détecte 2 voisins de rang 3, deux de rang 1 et trois voisins de rang 2. Ce nouveau nouveau s'affectera le rang 2 comme représenté dans la figure 4.6(b).
- **Enfin**, dans le cas où le nouveau nœud détecte des nœuds dans une zone couverte et des nœuds dans une zone non-couverte, il enregistre la liste des nœuds couverts et leurs rangs. Puis il commence à joindre la topologie Legos dans la zone non-couverte, en annonçant à la topologie qu'il se trouve sur la frontière d'une zone couverte. Nous avons choisi que ce nouveau



(a) Déploiement d'un nouveau nœud

(b) Le nouveau nœud calcule son nouveau rang

FIGURE 4.6: Procédure de calcul de rang pour un nouveau nœud déployé dans le réseau

nœud doit rejoindre la topologie de la zone non-couverte afin d'augmenter le nombre de nœuds qui peuvent relier cette zone non-couverte à la zone couverte.

4.3 Evaluation de performance Far-Legos

4.3.1 Paramètres de simulation

Pour les résultats qui suivent, chaque protocole a été implémenté sur le simulateur WSNet [67]. La modélisation de la couche physique prend en compte un modèle de propagation à deux rayons [39] et des interférences orthogonales. En environnement réel, il est peu probable que le seul chemin de propagation entre une source et une destination soit le chemin direct. Le modèle à deux rayons utilisé pour toutes les simulations considère à la fois le chemin direct et une réflexion sur le sol. Cela implique que le voisinage n'est pas nécessairement persistant et que des pertes de paquets peuvent subvenir. La cardinalité du réseau est de 121 nœuds distribués sur deux topologies : une topologie en grille régulière et une topologie en grille aléatoire. La puissance d'émission des nœuds actionneurs varie entre 2 à 5 fois celle des nœuds capteurs. La puissance d'émission des nœuds capteurs est fixée et ne forme généralement pas un disque unitaire.

La couche MAC (Medium Access Control) utilisée est le mode non coordonné du standard 802.15.4. Ce mode est généralement utilisé pour ces types de réseaux (WSNs et WSANs) où les nœuds sont souvent en veille (qui dorment la majorité du temps). Cette solution a pour avantage d'optimiser l'autonomie des batteries des capteurs et d'utiliser le canal uniquement lorsqu'il est nécessaire de transmettre des données utiles.

Les messages de données sont générés par les capteurs du réseau et envoyés au(x) nœud(s) actionneur(s) selon un routage multi-sauts. Le routage suit la topologie logique construite par les algorithmes d'auto-organisation déclenchés au début du déploiement des nœuds.

Suivant une loi aléatoire et uniforme, nous déclenchons plusieurs évènements applicatifs qui seront détectés chacun par un nœud capteur et remontés vers le nœud actionneur approprié. Le nombre d'évènements dans le réseau varie entre 10 et 50 évènements. Chaque scénario est simulé entre 100 et 200 fois avec un intervalle de confiance de 95%.

Pour les scénarii à plusieurs actionneurs, nous choisissons d'une manière aléatoire leurs emplacements dans le réseau. Nous supposons aussi que les évènements seront reportés vers l'actionneur avec lequel le nœud capteur détectant l'évènement a la plus faible valeur de rang.

Le tableau 4.1 résume les principales caractéristiques du réseau.

Paramètre	Valeur
Topologie	Grille régulière et Grille aléatoire
Nombre de nœuds	121 nœuds
Nombre d'actionneurs	1 .. 5
Modèle de trafic	Evénementiel (1 événement à la fois)
Nombre d'évènements indépendants	10 .. 50
Nombre moyen de voisins / nœud	8 nœuds
Portée de transmission des capteurs	15m .. 45m
Largeur des couronnes	15m .. 45m
Protocole MAC	802.15.4 (CSMA)
Propagation	Modèle à deux rayons
Nombre de sauts maximum autorisé	16 sauts
Intervalle de confiance	95%
Simulateur	WSNet [67]

TABLE 4.1: Paramètres de simulation communs pour Far-Legos et BBDD

4.3.2 Analyse de complexité

Dans cette section, nous commençons par faire une étude de complexité de notre proposition et la comparer avec le protocole BBDD [96]. Nous avons choisi de comparer notre proposition Far-Legos avec le protocole BBDD parce qu'ils se basent tous les deux sur le même protocole d'auto-organisation Legos [97]. En effet, BBDD construit la topologie en épine dorsale basée sur Legos dans tout le réseau, Far-Legos construit cette même topologie seulement dans les zones non-couvertes par les nœuds actionneurs. Le tableau 4.2 récapitule cette comparaison.

Notation

Nous supposons que :
- N : Nombre de nœuds dans le réseau.
- M : Nombre de nœuds *Leader* dans la topologie construite par le protocole Legos pour BBDD ($M < N$).
- L : Nombre de nœuds *Leader* dans la topologie construite dans les zones non-couvertes par Far-Legos ($L < M$).
- Δ : Degré maximal des nœuds.

Complexité de message

Durant l'exécution de notre proposition Far-Legos, le coût d'échange de message pour un nœud afin de joindre la topologie est de $O(\Delta)$, comme dans Legos et BBDD, parce que chaque nœud envoie un nombre constant de message de contrôle (un message de découverte de voisinage et un

message d'attachement à la topologie dans les zones non-couvertes et Δ messages de la part de ses voisins).

Dans [95], l'auteur montre que dans Legos le nombre de nœuds *Leader* et *Gateway* est borné et indépendant du nombre de nœuds dans le réseau. Puisque avec BBDD seulement les nœuds *Leaders* envoient des messages de contrôle périodiquement, alors la complexité de message de BBDD est de l'ordre de O(M.Δ). En ce qui concerne notre proposition Far-Legos, la complexité de message est de l'ordre de O(Δ) dans les zones couvertes (puisque les nœuds évitent d'échanger des messages de contrôle périodiques) et O(L.Δ) dans les zones non-couvertes (puisque seulement les nœuds *Leaders* envoient périodiquement des messages de diffusion).

Complexité de calcul

La proposition Far-Legos a la même complexité de calcul du protocole BBDD. Cette complexité est de l'ordre de O(Δ) puisque dans les deux cas les nœuds parcourent leurs listes de voisinage pour déterminer le prochain saut durant la phase de collecte de donnée.

Algorithme	Info. réseau	Info. de localisation	Compl. de calcul	Compl. de message
BBDD	Locale	Non	O(Δ)	O(M.Δ)
Far-Legos	Locale	Non	O(Δ)	O(L.Δ) (<O(M.Δ))

Table 4.2: Etude de complexité : Far-Legos et BBDD

4.3.3 Evaluation de la consommation énergétique de la phase d'affectation de rang

Nous commençons par évaluer le coût en émission et en réception des messages durant la phase d'affectation de rang. Nous comparons notre proposition d'affectation de rang basée sur des diffusions avec des puissances d'émissions décrémentées à une technique connue dans la littérature se basant sur les MPRs [122]. Les techniques basées sur les MPRs sont l'exemple typique assurant ce type de mission puisque l'objectif de cette phase est d'assurer l'acheminement d'un message d'affectation de rang à partir d'un nœud source (un nœud actionneur) vers tous les nœuds qu'il peut couvrir.

Nous supposons un déploiement uniforme régulier avec une densité géographique fixe. Nous supposons aussi que les messages envoyés pour l'affectation de rang par Far-Legos et par MPR sont de même taille. Pour simplifier le calcul, nous supposons une consommation énergétique linéaire et que le coût d'une unité de temps de transmission sera égal au coût d'une unité de temps de réception. En effet, concernant la consommation énergétique en émission et en réception, les auteurs de [124] montrent que la consommation en émission et en réception pour des capteurs de type *MEDUSA* similaires aux nœuds développés dans le projet *SmartDust* [78] ou de type *Rockwell WINS*[1] sont presque égales.

Dans cette partie, nous avons choisi de nous intéresser à l'énergie consommée en émission et en réception. Ainsi, nous ne prenons pas en compte l'énergie consommée par l'écoute au niveau d'un nœud avant la réception d'un message d'affectation de rang. Intuitivement, nous pouvons déduire que l'énergie consommée dans ce cas au niveau de la proposition Far-Legos est inférieure à celle consommée par MPR. En effet, avec Far-Legos, les nœuds capteurs qui sont loin du nœud actionneur sont atteints plus rapidement par les messages d'affectation de rang que dans le cas de MPR. En

1. WINS project, Rockwell Science Center : http ://wins..rsc.rockwell.com

effet, dans MPR, avant de passer d'une couronne à une autre, la phase de découverte de voisinage prolonge la période d'écoute passive des nœuds des couronnes suivantes.

Nous avons choisi aussi de comparer notre proposition Far-Legos avec une variante du protocole MPR, où nous supposons la connaissance du voisinage est gratuite. En d'autre termes, nous comparons aussi Far-Legos avec une variante, que nous appelons *MPR-sans-Hello*, sans tenir compte des paquets *Hello* échangés. En effet, il faut échanger des messages *Hello* entre les nœuds pour sélectionner ceux qui vont relayer les messages de diffusion. Nous supposons que la connaissance de voisinage pour cette extension de MPR est disponible gratuitement (les tables de voisinage sont préenregistrées au niveau des nœuds avant le déploiement par exemple).

La figure 4.7 représente, pour un degré moyen fixe (8 voisins par nœud) et une portée théorique de transmission des nœuds capteurs égale à la largeur des couronnes créées par le nœud actionneur, la quantité d'énergie consommée en émission et en réception en fonction du nombre de couronne dans le réseau. Nous remarquons que pour un petit nombre de couronne autour d'un nœud actionneur (10 et 17 couronnes autour de l'actionneur), notre proposition Far-Legos consomme moins d'énergie que *MPR-sans-Hello* et *MPR-avec-Hello* respectivement. Pour un nombre important de couronnes, la consommation énergétique de Far-Legos devient supérieure à celle de *MPR-sans-Hello* (à partir de 10 couronnes) et à celle de *MPR-avec-Hello* (à partir de 17 couronnes). En effet, pour affecter un rang aux nœuds appartenant à la couronne (N+1), Far-Legos engendre une réception de ce message d'affectation de rang par tous les nœuds dans les couronnes inférieures. Alors que pour les deux variantes de MPR, l'ajout d'une couronne (N+1) engendre seulement une réception de ce message d'affectation de rang au niveau des nœuds de la couronne N.

Ainsi, pour une organisation autour de chaque actionneur de diamètre inférieure à 17, une stratégie d'organisation Far-Legos sera toujours préférable qu'une organisation en MPR. Notons que dans les réseaux WSNs et WSANs urbains supposent des diamètres faibles, faisant de Far-Legos une solution adéquate.

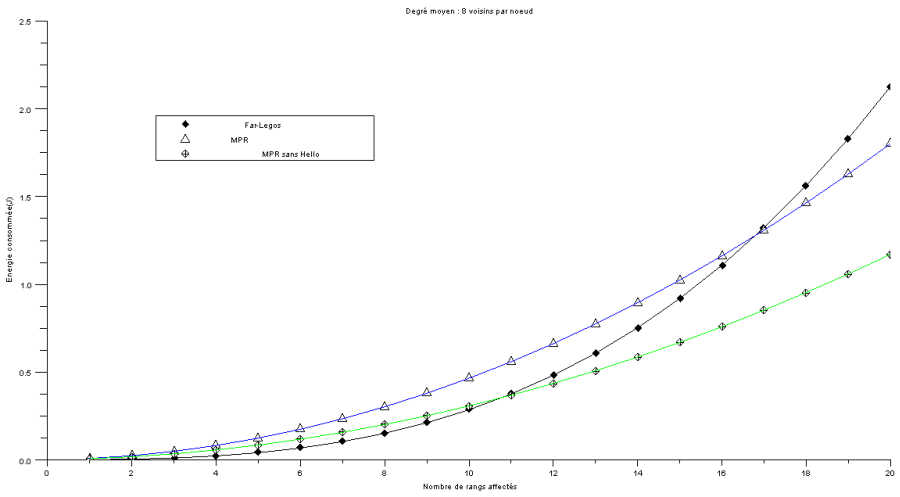


FIGURE 4.7: Comparaison de la consommation énergétique de la phase d'affectation de rang

4.3.4 Performances en fonction de la portée radio et la largeur des couronnes

Dans cette section, nous évaluons l'évolution du délai moyen de bout-en-bout en fonction des portées de communication des nœuds capteurs (Figure 4.8(a)) et du taux de livraison en fonction de la largeur des couronnes qui entourent le nœud actionneur central (Figure 4.9(a)). Nous évaluons aussi la distribution du délai et du taux de livraison en fonction des rangs des nœuds respectivement dans les figures 4.8(b) et 4.9(b). Nous entendons par délai moyen de bout-en-bout le temps que passe en moyenne un message envoyé par un nœud capteur source jusqu'à atteindre la destination finale (un nœud actionneur).

D'une part, nous remarquons que le délai diminue lorsque nous augmentons la portée radio des nœuds capteurs dans le réseau avec une largeur fixe des couronnes (Figure 4.8(a)). Sur la figure 4.8(b), la distribution du délai montre aussi que le délai moyen diminue quand nous augmentons le nombre de nœuds actionneurs dans le réseau.

En effet, dans ce cas de figure, les messages envoyés par les nœuds capteurs seront capables de faire moins de sauts pour atteindre la couronne suivante et par la suite le nœud actionneur de destination.

A partir de ces premiers résultats, nous pouvons conclure que plus la largeur des couronnes autour des nœuds actionneurs est faible, plus les nœuds capteurs ont des voisins ayant des rangs plus petits pouvant être des candidats pour l'acheminement des données. Mais la décomposition du réseau sous la forme de couronnes très fines est très coûteuse en termes d'énergie consommée. Nous vérifions alors dans ce qui suit la largeur optimale que doit avoir les couronnes construites autour des nœuds actionneurs.

D'autre part, pour une portée fixe des nœuds capteurs, nous avons choisi d'évaluer le taux de livraison des messages de données dans le réseau. Lorsque nous augmentons la largeur de couronnes autour du nœud actionneur le taux de livraison diminue (Figure 4.9(a)). Ceci est dû au fait qu'un message de données fait plusieurs sauts dans la même couronne avec le même rang. Ainsi, les boucles ne peuvent pas être évitées et les messages atteignent le nombre de saut maximum autorisé qui est égale à 16 sauts dans nos scénarii de simulation. Nous pouvons vérifier, dans la figure 4.9(b), que plus les nœuds sont loin de la destination, plus le taux de livraison des messages envoyés par ces nœuds se dégrade. Même quand nous augmentons le nombre de nœuds actionneurs dans le réseau, nous remarquons que pour des couronnes avec une largeur supérieure à la portée radio, le taux de livraison reste faible.

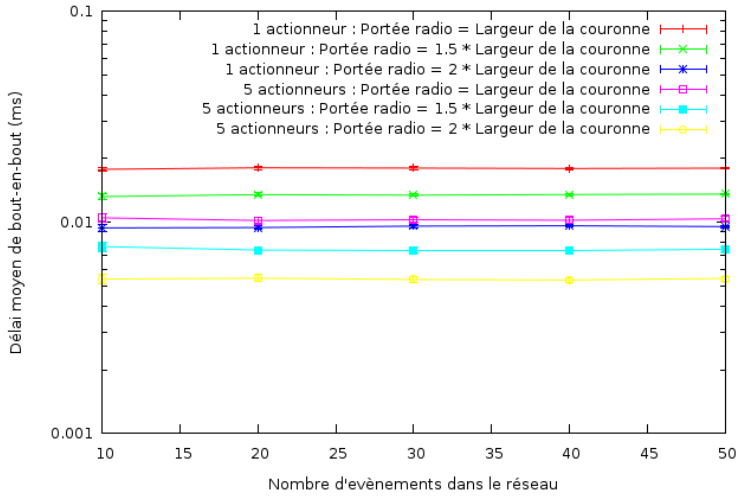
D'après ces résultats de simulations, nous pouvons déduire que pour offrir un meilleur taux de livraison tout en assurant un meilleur délai de bout-en-bout, la largeur des couronnes autour des nœuds actionneurs doit être égale à la portée radio des nœuds capteurs.

Ainsi, dans toutes les simulations qui suivent, nous supposons que la largeur des couronnes est égale à la portée des nœuds capteurs.

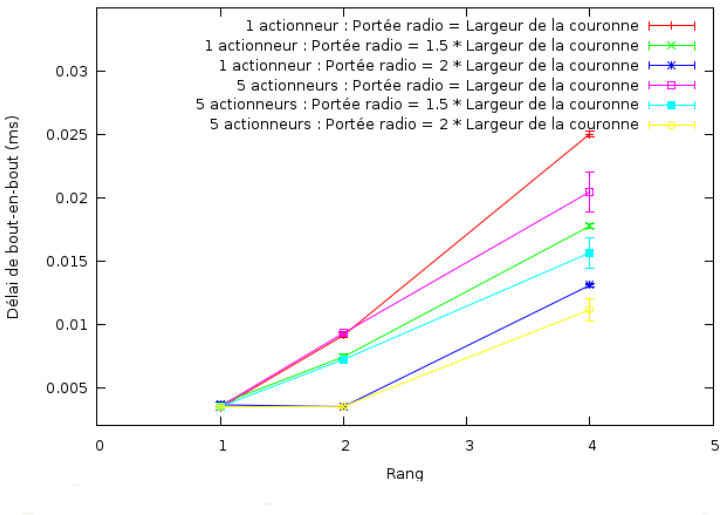
4.3.5 Evaluation du délai et du taux de livraison de Far-Legos

Dans cette section, nous nous intéressons à deux topologies : une topologie en grille régulière et une topologie en grille aléatoire.

Pour les scénarii avec un seul actionneur, nous supposons que cet actionneur est déployé au centre de la topologie. Nous faisons varier la taille de la zone non-couverte dans le réseau en faisant varier la portée de(s) nœud(s) actionneur(s). Nous comparons les performances de notre proposition Far-Legos avec les performances obtenues avec le protocole BBDD [96].



(a) Délai moyen de bout-en-bout

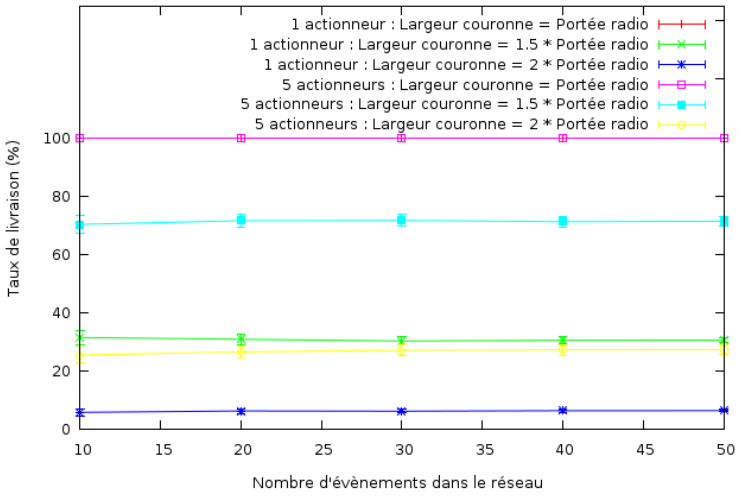


(b) Distribution du délai moyen

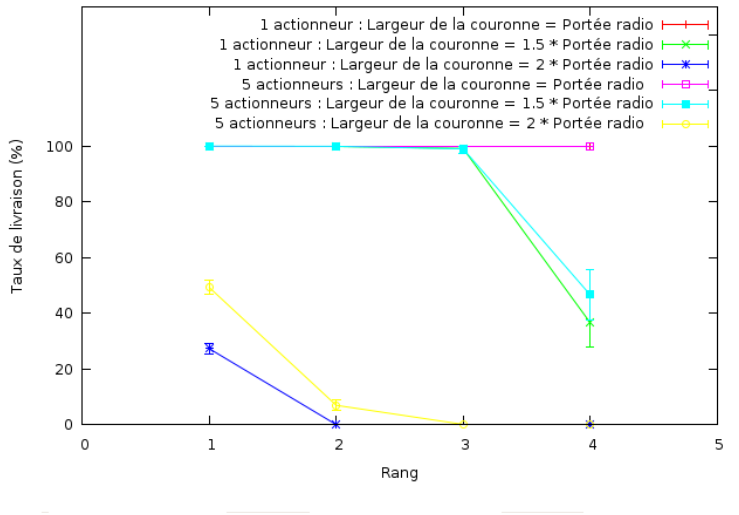
FIGURE 4.8: Variation du délai moyen en fonction de la portée radio des nœuds capteurs

Pour la topologie en grille régulière, les deux figures 4.10(a) et 4.10(b) représentent les topologies obtenues avec notre proposition Far-Legos pour une petite et large zone non couverte respectivement. Les figures 4.11(a) et 4.11(b) représentent les topologies obtenues avec notre proposition Far-Legos pour une petite et large zone non couverte respectivement avec un déploiement des nœuds capteurs sur une grille aléatoire.

Tout d'abord, pour une petite zone non-couverte, nous notons que Far-Legos offre un meilleur délai de bout-en-bout que BBDD pour les deux types de topologies. En effet, les messages de données passent par moins de sauts quand ils voyagent à travers la zone couverte (voir la figure 4.12).



(a) Taux de livraison moyen

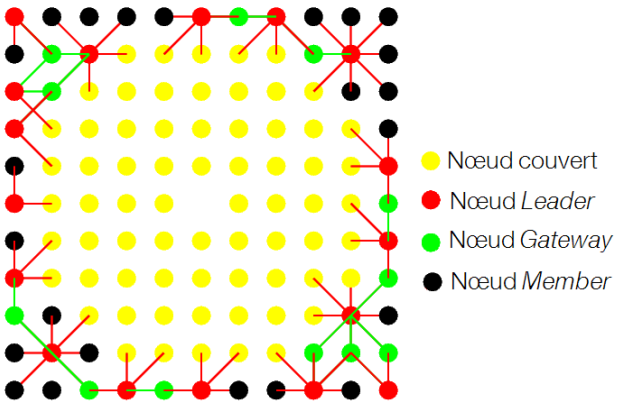


(b) Distribution du taux de livraison

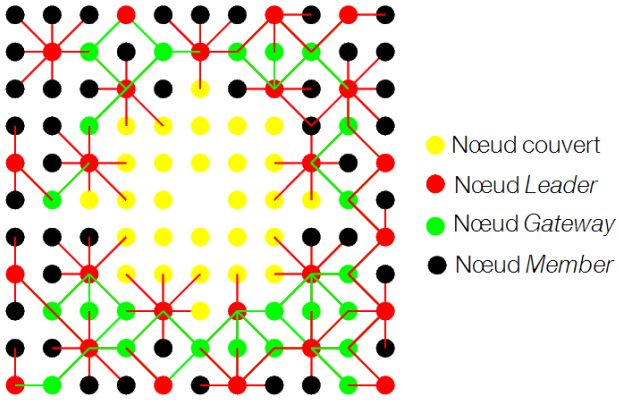
FIGURE 4.9: Variation du taux de livraison en fonction de la largeur des couronnes

Quand nous augmentons le nombre d'actionneurs dans le réseau, le délai des messages utilisant Far-Legos est inférieur à celui offert par BBDD et la différence entre les délais offerts par les deux protocoles augmente. En effet, comme les nœuds capteurs envoient vers l'actionneur le plus « proche »(avec lequel ils ont le plus petit rang), le nombre de sauts séparant la source de données et l'actionneur de destination diminue ainsi que le délai de bout-en-bout. Pour BBDD, suivre l'épine dorsale construite avec Legos allonge le chemin et augmente par la suite le délai. Ainsi, notre proposition Far-Legos sait tirer partie de l'hétérogénéité dans ce type de réseu.

Deuxièmement, pour une large zone non-couverte, nous évaluons le taux de livraison de Far-Legos



(a) Petite zone non-couverte



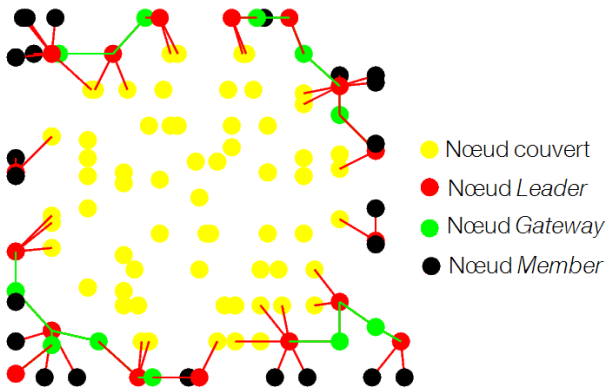
(b) Large zone non-couverte

FIGURE 4.10: Topologies en grille régulière

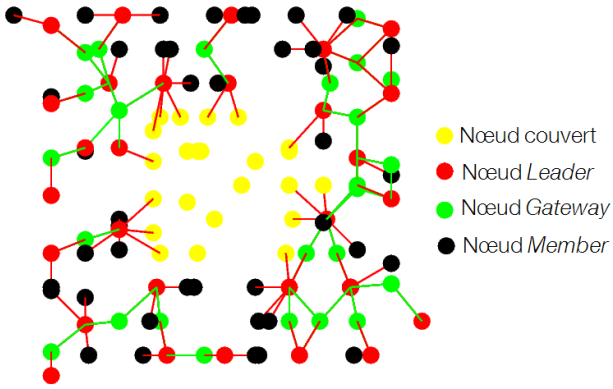
et de BBDD. Dans la figure 4.13, nous représentons le taux de livraison lorsque le nombre de nœuds sources de messages de données augmente dans le réseau pour les deux topologies et pour un et plusieurs actionneurs dans le réseau.

Nous notons qu'avec les deux cas le taux de livraison est proche de 100%. Avec BBDD, tout le réseau est couvert par l'épine dorsale et ce dernier est orienté. Avec Far-Legos, et dans les zones non-couvertes, les nœuds capteurs ont le même rang. L'acheminement des données ne peut pas éviter les boucles. Malgré ce point critique d'envoi aléatoire qui se présente dans les zones non-couvertes, nous remarquons que le taux de livraison de Far-Legos est assez élevé.

Ainsi, nous venons donc de vérifier, dans cette section, que notre proposition Far-Legos réduit le délai de bout-en-bout surtout dans les zones couvertes et offre un taux de livraison élevé en présence d'une petite ou d'une large zone non-couverte.



(a) Petite zone non-couverte



(b) Large zone non-couverte

FIGURE 4.11: Topologies en grille aléatoire

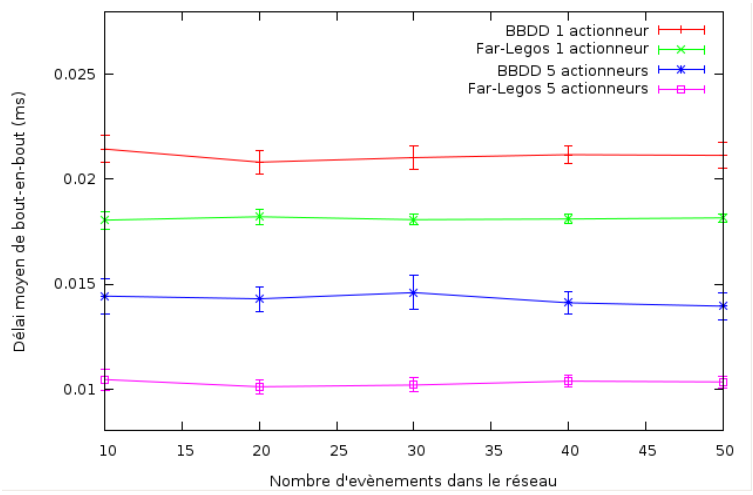
4.3.6 Evaluation du nombre de sauts de Far-Legos

Dans cette section, nous comparons le nombre de sauts moyen que fait un message de données avant d'atteindre la destination finale avec Far-Legos et avec BBDD. Dans la figure 4.14, nous comparons le nombre moyen de sauts offert par Far-Legos et par BBDD comparé au nombre de sauts obtenu avec un algorithme du plus court chemin pour une petite et large zone non-couverte.

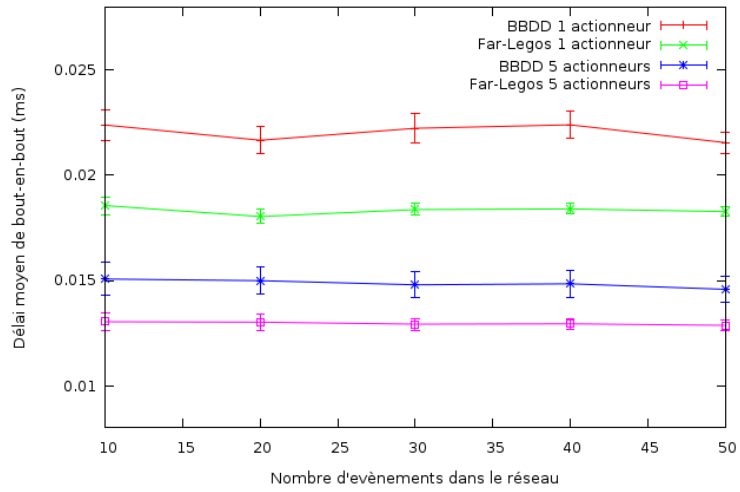
Le nombre de sauts obtenu avec Far-Legos est inférieur à celui obtenu avec BBDD dans tous les scénarii. Cela est dû au fait que, dans la zone couverte, Far-Legos offre un nombre de saut inférieur à celui de BBDD et proche du nombre de sauts obtenu avec l'algorithme du plus court chemin.

En fait, dans la zone couverte, le nombre de sauts effectués par Far-Legos est optimal : en effet, puisque la portée radio de nœuds capteurs est égale à la largeur des couronnes autour des nœuds actionneurs, chaque capteur dans la zone couverte a au moins un voisin avec un rang inférieur. Ainsi, dans la zone couverte, le message ne fait qu'un saut par rang jusqu'à ce qu'il atteigne la destination finale (un nœud actionneur).

Alors que pour BBDD, chaque nœud *Member* doit envoyer ses données vers son *Leader*, même



(a) Topologie en grille

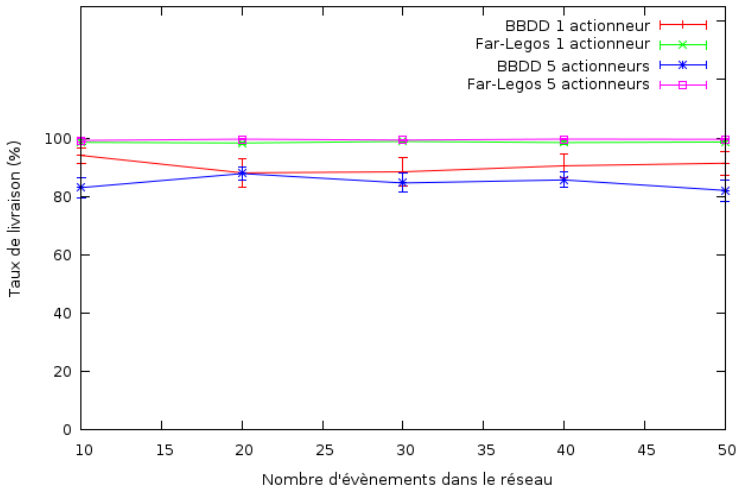


(b) Topologie aléatoire

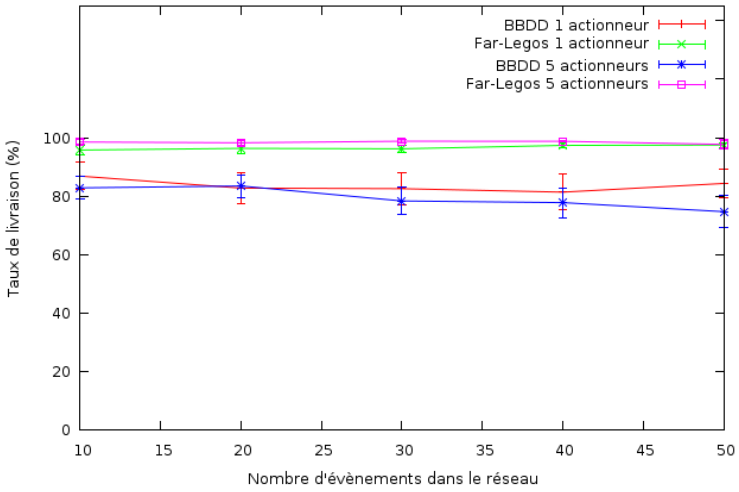
FIGURE 4.12: Délai de bout en bout pour une petite zone non-couverte

s'il peut communiquer directement avec la destination finale. Donc, en envoyant les données à travers l'épine dorsale même dans la zone couverte, BBDD effectue plus de sauts pour atteindre la destination finale par rapport à Far-Legos.

Lorsqu'un message de données a besoin de moins de sauts pour atteindre la destination, cela signifie aussi que nous utilisons moins d'énergie au niveau des nœuds dans le réseau et donc nous augmentons la durée de vie du réseau.



(a) Topologie en grille

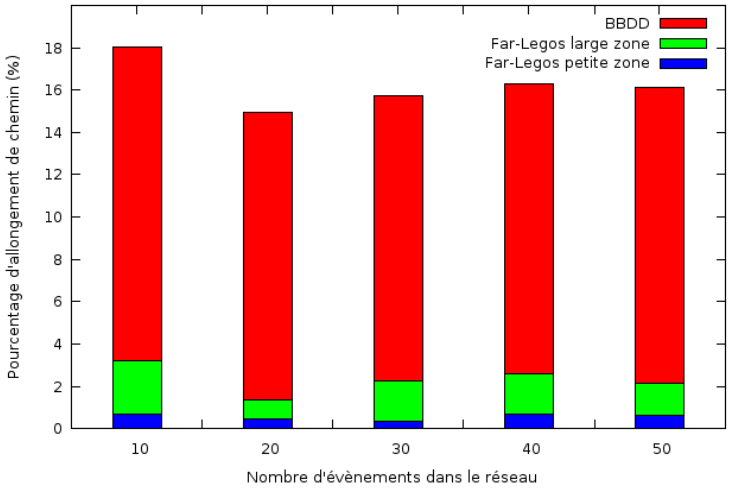


(b) Topologie aléatoire

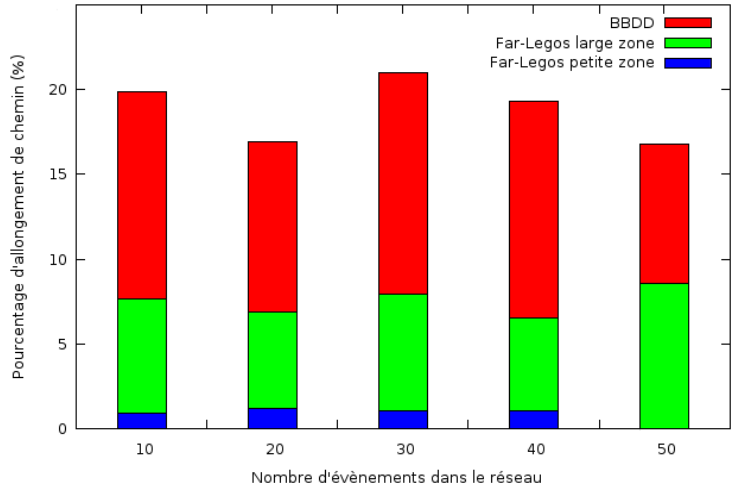
FIGURE 4.13: Taux de livraison pour une large zone non-couverte

4.4 Discussion et optimisation de Far-Legos

Compte tenu de la phase de collecte de données, l'allongement du chemin peut être important en raison de la sélection du prochain saut d'une manière aléatoire dans le cas où le nœud ne trouve pas un voisin ayant un rang plus petit que le sien. En effet, si le processus de l'acheminement de données semble être orienté pour le routage entre les couronnes, appelé inter-couronne, puisque le prochain saut est nécessairement un nœud avec un rang plus petit, le processus d'acheminement de données dans une même couronne, appelé intra-couronne, n'est pas optimisé. En effet, lorsqu'un nœud ne trouve pas un voisin ayant un rang plus petit (présence d'un trou dans le réseau par exemple), le prochain saut choisi sera un nœud appartenant à la même couronne ayant un même rang : cette



(a) Topologie en grille



(b) Topologie aléatoire

FIGURE 4.14: Comparaison de l'allongement du chemin pour Far-Legos et BBDD

sélection est aléatoire. Une vue d'ensemble de ce problème est illustrée dans la figure 4.15.

Supposons qu'il y a 11 nœuds dans le réseau. Comme le montre la figure 4.15, les nœuds A, B, C, D, E, F, G, H, I et J ont la même valeur de rang (N+1) tandis que le rang du nœud K est égal à N, et seulement le nœud J détecte un voisin de la couronne suivante (nœud K, valeur du rang égal à 1). Lorsque le nœud A a un message à envoyer vers le nœud actionneur, il doit choisir le prochain saut parmi ses voisins appartenant à la même couronne. Pour contourner ce minimum local, une transmission intra-couronne est nécessaire. En supposant que le message passe une seule fois par un nœud, le message envoyé par le nœud A nécessite 9 sauts pour atteindre le nœud J qui a un voisin appartenant à la couronne suivante. Dans la section suivante, nous proposons deux optimisations locales afin d'améliorer le comportement des Far-Legos dans ce genre de situation.

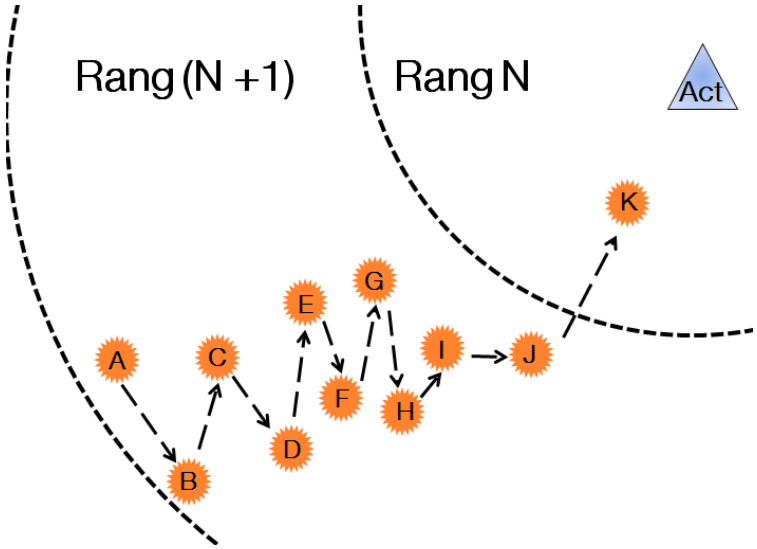


FIGURE 4.15: Collecte de données intra-couronne : le pire cas

4.4.1 Première variante : *Oriented-FAR*

Idée de base de *Oriented-FAR* : Adaptation dynamique de rang

La phase d'affectation de rang dans Far-Legos est un processus statique : la valeur des rangs n'évolue pas en fonction de la situation des nœuds dans la couronne et en fonction de l'évolution du réseau. Les nœuds connaissent leurs rangs lors de la réception des messages d'affectation de rang envoyés par les nœuds actionneurs. L'idée de base de la variante *Oriented-FAR* est d'adapter la valeur du rang en utilisant des informations locales, pour tenir compte de la proximité d'une autre couronne.

Description détaillée *Oriented-FAR*

La variante *Oriented-FAR* est basée sur la proposition de base Far-Legos, le fonctionnement de *Oriented-FAR* peut aussi être divisé en quatre phases que nous avons résumé précédemment : une phase d'affectation de rang, une phase de découverte de voisinage, une phase de construction de la topologie et une phase de collecte de données.

Le but de cette variante est de diminuer l'allongement de chemin lors d'un acheminement intra-couronne en utilisant une affectation dynamique de rang.

Avec *Oriented-FAR*, la deuxième phase (découverte de voisinage) est modifiée pour ajouter une adaptation dynamique des rangs. Les trois autres phases restent inchangées.

La nouvelle phase de découverte de voisinage peut être divisée en trois sous-phases :

- **Découverte des voisins** : Comme il a été décrit précédemment, chaque nœud capteur envoie un message *Hello* pour découvrir ses voisins. Dans ce message *Hello*, chaque nœud ajoute son ID et son rang. En recevant un message *Hello*, chaque capteur met à jour sa table de voisinage.
- **Calcul du nouveau rang** : Après avoir récolté des informations sur son voisinage, chaque capteur calcule un nouveau rang en se basant sur les informations locales recueillies dans la sous-phase précédente. Le nouveau rang sera calculé comme dans l'équation 4.2 ci-dessous. L'idée ici est de permettre au nœud de diminuer son rang quand il est proche de la couronne suivante

et d'augmenter son rang quand il est proche de la couronne précédente. Dans l'équation 4.2, α représente le nombre de capteurs appartenant à la couronne précédente (ayant un rang plus grand que lui-même) et β représente le nombre de capteurs appartenant à la couronne suivante (ayant un rang plus petit). Notez que les valeurs de α et de β sont maintenues à une valeur maximale égale à 9 si jamais le nombre de voisins dépasse cette valeur. Nous fixons cette limite de α et de β pour conserver la décroissance des rangs entre les couronnes en se rapprochant du nœud actionneurs. Le terme $Rang\_Assigned$ représente le rang affecté initialement par le(s) nœud(s) actionneur(s).

– **Annonce du nouveau rang** : Après avoir calculé la valeur de son nouveau rang (comme dans l'équation 4.2), chaque nœud capteur vérifie si le nouveau rang est différent du rang attribué par l'actionneur pendant la première phase. Si le rang a été modifié, le capteur annonce ce nouveau rang dans son voisinage. En recevant cette nouvelle annonce, les nœuds voisins vont mettre à jour la valeur du rang de ce nœud dans leurs tables de voisinage.

$$Nouveau\_Rang = Rang\_Assigned + \alpha * 0,1 - \beta * 0.1 \tag{4.2}$$

Les deux dernières phases (construction de la topologie dans les zones non-couvertes et la phase de collecte de données) ne sont pas modifiées. Au cours de la phase de collecte de données, chaque nœud source envoie ses données vers son voisin ayant le rang le plus petit dans sa table de voisinage. Ainsi, la phase de collecte des données intra-couronne sera orientée : un message de donnée sera envoyé aux nœuds qui ont au moins un voisin avec un plus petit rang c'est-à-dire un nœud plus proche de la destination finale.

Nous reprenons la topologie précédente et nous la représentons dans la figure 4.16 avec Oriented-FAR. Le nœud J met à jour son rang, égale à « 1.9 », puisqu'il détecte un voisin appartenant à la couronne suivante et le diffuse dans son voisinage. Les autres nœuds gardent leurs rangs affectés par le nœud actionneur. Le nœud A ayant des données à envoyer vers l'actionneur, va choisir parmi ses voisins le nœud ayant le plus faible rang qui sera le nœud J. De ce fait, le choix des prochains sauts sera orienté et nous pouvons optimiser ainsi l'acheminement des données intra-couronne. Quand le nœud A a un message de donnée à envoyer vers le nœud actionneur, il va choisir ke nœud J comme prochain saut (voir figure 4.16).

4.4.2 Deuxième variante : *Clustered-FAR*

Idée de base de *Clustered-FAR* : création de *clusters* dans les zones couvertes

Afin de minimiser l'allongement des chemins entre un nœud source et l'actionneur, la sélection du « bon »prochain saut est une question clé. Pour l'acheminement inter-couronnes, nous avons proposé de sélectionner un nœud avec un rang plus petit que le rang du nœud courant, cela signifie que le prochain saut appartient à la couronne suivante. Pour l'acheminement intra-couronne nous devons choisir un nœud qui sera le plus proche possible de la couronne suivante. Intuitivement, le prochain saut sera un nœud ayant le plus grand nombre de nœuds voisins appartenant à la couronne suivante. Nous proposons ainsi de construire des *clusters* au sein de chaque couronne, où les *clusterheads* sont toujours les nœuds qui sont proches de la couronne suivante.

En utilisant ce mécanisme simple et efficace, l'allongement des chemins sera réduit.

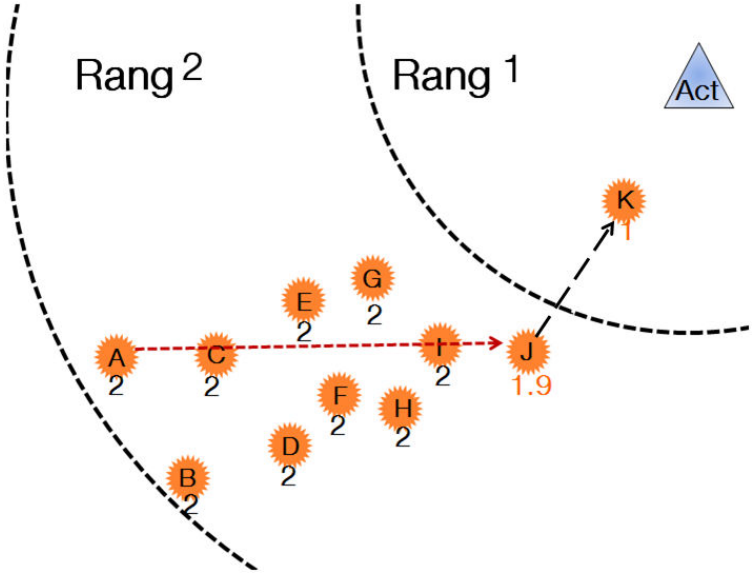


FIGURE 4.16: Exemple d'adaptation dynamique des rangs.

Description détaillée de *Clustered-FAR*

Clustered-FAR, basé sur le protocole Far-Legos, ne nécessite pas de modification dans les deux premières phases (phase d'affectation de rang et la phase de découverte de voisinage).

La troisième phase de construction de topologie ne concerne plus seulement la zone non-couverte. En effet, avec *Clustered-FAR*, après la phase de découverte de voisinage, chaque nœud ayant détecté au moins un voisin appartenant à la couronne suivante calcule un délai d'attente avant de commencer à diffuser un message d'annonce pour devenir un *clusterhead* (CH) dans son voisinage. Ce délai est utilisé pour être sûr que les nœuds plus proches des actionneurs démarrent les premiers la construction des *clusters*. Ce délai est inversement proportionnel au nombre de voisins détectés appartenant à la couronne suivante pour chaque nœud : le nœud détectant plus de voisins appartenant à la couronne suivante calcule un plus petit délai qu'un autre nœud détectant moins de voisins de la couronne suivante. Après expiration de ce délai, chaque nœud diffuse dans son voisinage un message d'annonce de changement de statut pour devenir un CH. Les nœuds, n'ayant que des voisins appartenant à la même couronne, recevant ce message d'annonce de CH seront des nœuds *Membres* attachés à ce CH. Un nœud peut recevoir plusieurs annonces de CH, il conserve la liste de ces CHs en leur affectant des préférences : plus le CH s'annonce en premier, plus le nœud le considère comme son CH préféré auquel il va envoyer ses messages pendant la phase d'acheminement de donnée.

Dans la phase de collecte des données chaque nœud envoie les données vers son CH préféré ou vers un voisin appartenant à la couronne suivante dans la direction de l'actionneur (voir figure 4.17).

4.5 Evaluation de performance des variantes de Far-Legos

4.5.1 Paramètres de simulation

Nous considérons un réseau de capteurs sans fil avec un ensemble de nœuds capteurs et un ensemble de nœuds actionneurs déployés dans une topologie en grille régulière et une topologie en grille aléatoire. Nous supposons que tous les nœuds capteurs peuvent être couverts par les actionneurs

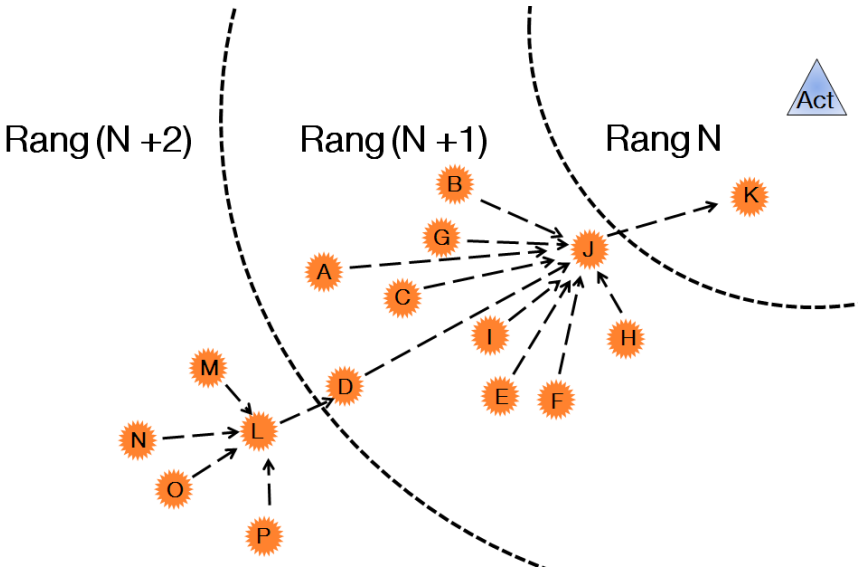


FIGURE 4.17: Exemple de structuration de *Clustered-FAR*

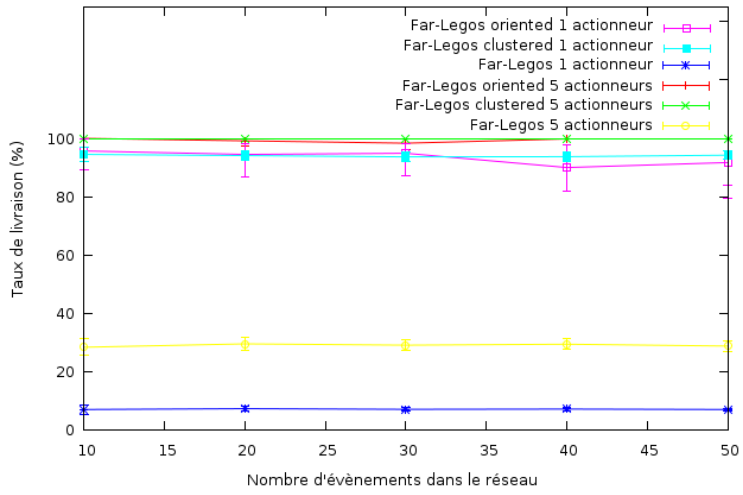
présents dans le réseau. Depuis les modifications apportées sur *Oriented-FAR* et *Clustered-FAR* ne conviennent que pour la zone couverte, nous supposons dans cette section qu'il n'y a pas de zone non-couverte. Pour créer le problème auquel s'adresse les deux variantes *Oriented-FAR* et *Clustered-FAR*, nous supposons que la largeur des couronnes est égale à deux fois la portée de transmission maximale des nœuds capteurs. Les paramètres de simulation sont les mêmes que les simulations précédentes à savoir une couche MAC 802.15.4 et un modèle de propagation à deux rayons. Nous déclenchons un certain nombre d'événements applicatifs aléatoires dans le réseau. Chacun de ces événements sera détecté par un nœud capteur qui le reportera au nœud actionneur le plus approprié. Comme dans les simulations précédentes, nous supposons aussi que chaque nœud capteur choisit l'actionneur avec lequel il a le plus petit rang. Le tableau 4.3 résume les principales caractéristiques du réseau.

Paramètre	Valeur
Topologie	Grille régulière et Grille aléatoire
Nombre de nœuds	121 nœuds
Largeur des couronnes	2 * la portée des capteurs
Nombre d'évènements	10 .. 50
Nombre de voisin / nœud	8 nœuds
Protocole MAC	802.15.4 (CSMA)
Propagation	Modèle à deux rayons
Nombre de sauts maximum autorisé	16 sauts
Intervalle de confiance	95%
Simulateur	WSNet [67]

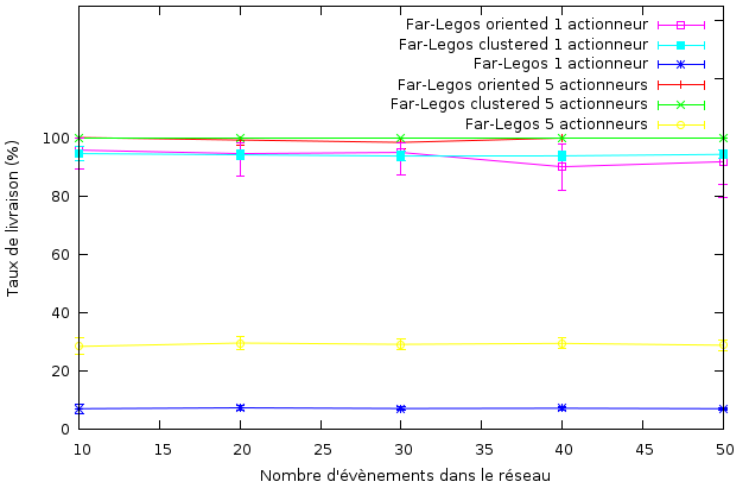
TABLE 4.3: Paramètres de simulation communs pour *Oriented-FAR* et *Clustered-FAR*

4.5.2 Evaluation du taux de livraison

Dans la figure 4.18, nous représentons le taux de livraison lorsque le nombre d'évènements détectés dans le réseau augmente pour les deux topologies (grille régulière et grille aléatoire). Notez que le taux de livraison de la proposition Far-Legos est faible, tandis que les taux de livraison des deux variantes *Oriented-FAR* et *Clustered-FAR* sont proches de 100%. Cela est dû à la transmission aléatoire dans la même couronne quand un nœud ne trouve pas un voisin ayant un rang plus petit que le sien comme prochain saut pour Far-Legos : ainsi les boucles ne peuvent pas être évitées et les messages de données seront abandonnés une fois le nombre de sauts maximal est atteint. Les deux variantes permettent d'éviter ce problème de boucles et assurent la livraison des messages de données tout en évitant l'acheminement aléatoire intra-couronne.



(a) Topologie en grille



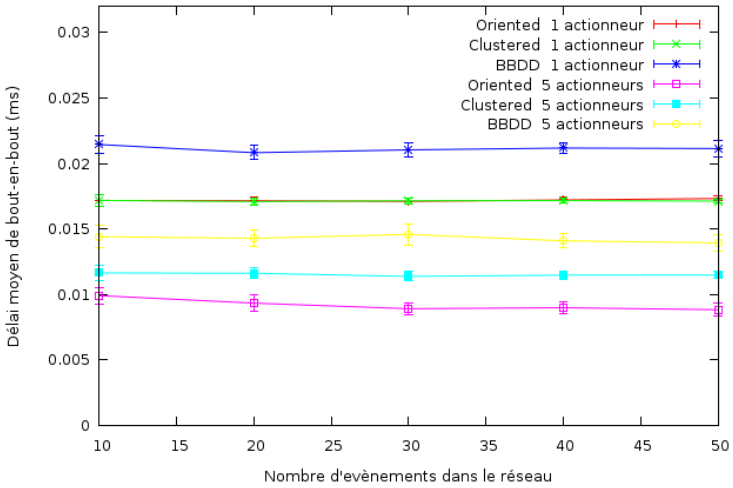
(b) Topologie aléatoire

FIGURE 4.18: Taux de livraison Far-Legos, *Clustered-FAR* et *Oriented-FAR*

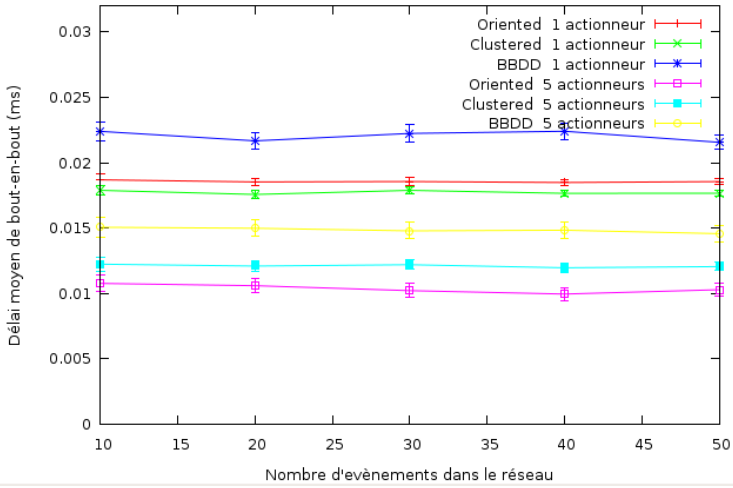
4.5.3 Evaluation du délai

Nous comparons le délai de bout-en-bout des deux variantes *Oriented-FAR* et *Clustered-FAR* avec le délai de bout-en-bout avec BBDD. La figure 4.19 représente le délai moyen de bout en bout lorsque le nombre d'événement augmente dans le réseau : les deux variantes offrent un délai inférieur à celui de BBDD. Ceci est dû au fait qu'avec BBDD, un message de donnée effectue plus de saut avant d'atteindre la destination finale.

Les délais de bout-en-bout des deux variantes *Oriented-FAR* et *Clustered-FAR* sont proches avec un meilleur délai offert par *Oriented-FAR*. En effet, avec *Clustered-FAR* un message de donnée quand il voyage au sein d'une même couronne, il passe par les nœuds *clusterheads*, ce qui augmente le nombre de saut et par la suite le délai de bout-en-bout.



(a) Topologie en grille



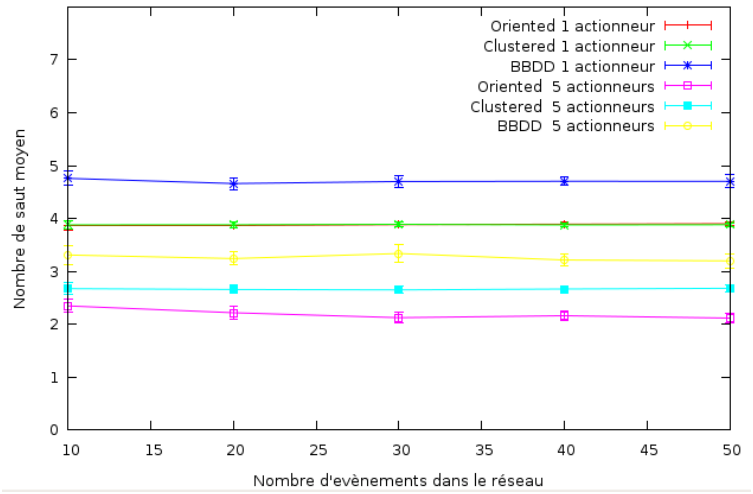
(b) Topologie aléatoire

FIGURE 4.19: Délai de bout en bout *Clustered-FAR* et *Oriented-FAR*

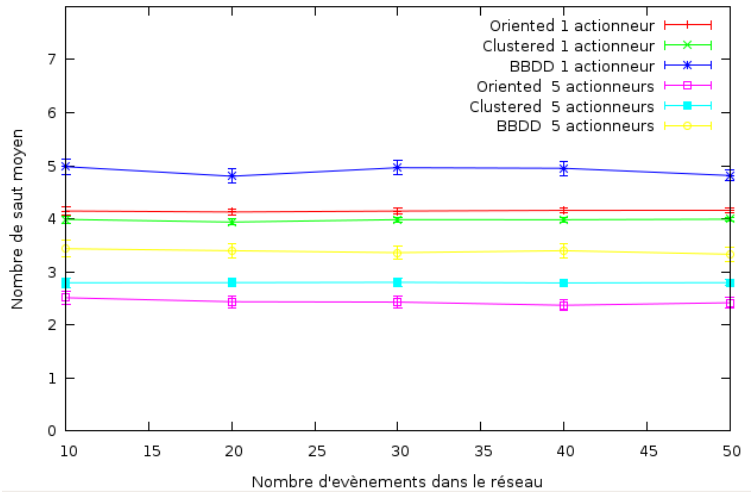
4.5.4 Evaluation du nombre de saut

Dans cette section, nous comparons le nombre de sauts pour les deux variantes *Oriented-FAR* et *Clustered-FAR*. La figure 4.20 représente le nombre moyen de sauts en fonction des événements détéctés dans le réseau.

Une fois de plus, les deux variantes présentent un comportement meilleur que BBDD et proche l'un de l'autre. Le prochain saut choisit avec *Oriented-FAR* et *Clustered-FAR* est le voisin le plus proche de la couronne suivante : le nœud ayant le plus petit rang (pour *Oriented-FAR*) et le nœud déclaré comme *clusterhead* (pour *Clustered-FAR*).



(a) Topologie en grille

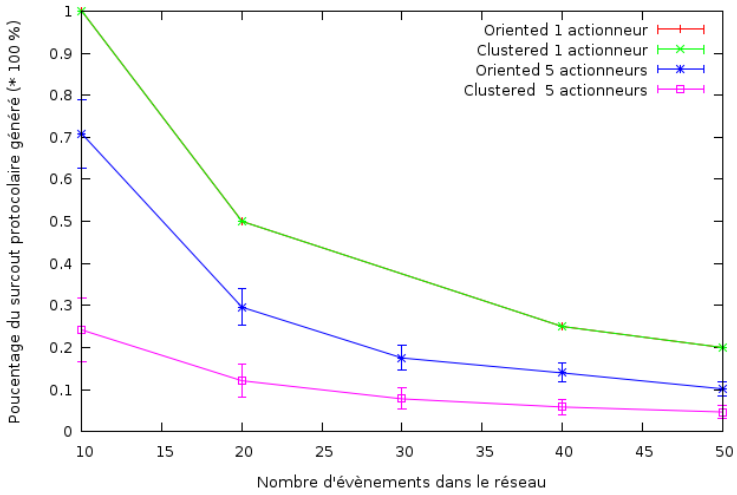


(b) Topologie aléatoire

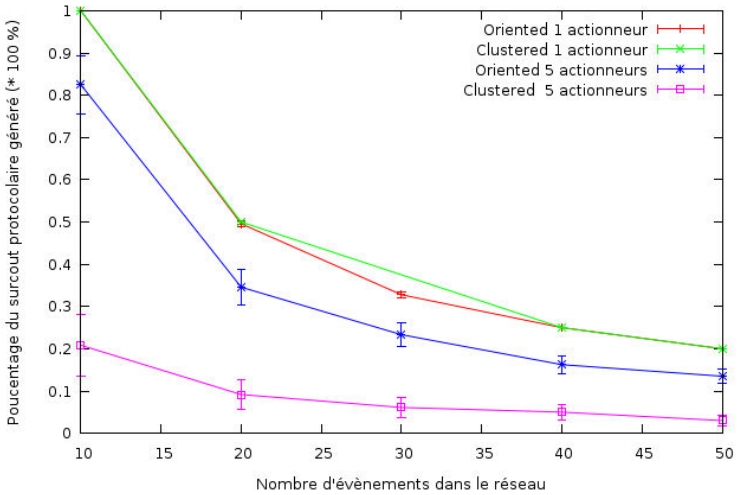
FIGURE 4.20: Nombre de saut moyen pour *Clustered-FAR* et *Oriented-FAR*

4.5.5 Evaluation du surcoût protocolaire

Dans cette section, nous comparons le nombre de messages de contrôle utilisé par les deux variantes *Oriented-FAR* et *Clustered-FAR* (voir figure 4.21). Comme il est illustré dans la figure 4.21, le nombre de messages de contrôle transmis par les deux variantes de Far-Legos diminue en fonction du nombre d'événements dans le réseau. En effet, ces messages de contrôle sont utilisés qu'une seule fois un nœud a changé son rang (pour *Oriented-FAR*) ou est devenu un *clusterhead* (pour *Clustered-FAR*). Dans la figure 4.21, le nombre de message de contrôle utilisé par *Clustered-FAR* est inférieur à celui de *Oriented-FAR*, parce que dans *Clustered-FAR* seulement les nœuds qui deviennent des *clusterheads* envoient un message de notification, alors qu'avec *Oriented-FAR*, chaque nœud dans le réseau envoie un nouveau message après la phase de calcul de son nouveau rang.



(a) Topologie en grille



(b) Topologie aléatoire

FIGURE 4.21: Surcoût protocolaire de *Clustered-FAR* et *Oriented-FAR*

4.6 Conclusion

Nous avons proposé dans ce chapitre un protocole d'auto-organisation et de collecte de données, Far-Legos, approprié pour un réseau hétérogène de capteurs et d'actionneur.

L'objectif de notre proposition est d'exploiter l'hétérogénéité au niveau des nœuds actionneurs. Notre proposition a pour but préserver de l'énergie au niveau des nœuds capteurs, de minimiser le délai de bout-en-bout et d'offrir un taux de livraison élevé tout en tirant parti des ressources disponibles au niveau des nœuds actionneurs. Nous avons montré par simulation que notre proposition offre un délai de bout-en-bout faible et un taux de livraison élevé dans les petites et larges zones non-couvertes dans un réseau stable.

Far-Legos permet de profiter de la puissance d'émission des nœuds actionneur pour apporter une information de gradient au niveau des nœuds capteurs. Cette information de gradient, appelé rang, permet de diminuer l'allongement du chemin dans les zones couvertes. En effet, nous avons montré par simulation que l'allongement de chemin par rapport à l'algorithme du plus court chemin de notre proposition ne dépasse pas les 8% sur des topologies en grille régulière et aléatoire par rapport à un algorithme du plus court chemin.

Ensuite, nous avons présenté deux variantes de notre proposition pour répondre au problème de la transmission aléatoire dans la phase de collecte des données : *Clustered-FAR* et *Oriented-FAR*. Les deux variantes présentent des performances améliorées. Les deux variantes conduisent à des performances proches en termes de taux de livraison et le nombre de saut.

Cependant, d'une part, *Clustered-FAR* a un surcoût protocolaire inférieur à celui de la variante *Oriented-FAR*. En effet, avec *Clustered-FAR* seulement les nœuds qui se déclarent en tant que *clusterhead* envoient un message de contrôle en plus. Alors que pour *Oriented-FAR*, tous les nœuds diffusent la nouvelle valeur de leurs rangs.

D'autre part, *Oriented-FAR* offre un délai plus faible que celui offert par *Clustered-FAR*. En effet, l'adaptation dynamique de tous les rangs des nœuds du réseau permet de choisir le meilleur prochain saut. Ce qui permet d'optimiser la longueur du chemin et par la suite le délai que met un message de donnée pour atteindre la destination finale.

Dans le chapitre suivant, nous nous intéressons aussi à l'hétérogénéité au niveau des liens radio constituant le réseau. Comme l'objectif de ce chapitre, le travail suivant a pour objectif de tirer profit de l'hétérogénéité au niveau des liens pour assurer une collecte de données efficace. En d'autres termes, nous nous intéressons aux liens asymétriques et nous proposons un protocole de routage exploitant ce type de liens.

Collecte de données dans un réseau hétérogène

5

Sommaire

5.1 Introduction

Après avoir introduit dans le chapitre précédent une stratégie d'auto-organisation permettant de tirer profit de l'hétérogénéité dans un réseau de capteurs et actionneurs, nous traiterons dans ce chapitre la problématique de la collecte de données dans un réseau hétérogène. En effet, la présence de différents types de nœuds avec différentes portées de transmission peut provoquer l'apparition des liens asymétriques. Ces derniers peuvent détériorer les performances des protocoles de routage. Nous proposons dans ce chapitre, une adaptation du protocole de collecte de données Legos dans un contexte hétérogène. Nous nous intéressons aussi au protocole de routage RPL [155] proposé comme standard par le groupe de travail ROLL [1] (*Routing Over Low power and Lossy Networks*) à l'IETF [2] (*Internet Engineering Task Force*).

En effet, le protocole RPL construit et maintient une topologie logique sous forme d'un graphe acyclique orienté DAG (*Directed Acyclic Graph*) ayant comme racine un ou plusieurs puits ou actionneurs. Les données transmises par les nœuds du réseau ne seront transmises que sur les liens du DAG. RPL assure les trois types de trafics de données à savoir le trafic multi-points à point (MP2P), point à multi-points (P2MP) et le trafic point à point (P2P). Toutefois, le trafic MP2P est le trafic dominant dans les réseaux de capteurs et actionneurs appelés aussi LLNs (*Low power and Lossy Networks*).

Dans ce chapitre, nous commençerons par donner une description plus détaillée du protocole RPL. Ensuite nous analyserons ce dernier dans un contexte hétérogène pour introduire une nouvelle métrique de calcul de rang. Celle-ci sera utile pour détecter et éviter les liens asymétriques au niveau de la couche réseau. Nous présenterons par la suite la description d'une adaptation du protocole Legos pour tenir compte aussi des liens asymétriques. Ce chapitre s'achèvera par la présentation des résultats de performance de l'adaptation de RPL et de l'adaptation de Legos dans un contexte hétérogène.

5.2 Un aperçu sur le mode de fonctionnement de RPL

Le protocole RPL [155] est un protocole de routage à vecteur de distance pour les LLNs qui décrit une méthode de construction d'une topologie logique appelée DODAG (*Destination Oriented Directed Acyclic Graph*) utilisant une fonction d'objectif [145] et un ensemble de métriques et de contraintes [149]. La fonction d'objectif se base sur une combinaison de métriques et de contraintes pour calculer le « meilleur »chemin acceptable. Il pourrait y avoir plusieurs fonctions d'objectifs considérées par le même nœud ou le même réseau. Par exemple, plusieurs DODAGs peuvent être utilisés avec l'objectif de (1) Trouver les chemins avec les meilleurs valeurs de ETX [46] (métriques) et d'éviter les liens non-cryptés (contrainte) ou (2) Trouver le chemin offrant la plus faible latence (métrique) tout en évitant les nœuds fonctionnant sur batterie (contrainte). La fonction d'objectif ne doit pas nécessairement préciser les métriques/contraintes, mais elle dicte des règles qui cadrent la formation du DODAG (par exemple, le nombre de parents secondaires, l'utilisation de l'équilibrage de charge, etc...).

Le graphe construit par RPL est une topologie logique construite sur une topologie physique pour répondre à des critères spécifiques. Un réseau peut avoir plusieurs topologies de routage actives en même temps. Ces topologies sont utilisées pour transporter différents trafics avec des métriques et des contraintes différentes.

1. ROLL IETF Working Group : http ://datatracker.ietf.org/wg/roll/charter/
2. Internet Engineering Task Force : http ://www.ietf.org

5.2.1 Construction du DODAG

Le processus de construction de la structure DODAG commence à la racine ou LBR (*LoWPAN Border Router*) qui est généralement le nœud de collecte de données (le puits ou un actionneur). Il pourrait y avoir des racines multiples configurées dans le réseau. Le protocole de routage RPL spécifie un ensemble de nouveaux messages de contrôle ICMPv6 pour échanger des informations liées à la construction de la structure DODAG.

La racine commence la diffusion des informations concernant la structure en utilisant le message DIO (*DODAG Information Object*). Les nœuds à portée de communication de la racine recevront et traiteront ce message DIO, puis ils rendront une décision (joindre la structure ou pas) fondée sur certaines règles (selon la fonction d'objectif, les caractéristiques du DAG et le coût du chemin annoncé). Une fois que le nœud s'est joint à la structure, il a une route vers la racine de la structure DODAG (voir figure 5.1(a)).

La racine du DODAG est appelée le parent du nœud. Le nœud calcule son rang dans le graphe, qui représente la position du nœud dans la structure DODAG. Si ce nœud est configuré pour agir comme un routeur dans le réseau (ce qui est généralement vrai dans les WSNs et WSANs), il commence à diffuser à son tour dans son voisinage les nouvelles informations de la structure qu'il vient de rejoindre à travers des messages DIOs. Si le nœud n'est pas configuré pour être un routeur alors il rejoint tout simplement la structure DODAG et n'envoie pas de message DIO. Les nœuds voisins recevant cette annonce vont répéter ce processus de sélection de parent, d'ajout d'itinéraire et d'annonce des nouvelles informations concernant la structure DODAG à l'aide des messages DIOs (voir figures 5.1(b) et 5.1(c)). Ce processus continue jusqu'à couvrir tous les nœuds du réseau. Chaque nœud de la structure DODAG a une entrée de routage vers son parent (ou plusieurs parents selon la fonction d'objectif) à travers lequel ce nœud peut atteindre la racine de la structure DODAG.

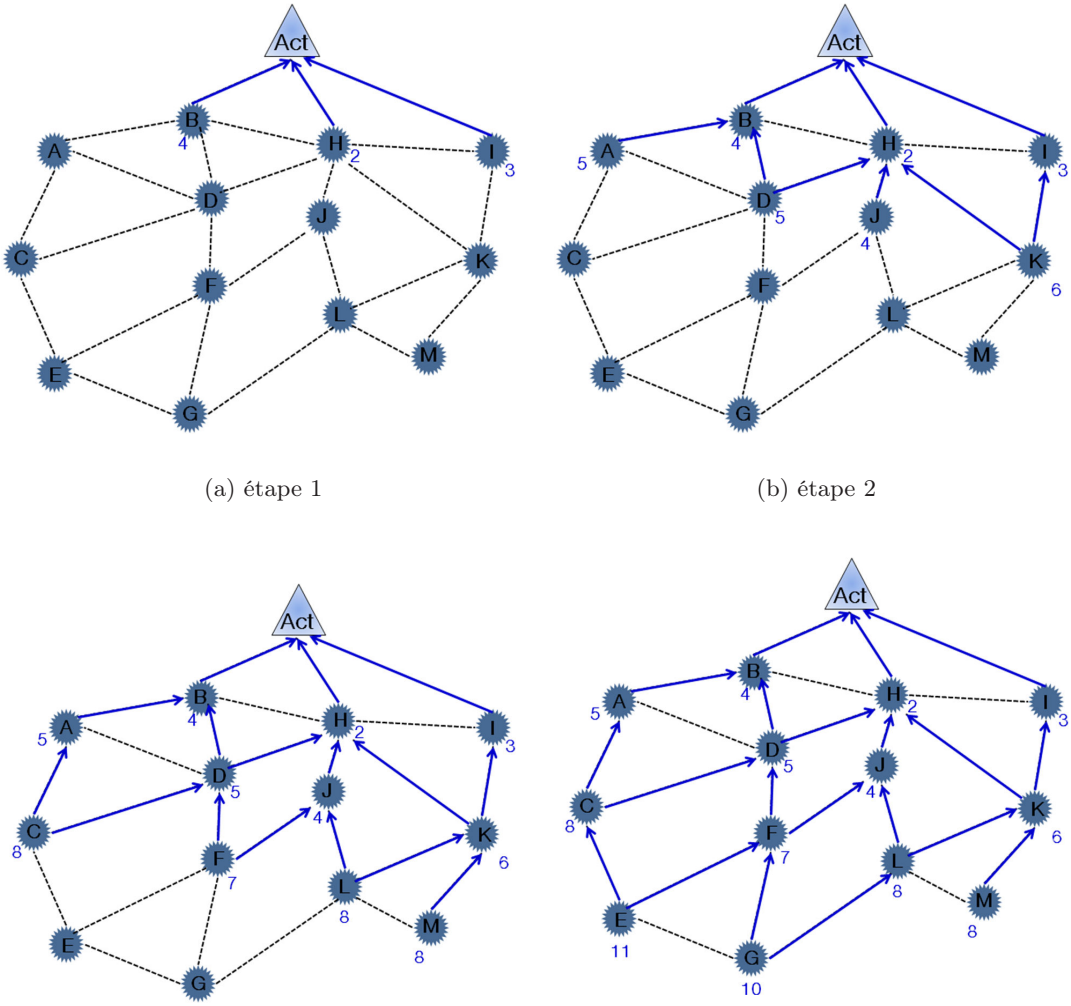
Chaque nœud dans le graphe a un rang qui représente la position relative de ce nœud par rapport à la racine de la structure DODAG (voir figure 5.1(d)). La notion de rang est utilisée par RPL à des fins diverses, y compris l'évitement les boucles. Les différentes étapes du processus de construction graphique sont représentées dans la figure 5.1.

Les messages DAO (*Destination Advertisement Object*) visent à maintenir les routes descendantes et ne sont utilisés que pour des applications nécessitant des trafics de type point à multi-point et point à point.

5.2.2 Trafics supportés par le DODAG

Après la construction de la structure logique en DODAG, quand un nœud a des données à envoyer vers la racine, il les envoie vers un de ses parents (appelé son parent préféré). Ces données vont remonter la structure jusqu'à atteindre la destination finale. Ce modèle représente le modèle de trafic MP2P (multi-points à point). Nous appelons ce type de routage dans ce chapitre trafic convergecast : les messages circulent des nœuds feuilles au(x) nœud(s) racine(s). D'autres applications nécessitent la présence d'un trafic dans le sens opposé. Ce trafic, P2MP (point à multi-points), écoule les informations vers les nœuds feuilles. Ce trafic peut provenir de l'extérieur du réseau, à partir de(s) nœud(s) racine(s), nous appelons ce type de trafic dans ce chapitre trafic divergecast.

Tout cela nécessite une table de routage qui doit être construite au niveau de chaque nœud et un mécanisme pour remplir ces routes. Ceci est accompli par le message DAO (*Destination Advertisement Object*). Les messages DAOs sont utilisés pour annoncer l'accessibilité vers les nœuds qui peuvent être des destinations potentielles. Un nœud appartenant à la structure DODAG enverra



(a) étape 1

(b) étape 2

(c) étape 3

(d) étape 4

FIGURE 5.1: Exemple illustrant la construction d'un DODAG (Les messages DIOs ne sont pas représentés)

un message DAO à son ensemble de parents. A la réception de ce message DAO, un nœud parent ajoute une entrée dans la table de routage et il envoie à son tour un message DAO à son ensemble de parents (des agrégations des informations reçues peuvent être envisagées). Ce processus se poursuit jusqu'à ce que l'information atteigne la racine du DODAG. Il est à noter que ce mode est appelé le « mode de fonctionnement avec stockage »où tous les nœuds intermédiaires disposent d'une mémoire disponible pour stocker les tables de routage. Le protocole RPL soutient également un autre mode appelé « mode de fonctionnement sans stockage »où aucun nœud intermédiaire ne stocke les routes vers les nœuds qui viennent de s'annoncer avec des DAOs, le nœud puits utilisent alors un routage par la source

Le protocole RPL prend également en charge le trafic de type point à point (P2P) (trafic à partir de n'importe quel nœud vers un autre nœud dans la structure DODAG). Quand un nœud envoie un message vers un autre nœud dans la structure DODAG, ce message voyage en direction de la racine du DODAG jusqu'à atteindre un nœud ancêtre commun, ayant une connaissance de la route, au niveau duquel le message sera transmis en direction de la destination finale. Une technique pour optimiser ce type de trafic est à l'étude par le groupe du travail ROLL dans [63].

5.2.3 Détection et évitement des boucles

Un des aspects important pour un protocole de routage comme RPL est la détection et l'évitement des boucles. Dans les réseaux traditionnels des boucles temporaires sont formées en raison de changements de topologie et de manque de synchronisation entre les nœuds. Ces boucles doivent être détectées le plus rapidement possible pour éviter les pertes de paquets (en raison de l'expiration du TTL, ou d'atteinte du nombre de sauts maximal autorisé) et la congestion des liens. Ainsi différents mécanismes d'optimisation ont été proposés et mis en place pour éviter de telles boucles.

En effet, il est recommandé de réagir d'une manière contrôlée pour supprimer la condition qui a créé ces boucles et les éliminer. La réaction doit être contrôlée et non excessive pour ne pas mener à d'autres oscillations de routage et de consommation d'énergie. Ainsi, RPL ne garantit pas l'absence de boucles, mais tente plutôt de les éviter et propose des mécanismes pour les détecter avec des techniques de validation de chemin de données. A cet effet, RPL spécifie deux règles pour éviter les boucles. Ces règles se basent sur la propriété du « rang »des nœuds.

- Tout d'abord, un nœud n'est pas autorisé à sélectionner en tant que parent un nœud voisin qui est plus profond que lui (c'est à dire dont le rang de ce voisin est supérieur à celui du nœud).
- Deuxièmement, un nœud n'est pas autorisé à tenter de descendre dans la structure DODAG (en augmentant son rang) pour augmenter le nombre de parents.

Les boucles dans les LLNs sont inévitables, il y a donc un besoin de les détecter et les éviter. Une façon d'y parvenir est d'utiliser un bit indicateur dans l'entête RPL pour valider le chemin de données. Par exemple, quand un nœud envoie un paquet destiné à un de ses enfants, il active ce bit, en le mettant à 1 par exemple, et transmet le paquet au prochain saut. Après avoir reçu un paquet avec le bit activé, si la table de routage indique que la source est considérée comme un descendant du nœud actuel, cela indique une incohérence ou une boucle. Le paquet doit être supprimé et une réparation locale doit être déclenchée.

5.2.4 Réparation globale et locale

La réparation est un élément clé pour tous les protocoles de routage et se réfère à la capacité de réparer la topologie logique en cas de défaillance. Le protocole RPL propose des mécanismes de réparation dans le cas des ruptures de liens et/ou de défaillances de nœuds. Il spécifie deux

techniques qui sont complémentaires : la réparation locale et la réparation globale. Quand une défaillance d'un lien ou d'un nœud voisin est détectée et que le nœud n'a pas d'autre chemin vers la racine du DODAG, une réparation locale est déclenchée pour trouver rapidement un autre parent à un saut ou un nouveau chemin permettant la connexion avec le nœud racine. Il s'agit d'une réparation dite locale avec peu d'incidence globale sur l'ensemble de la structure. Cette réparation locale est gérée en utilisant les messages DIOs (voir [155] pour plus de détail).

Comme les réparations locales ont lieu sur la topologie initiale, cette dernière peut commencer à s'écarter de sa forme optimale, à quel point il peut être nécessaire de reconstruire toute la structure DODAG de nouveau, c'est ce que RPL considère comme la réparation globale. Cette dernière est un mécanisme de réparation qui reconstruit toute la topologie. Il s'agit d'une technique d'optimisation, mais ayant un coût non négligeable. La réparation globale ne peut être déclenchée qu'à partir de la racine du DODAG. RPL propose des réparations globales périodiques ou événementielles.

5.2.5 Utilisation d'une période d'émission adaptative

Pour les LLNs, lorsque le réseau est composé de nœuds qui doivent économiser de l'énergie, il est impératif de limiter le trafic de contrôle. La plupart des protocoles de routage échangent périodiquement des informations locales pour maintenir les tables de routage à jour. Cet échange périodique serait coûteux pour les LLNs où les nœuds sont contraints en ressources. Le protocole RPL utilise un mécanisme de *timer* adaptatif appelé « Trickle »(goutte à goutte) [89]. Ce mécanisme permet de contrôler le débit d'émission des messages DIOs.

Trickle double l'intervalle séparant deux émissions successives de DIOs à chaque fois que le réseau est stable, et ce jusqu'à une valeur maximale. Cela se traduit par une diminution du nombre de messages de contrôle dans le réseau. Dès que des incohérences sont détectées, Trickle réinitialise la période des émissions des messages DIOs. L'utilisation de ce mécanisme d'adaptation de la fréquence des messages DIOs dépend de la stabilité du réseau. En d'autres termes, quand le réseau est stable, le nombre de messages DIOs diminue. Quand une incohérence est détectée (par exemple, une boucle ou un changement dans les paramètres du DODAG) les périodes sont remises à leur valeur initiale pour résoudre rapidement cette incohérence. Parmis les principaux avantages de l'utilisation du *timer* Trickle est qu'il permet de diminuer la congestion et permet aussi d'économiser l'énergie au niveau des nœuds à faibles ressources. De plus, Trickle ne nécessite pas de code complexe et il est assez facile à mettre en œuvre.

5.3 Analyse de RPL dans un contexte hétérogène et dynamique

Le protocole RPL suppose que les liens sont symétriques. Ainsi, quand un nœud reçoit un message DIO de la part d'un nœud source, le protocole RPL suppose que le nœud en réception peut répondre à ce DIO reçu. Or, la présence des liens asymétriques est inévitable dans plusieurs cas de figures [58] [82] [132] [165].

D'abord, la présence d'une hétérogénéité au niveau des nœuds dans le réseau peut provoquer la présence des liens asymétriques comme nous l'avons discuté dans le chapitre précédent. En effet, en présence de plusieurs niveaux de puissance de transmission dans le réseau, la probabilité de l'existence des liens asymétriques augmente, et les protocoles de routage ne prenant pas en compte ce type de liens deviennent inefficaces.

Ensuite, les liens asymétriques peuvent être causés par le déploiement réel du réseau. En effet, prenons l'exemple d'un réseau urbain déployé dans une ville pour la gestion des places de parking.

Considérons deux nœuds identiques, l'un déployé au dessus d'un lampadaire au niveau d'une intersection et l'autre déployé sur le trottoir. Les deux nœuds n'auront pas la même zone de couverture ou zone de transmission. En effet, le premier nœud aura une zone de transmission plus grande que celle du second nœud. Ce déploiement peut ainsi causer la présence des liens asymétriques dans le réseau.

Enfin, même si on considère que les nœuds sont homogènes au niveau des portées de transmission et qu'ils sont tous déployés sur un même plan du réseau, les liens asymétriques peuvent être causés dans ce cas là par les interférences présentes dans le réseau. En effet, lorsqu'il y a un nœud qui génère un trafic continu, ce nœud va créer des liens asymétriques entre ses voisins directs et leurs voisins respectifs : les voisins à un saut de ce nœud générant ce trafic interférant peuvent réussir à envoyer des messages vers leurs voisins mais ne peuvent pas recevoir correctement les messages de ces voisins.

5.3.1 Défaillance du protocole RPL en présence de l'hétérogénéité

Nous nous intéressons ici aux réseaux hétérogènes où les nœuds ont différents niveaux de puissance de transmission. Comme nous l'avons mentionné ci-dessus, la présence d'une telle hétérogénéité provoque la présence des liens asymétriques. Or, la construction du DODAG avec RPL se base sur l'hypothèse que les liens sont symétriques. En effet, RPL suppose que la qualité et la nature des liens sont prises en compte au niveau de la couche 2. Ainsi, RPL suppose que les liens asymétriques sont supprimés au niveau de cette couche. Dans ce chapitre, nous proposons un mécanisme qui permet d'éviter les liens asymétriques au niveau de la couche routage. Comme pour les autres protocoles de routage qui ne sont pas performants en présence des liens asymétriques [166], les performances de RPL peuvent être dégradées quand les liens asymétriques ne sont pas évités.

Prenons l'exemple simple de la figure 5.2, où le nœud actionneur Act est la racine pour construire un DODAG. Cet actionneur ainsi que le nœud B possèdent un niveau de puissance de transmission double de celle des nœuds A et C. L'actionneur commence par diffuser des DIOs qui seront reçus par les trois nœuds A, B et C. Chacun calcule son rang et s'attache à cet actionneur puisqu'ils n'appartiennent pas encore à un DAG. Notons que le nœud C déclare que cet actionneur est son parent préféré alors que le lien entre eux deux est un lien asymétrique. Après les diffusions des nœuds A et B, le nœud C garde l'actionneur en tant que parent préféré puisque ce dernier peut offrir un meilleur rang. Le nœud C considère les nœuds A et B comme étant des parents secondaires. Quand il y a des données à envoyer, le nœud C choisit son parent préféré auquel il va transférer ses messages à savoir l'actionneur. Les messages envoyés par le nœud C ne peuvent pas arriver directement à l'actionneur. Même si l'exemple de la figure 5.2 est un cas particulier, cette situation peut être généralisée et ainsi la fiabilité de livraison de RPL pourrait se voir dégrader en présence des liens asymétriques.

5.3.2 Métrique de calcul de rang pour éviter les liens asymétriques au niveau de la couche réseau

Notre idée est de proposer une fonction d'objectif qui permettra de calculer les rangs des nœuds tout en prenant en compte l'hétérogénéité dans le réseau.

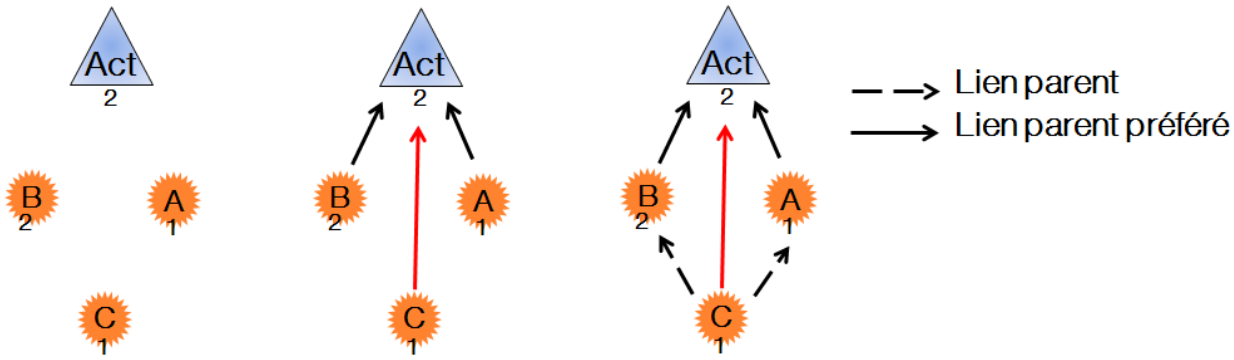


FIGURE 5.2: Défaillance de RPL en présence des liens asymétriques

Hypothèses

Nous supposons que chaque nœud connait son niveau de puissance de transmission. De plus, chaque nœud garde en mémoire le niveau de puissance de transmission de son parent préféré. Chaque nœud envoie dans les DIOs son niveau de puissance de transmission ainsi que l'identifiant de son parent préféré. D'une manière générale, le nœud ajoute l'adresse globale de son parent préféré dans le message DIO (voir la figure 5.3(b)). Les deux figures 5.3(a) et 5.3(b) représentent la structure des messages DIOs de base et celle des messages DIOs adaptés, respectivement.

Nous proposons d'utiliser le champ « *Reserved* »de la figure 5.3(a) pour que chaque nœud annonce son niveau de puissance de transmission (le champ « *Range* »dans la figure 5.3(b)). Pour l'annonce du parent préféré, nous proposons d'ajouter l'adresse unique du parent préféré dans le champ option.

(a) Format d'un message DIO de base [155]

(b) Format d'un message DIO adapté

FIGURE 5.3: Format des messages DIOs utilisés

Calcul de rang

L'idée derrière le calcul du nouveau rang est de rapprocher les nœuds qui ont des niveaux de puissance de transmission proches et d'éloigner les nœuds qui ont des niveaux de puissance de transmission différents. L'objectif est d'attribuer :

– Un rang proche de celui du nœud source quand les deux nœuds ont des niveaux de puissance de transmission proches et quand le niveau du nœud source est inférieure à celui du nœud en

réception.
- Un rang éloigné de celui du nœud source quand les deux nœuds ont des niveaux de puissance de transmission éloignés et quand le niveau du nœud en réception est inférieur à celui du nœud source.

Un exemple simplifié de calcul de rang est représenté dans la formule 5.1. Il est à noter qu'ici nous représentons une formule de base pour le calcul du rang. Cette métrique peut être couplée avec d'autres métriques telles que la qualité du lien.

$$\begin{aligned} Rang = & \text{Rang\_src} \\ & + |\text{niveau\_transmission\_src} - \text{niveau\_transmission\_nœud}| \\ & + \text{niveau\_transmission\_src/niveau\_transmlission\_nœud} \end{aligned} \tag{5.1}$$

Le rang du nœud en réception sera calculé en ajoutant deux composantes au rang du nœud source : La première représente la différence des niveaux de puissance de transmission du nœud en réception et du nœud source du DIO. La deuxième composante représente le ratio du niveau de puissance du nœud source sur le niveau de puissance du nœud en réception.

5.3.3 Description de l'algorithme : A-RPL

Pour éviter les liens asymétriques et la dégradation des performances du protocole RPL en présence de plusieurs niveaux de puissance de transmission, nous proposons de regrouper (en affectant des rangs proches) les nœuds qui ont des niveaux de puissance de transmission proches et d'éloigner (en affectant des rangs espacés) les nœuds qui ont des niveaux de puissance de transmission différents.

Quand un nœud, n'ayant pas de rang, reçoit un DIO, il calcule son rang en se basant sur la formule 5.1 décrite ci-dessus. Ce nœud déclare la source de ce DIO comme étant son parent préféré. Dans le cas où un nœud, appartenant déjà à un DAG, reçoit un DIO, il commence par vérifier si la source de ce DIO est son parent préféré ou pas. S'il s'agit de son parent préféré et que ce parent a changé de rang alors ce nœud calcule son nouveau rang et réinitialise son compteur de diffusion des messages DIOs (comme décrit dans le [155]). Dans le cas où ce parent préféré n'a pas changé de rang alors le nœud n'exécute aucune action (comme dans le [155]). Si la source du DIO n'est pas le parent préféré, le nœen réception commence toujours par vérifier s'il n'est pas le parent préféré de ce nœud source (puisque ce dernier envoie dans le message DIO l'identifiant de son parent préféré). Cette phase de vérification a pour but d'éviter les effets d'instabilités et de changements fréquents de parents préférés. Si le nœud en réception est le parent préféré du nœud qui a envoyé le DIO, alors le nœud ignore ce message. Dans le cas contraire, le nœud compare son niveau de puissance de transmission avec celui du nœud source. Trois cas de figure se présentent :

1. **Le niveau de puissance de transmission du nœud en réception est égal à celui du nœud source du DIO : le lien est symétrique** (voir figure 5.4)
   Quand le nœud en réception détecte que son niveau de puissance de transmission est égal à celui du nœud source, il commence par comparer son niveau de puissance de transmission avec celui de son parent préféré actuel. Si ce dernier a un niveau de puissance de transmission strictement supérieur à celui du nœud en réception, alors ce dernier change de rang et déclare le nœud source en tant que son nouveau parent préféré. L'objectif de ce changement

de parent préféré est de rapprocher les nœuds ayant les niveaux de puissance de transmission proches. Avec ce changement, le nœud en réception s'attachera à un voisin ayant un niveau de puissance de transmission proche du sien.

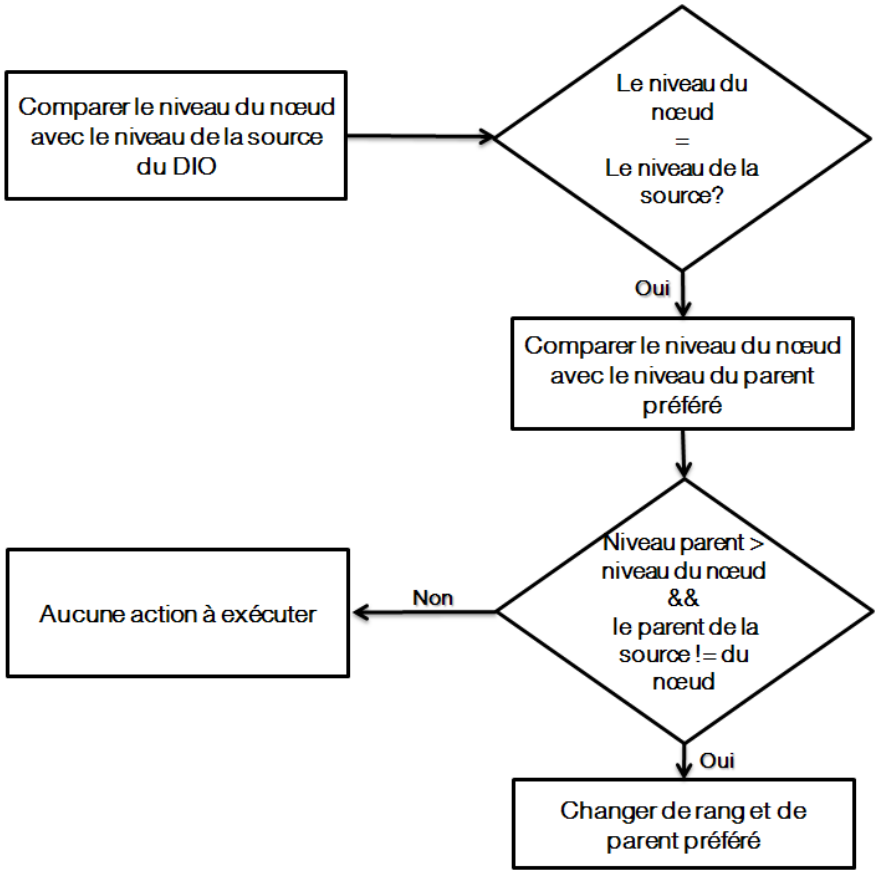


FIGURE 5.4: Principe de calcul de rang : le niveau du récepteur est égal à celui de la source

2. **Le niveau de puissance de transmission du nœud en réception est supérieur à celui du nœud source du DIO : le lien est symétrique** (voir figure 5.5)
   Quand le nœud en réception détecte que son niveau de puissance de transmission est supérieur à celui du nœud source, il commence par vérifier si ce dernier lui offre un meilleur rang. Si c'est le cas, le nœud en réception change de rang et de parent préféré (il calcule son rang par rapport à ce nœud source du DIO et s'attache à celui-ci). Dans le cas où le nœud source n'offre pas un meilleur rang, le nœud en réception compare son niveau de puissance de transmission avec celui de son parent préféré actuel. Si son parent actuel a un niveau de puissance de transmission supérieur au sien, alors ce nœud change de rang et de parent préféré (il calcule son rang par rapport au nœud source du DIO et s'attache à celui-ci). L'objectif est de privilégier l'attachement entre le nœud en réception et le nœud avec lequel le lien est sûr d'être symétrique Dans ce cas, le nœud en réception descend dans le DAG en augmentant son rang pour éviter les liens asymétriques.

3. **Le niveau de puissance de transmission du nœud en réception est inférieur à celui du nœud source du DIO : le lien peut être asymétrique** (voir figure 5.6)

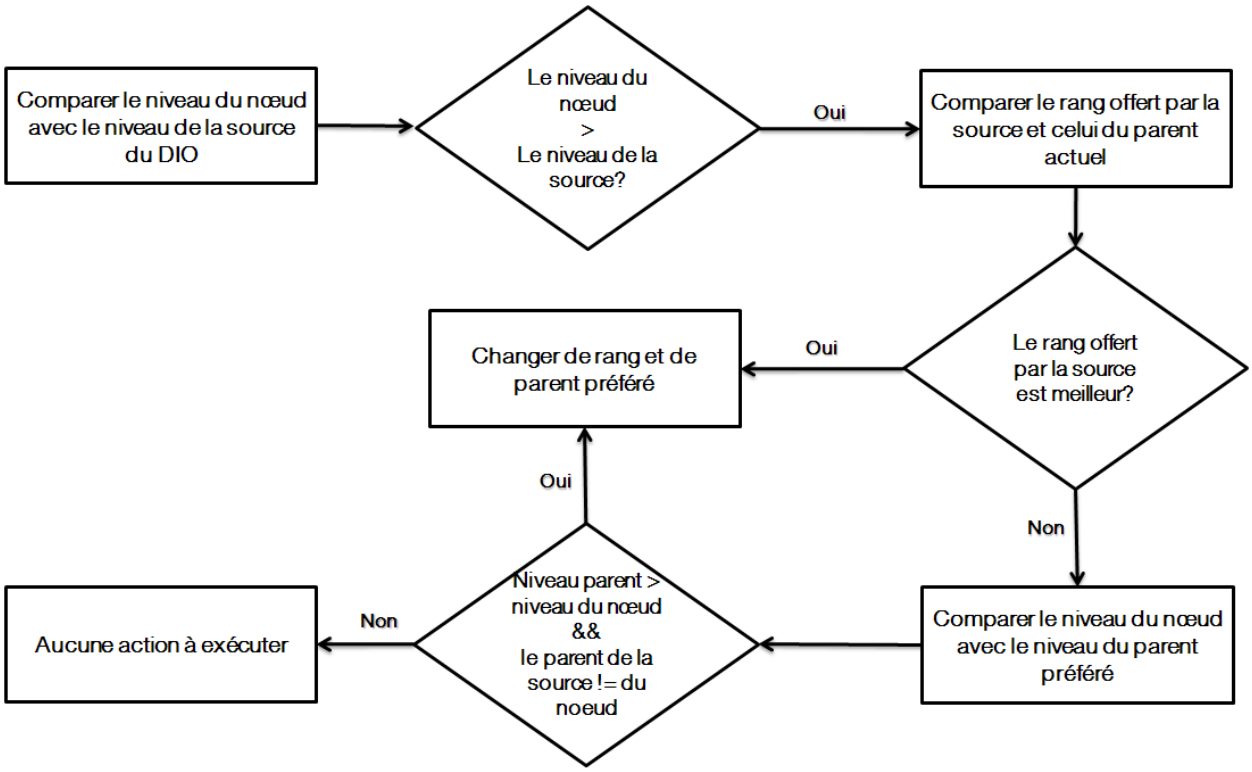


FIGURE 5.5: Principe de calcul de rang : le niveau du récepteur est supérieur à celui de la source

Quand le nœud en réception détecte que son niveau de puissance de transmission est inférieur à celui du nœud source, il commence par comparer son niveau de puissance de transmission avec celui de son parent préféré actuel. Si le parent préféré actuel a un niveau de puissance de transmission inférieur ou égal à celui du nœud en réception, alors ce dernier garde son parent préféré actuel. Dans le cas contraire, le nœud en réception compare le niveau de puissance de transmission de son parent préféré avec celui du nœud source du DIO. Trois possibilités se présentent :

- **Le niveau de puissance de transmission du parent préféré est strictement supérieur à celui du nœud source** : dans ce cas, le nœud en réception change de parent préféré. Le nœud en réception privilégie de s'attacher à un nœud ayant un niveau de puissance de transmission le plus proche du sien pour éviter les liens asymétriques.
- **Le niveau de puissance de transmission du parent préféré est strictement inférieur à celui du nœud source** : dans ce cas, le nœud en réception garde son parent préféré actuel puisque ce parent présente le niveau de puissance de transmission le plus proche du nœud en réception.
- **Le niveau de puissance de transmission du parent préféré est égal à celui du nœud source** : dans ce cas, le nœud en réception choisit de s'attacher au nœud le plus profond dans le DODAG. Le nœud en réception choisit, entre son parent préféré actuel et le nœud source, le nœud ayant le plus grand rang. Le choix du nœud ayant le plus grand rang permet de s'assurer qu'il est plus « proche logiquement »(en termes de rang) du nœud en réception. Ainsi le lien peut être un lien symétrique.

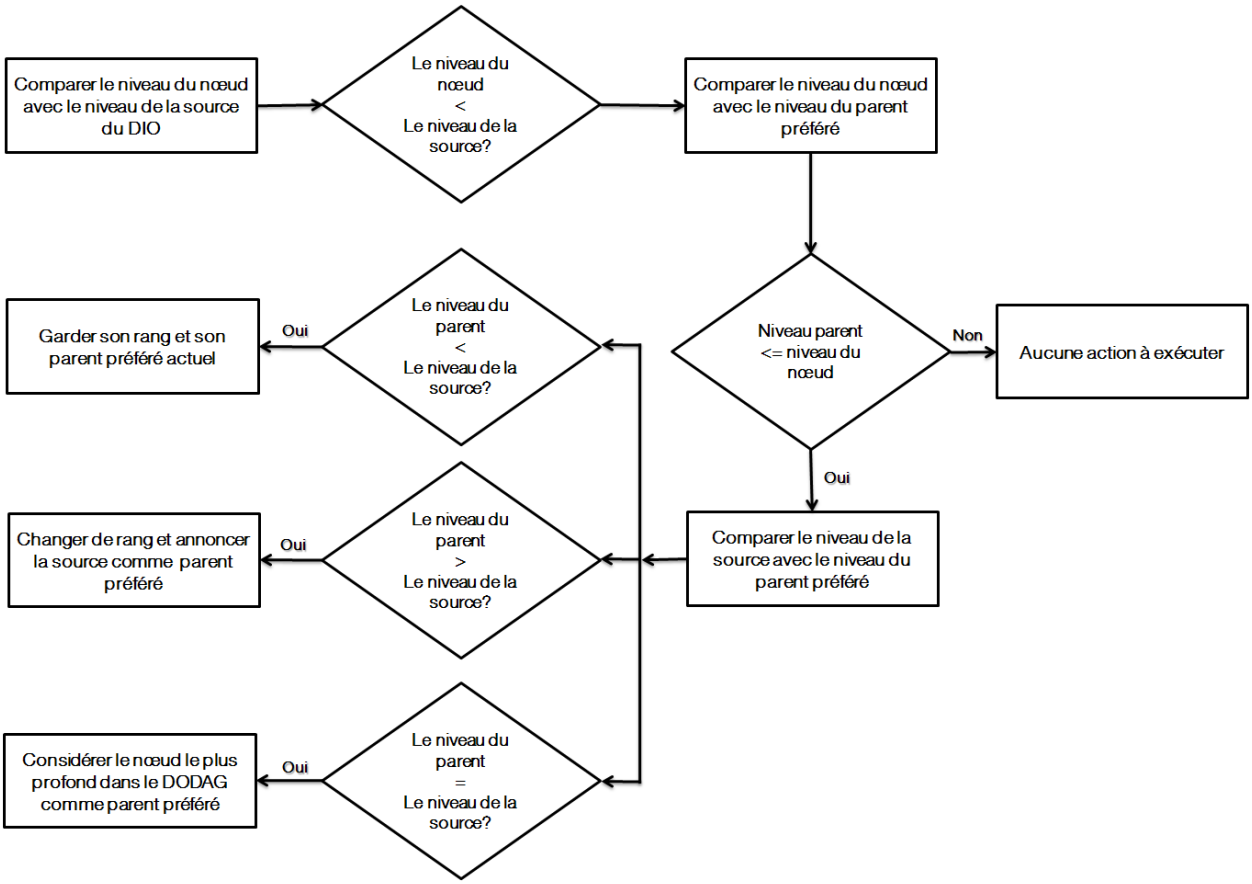


FIGURE 5.6: Principe de calcul de rang : le niveau du récepteur est inférieur à celui de la source

5.3.4 Exemple de construction du DODAG en présence d'une hétérogénéité

Pour mieux comprendre notre technique pour le calcul de rang et la construction du DODAG, nous représentons le scénario de la figure 5.7 où il y a un actionneur qui est la racine du DODAG et cinq nœuds capteurs. Chaque nœud du réseau possède un niveau de puissance de transmission représenté par les numéros au dessous des identifiants des nœuds. Pour simplifier, nous représentons la portée des nœuds par des cercles en pointillées.

- **L'actionneur commence par diffuser ses DIOs** :
  L'actionneur commence par diffuser ses DIOs qui seront reçus par les nœuds A et B (voir figure 5.8(a)). Ces derniers calculent leurs rangs en se basant sur la formule 5.1. Le nœud B considère maintenant que l'actionneur est son parent préféré. En fait le lien entre l'actionneur et le nœud B est un lien asymétrique : les messages envoyés par l'actionneur sont reçus par le nœud B mais pas inversement.
- **Le nœud A commence à diffuser ses DIOs** :
  Ensuite, le nœud A commence à diffuser ses DIOs (figure 5.8(b)) qui seront reçus par le nœud B. Ce dernier a le même niveau de puissance de transmission que le nœud A et constate que son parent préféré actuel a un niveau de puissance de transmission supérieur au sien, alors pour éviter que le lien soit asymétrique (ce qui est vrai dans cet exemple), le nœud B choisit de changer de parent préféré et de déclarer le nœud A comme étant son nouveau parent préféré (figure 5.8(b)).

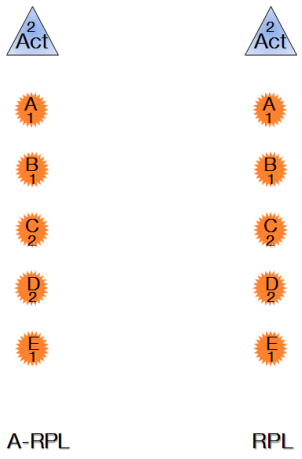


FIGURE 5.7: Réseau initial

- **Le nœud B commence à envoyer ses DIOs** :
  Par la suite, le nœud B commence à envoyer ses DIOs (figure 5.8(c)). A la réception, le nœud C déclare le nœud B en tant que son parent préféré.
- **Le nœud C diffuse des messages DIOs** :
  Le nœud C diffuse des messages DIOs. Ces DIOs seront reçus par les nœuds D et E (figure 5.8(d)). Ces derniers calculent leurs rangs et déclarent le nœud C comme étant leur parent préféré.
- **Le nœud D diffuse des messages DIOs** :
  Le nœud E a un niveau de puissance de transmission égal à 1 et il ne peut pas atteindre son parent préféré actuel. Avec notre algorithme de calcul de rang, le nœud E change de parent préféré quand il reçoit les DIOs diffusés par le nœud D (figure 5.8(e)). En effet, les deux nœuds C et D ont le même niveau de portée de transmission. Le nœud E choisit le plus profond dans la structure du DODAG comme étant son parent préféré pour éviter la possibilité d'avoir un lien asymétrique.

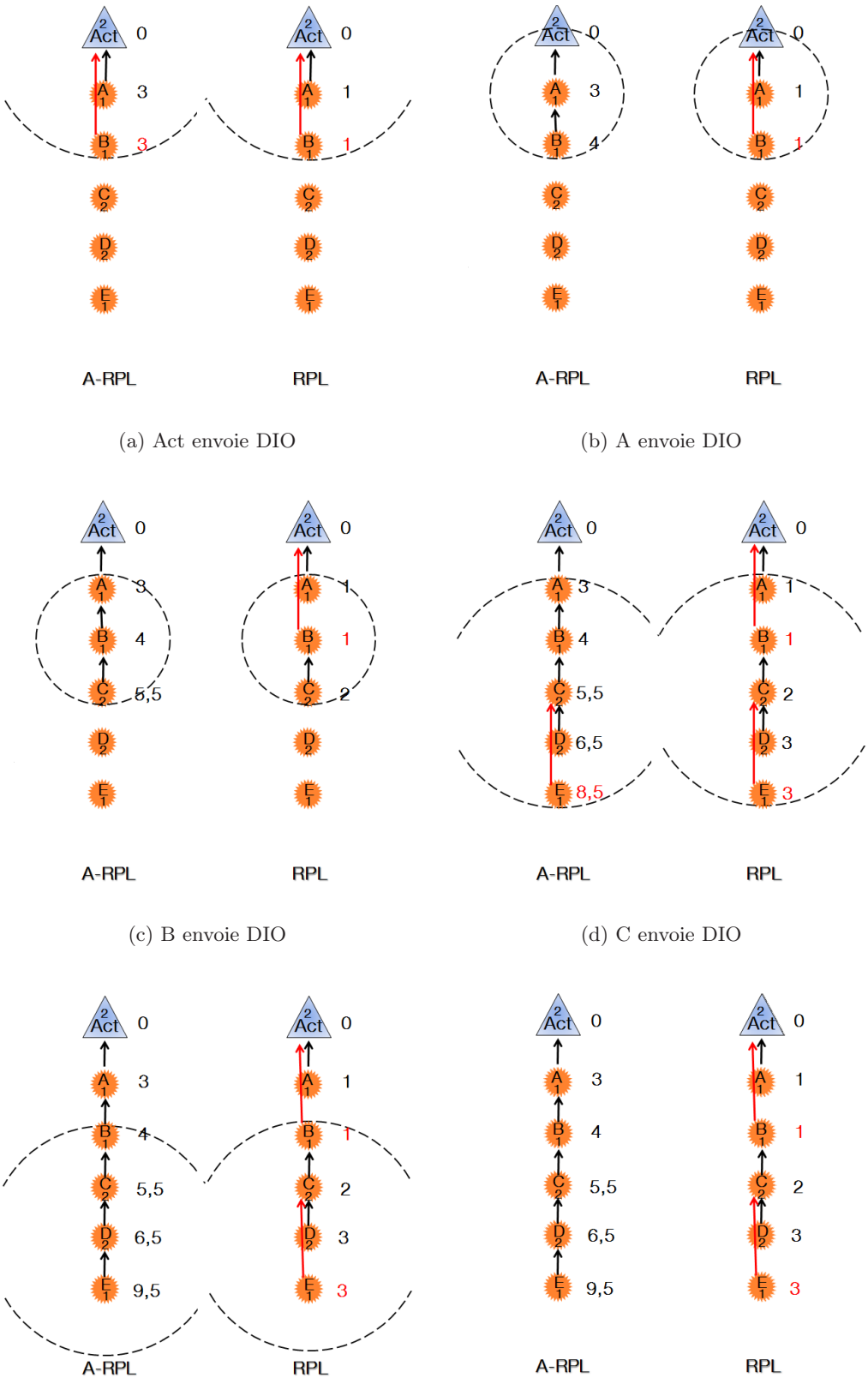
La structure DODAG finale est représentée par la figure 5.8(f). Sur cette topologie simple, nous pouvons, grâce à cet algorithme de calcul de rang, éviter les liens asymétriques causés par la présence de différents niveaux de portées de transmission.

5.4 Adaptation de Legos pour la création d'une topologie dans un contexte hétérogène : A-Legos

Après avoir introduit notre adaptation du protocole RPL pour détecter et éviter les liens asymétriques, nous proposons dans cette section une adaptation du protocole LEGOS [97] dans un même contexte. L'objectif de cette adaptation, appelée A-LEGOS, est de détecter la présence des liens asymétrique et de créer une topologie logique connexe facilitant une collecte de données fiable dans un réseau hétérogène.

5.4.1 Hypothèses et calcul de rang

Nous supposons que les nœuds ont une connaissance sur leurs voisinages à deux sauts. Chaque nœud aurait donc une information sur la liste de ses voisins directs et leurs voisins.



(a) Act envoie DIO

(b) A envoie DIO

(c) B envoie DIO

(d) C envoie DIO

(e) D envoie DIO

(f) Réseau final

FIGURE 5.8: Exemple de construction du DODAG dans un contexte hétérogène

A-LEGOS définit deux nouveaux types de nœuds dans le réseau : les *Sub-Member* et les *Sub-Gateway*.

- Un nœud *Sub-Member* est un nœud qui détecte dans son voisinage un nœud *Leader* mais qui ne peut pas le rejoindre à cause d'un lien asymétrique (Les messages diffusés par le *Leader* arrive à ce nœud mais les messages envoyés par ce dernier ne peuvent pas atteindre le nœud *Leader*).
- Un nœud *Sub-Gateway* est un nœud passerelle qui permet de relier un nœud *Sub-Member* à un *Leader* ou un nœud qui vient de s'attacher à la structure en tant que *Leader* à un nœud *Gateway*.

Comme dans LEGOS, seulement les nœuds *Leaders* diffusent périodiquement des messages dans leurs voisinages pour annoncer la présence d'une structure logique. Dans ces messages de diffusion, chaque *Leader* ajoute un champ contenant son rang. A la réception de ces messages de diffusion, les nœuds *Members* et *Gateways* attachés à ce *Leader* vont s'affecter un rang qui correspond au rang annoncé incrémenté d'une valeur de « 1 ». Les nœuds *Sub-Members* s'affectent le rang de leurs *Sub-Gateways* incrémentés d'une valeur de « 1 ». Les nœuds *Gateway* et les nœuds *Sub-Gateways* s'affectent, respectivement, le plus petit rang de leurs *Leaders* et de leurs *Gateways* ou *Leaders* incrémentés d'une valeur de « 1 ».

5.4.2 Description de l'algorithme : A-LEGOS

Construction de la topologie logique

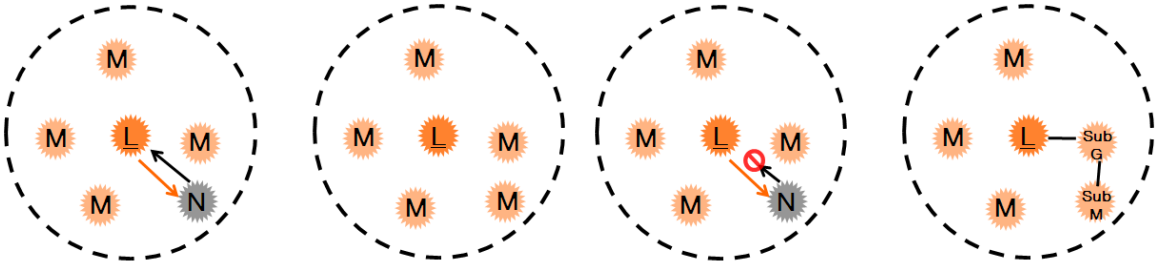
Le nœud puits ou le(s) actionneur(s) se déclare(nt) comme étant le(s) premier(s) *Leader(s)* de la structure. Il(s) diffuse(nt) ses (leurs) messages pour annoncer la présence d'une topologie logique dans le voisinage. Nous proposons d'utiliser le *timer* adaptatif Trickle pour éviter, comme pour RPL et A-RPL, les diffusions redondantes.

Pour intégrer la topologie logique, un nœud commence par écouter son voisinage :

1. s'il détecte un *Leader* dans son voisinage, alors deux cas de figure se présentent :
   - Si le lien est symétrique : Le nœud s'attache normalement à ce *Leader* comme le montre la figure 5.9(a).
   - Si le lien est asymétrique : Le nœud conclut qu'il peut recevoir les messages diffusés par ce *Leader* mais ne peut pas lui répondre parce que ses messages ne peuvent pas arriver jusqu'à ce *Leader*. Ce nœud envoie un message de découverte de voisinage, s'il détecte un nœud attaché à ce *Leader* alors il se déclare un *Sub-Member* et le nœud voisin passera en *Sub-Gateway* (figure 5.9(b)).
2. s'il ne détecte pas un *Leader* dans son voisinage, alors ce nœud lance une découverte dans son voisinage. Les nœuds recevant ce message ne répondent que lorsque le lien entre le nœud en réception et le nœud en phase de découverte de voisinage est symétrique. Quatre cas de figure se présentent :
   - Le message n'est reçu par aucun nœud. Après un délai sans aucune réponse à son message de découverte de voisinage, le nœud se déclare comme étant un *Leader* comme dans Legos (voir figure 5.10).
   - Si le message de découverte de voisinage est reçu par un *Member*, un *Gateway* ou un *Sub-Gateway*, ce dernier lui répond pour l'informer de l'existence d'une topologie logique dans son voisinage (Figure 5.11(a)). Dès la réception de cette notification, le nœud lançant cette découverte devient un *Leader* et le nœud lui informant de la présence de la structure devient un *Gateway* s'il ne l'est pas déjà (voir figure 5.11(a)).
   - Si le message de découverte de voisinage est reçu par un *Sub-Member*, ce dernier lui répond

pour l'informer de l'existence d'une topologie logique dans son voisinage (voir figure 5.11(b)). Dès la réception de cette notification, le nœud de découverte devient un *Leader*. Le nœud l'informant de la présence de la structure devient un *Sub-Gateway* et le nœud auquel est attaché ce dernier devient un *Gateway* s'il ne l'est pas déjà (voir figure 5.11(b)).

– Si le message de découverte de voisinage est reçu par un *Leader*, ce dernier réinitialise son compteur du *timer* Trickle et répond en diffusant son message d'annonce de présence d'une structure logique.



(a) Lien symétrique avec le *Leader*

(b) Lien asymétrique avec le *Leader*

FIGURE 5.9: A-Legos : Présence d'un *Leader* dans le 1-voisinage

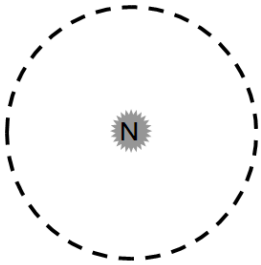
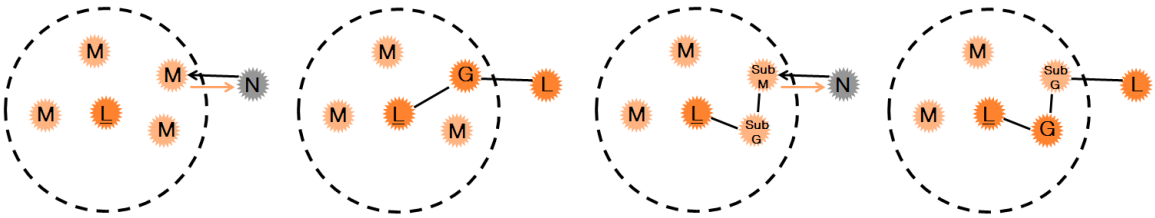


FIGURE 5.10: A-Legos : Aucune structure n'est détectée dans le voisinage



(a) Détection d'un nœud *Member* dans le voisinage

(b) Détection d'un nœud *Sub-Member* dans le voisinage

FIGURE 5.11: A-Legos : Présence d'un *Leader* dans le 2-voisinage

Collecte des données

Un nœud ayant des données à envoyer vers une destination finale, choisit le prochain saut en se basant sur son rôle au niveau de la topologie logique. Ce prochain saut est toujours un nœud ayant un rang plus faible que celui du nœud courant.

Le prochain saut sera choisi comme suit :
- Un nœud *Member* envoie ses données vers son *Leader*.
- Un nœud *Sub-Member* envoie ses données vers le nœud *Sub-Gateway* auquel il est attaché.
- Un nœud *Gateway* envoie ses données à un de ses *Leaders* ayant un rang plus petit que le rang de ce *Gateway*.
- Un nœud *Sub-Gateway* envoie ses données vers un de ses nœuds *Gateways* ou *Leaders* qu'il interconnecte, celui ayant le plus petit rang.
- Un nœud *Leader* envoie ses données vers le nœud *Gateway* ayant un rang plus faible que le sien.

5.5 Analyse de performances

Dans cette section, nous comparons les performances des deux propositions dans un réseau hétérogène. Ce réseau contient des liens asymétriques causés par la présence de différents types de nœuds ayant des niveaux de portées de communication différents.

5.5.1 Analyse de complexité

Nous commençons par faire une étude de complexité de nos deux adaptations A-RPL et A-Legos et nous les comparons avec le protocole de base RPL [155] et Legos [97]. Le tableau 5.1 récapitule cette comparaison.

Notation

Nous supposons que :
- N : Nombre de nœuds dans le réseau.
- M : Nombre de nœuds *Leader* dans la topologie construite par le protocole A-Legos ($M < N$).
- Δ : Degré maximal des nœuds.

Complexité de message

Dans [95], l'auteur montre que dans la structure construite par Legos le nombre de nœuds *Leader* et *Gateway* est borné et indépendant du nombre de nœuds dans le réseau. Puisque, pour A-Legos, seulement les nœuds *Leaders* envoient des messages de contrôle avec une période gérée par Trickle, alors la complexité de message pour cette proposition est de l'ordre de $O(M.\Delta)$. Pour A-RPL, tous les nœuds du réseau envoient des messages de contrôle avec une période gérée par Trickle. D'où la complexité de message pour cette proposition est de l'ordre de $O(N.\Delta)$. La proposition A-Legos engendre moins de complexité de message que A-RPL.

5.5 Analyse de performances

Complexité de calcul

Les deux propositions A-RPL et A-Legos ont la même complexité de calcul. Cette complexité est de l'ordre de $O(\Delta)$ puisque dans les deux cas les nœuds parcourent leurs listes de voisinage pour déterminer le prochain saut durant la phase de collecte de donnée.

Algorithme	Info. réseau	Info. de localisation	Compl. de calcul	Compl. de message
A-RPL	Locale	Non	$O(\Delta)$	$O(\Delta)$
A-Legos	Locale (2-voisinage)	Non	$O(\Delta)$	$O(\Delta)$

TABLE 5.1: Etude de complexité : A-RPL et A-Legos

5.5.2 Paramètres de simulation

Nous avons implémenté les deux propositions sous le simulateur WSNet [67]. Pour les résultats de simulation de cette section, nous considérons un réseau de 120 nœuds capteurs et un nœud puits déployé au centre du réseau. Deux types de topologies sont générés : une topologie en grille régulière et une topologie en grille aléatoire.

Nous supposons que les liens asymétriques sont causés par les niveaux de puissances de transmission des nœuds appelés super-nœuds. Suivant une loi aléatoire et uniforme, nous choisissons ces super-nœuds avec des portées de transmission différentes. Le nombre de super-nœuds varie entre 10% et 50%. Nous déclenchons plusieurs évènements applicatifs suivant une loi aléatoire et uniforme. Chaque évènement sera détecté par un nœud capteur et remonté vers le nœud puits. Le nombre d'évènements dans le réseau varie entre 10 et 50. Chaque scénario est simulé entre 100 et 200 fois avec un intervalle de confiance de 95%.

Le tableau 5.2 résume les principales caractéristiques du réseau et les paramètres de simulation.

Paramètre	Valeur
Topologies	Grille régulière et Grille aléatoire
Nombre de nœuds	120 nœuds capteurs
Portée des nœuds	1x, 2x et 3x portées des nœuds classiques
Super-nœuds	10% .. 30%
Nœuds source de données	10 .. 50
Protocole MAC	802.15.4 (CSMA)
Propagation	Modèle à deux rayons
Nombre de sauts maximum autorisé	16 sauts
Intervalle de confiance	95%
Simulateur	WSNet [67]

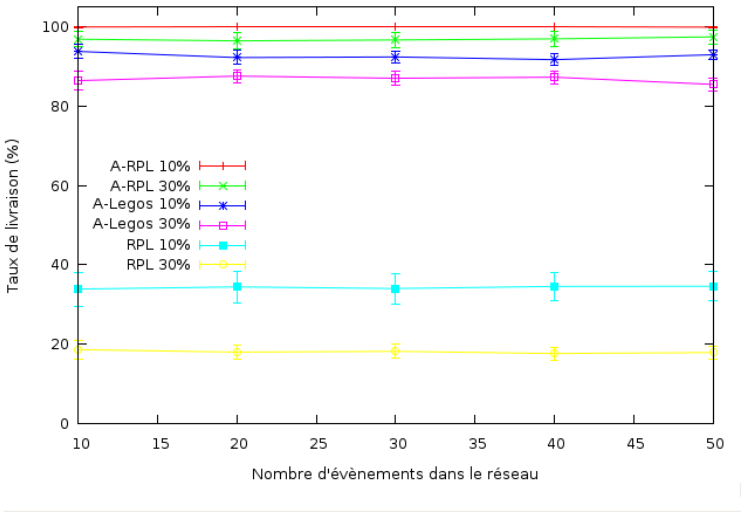
TABLE 5.2: Paramètres de simulations communs pour A-RPL et A-Legos

5.5.3 Taux de livraison

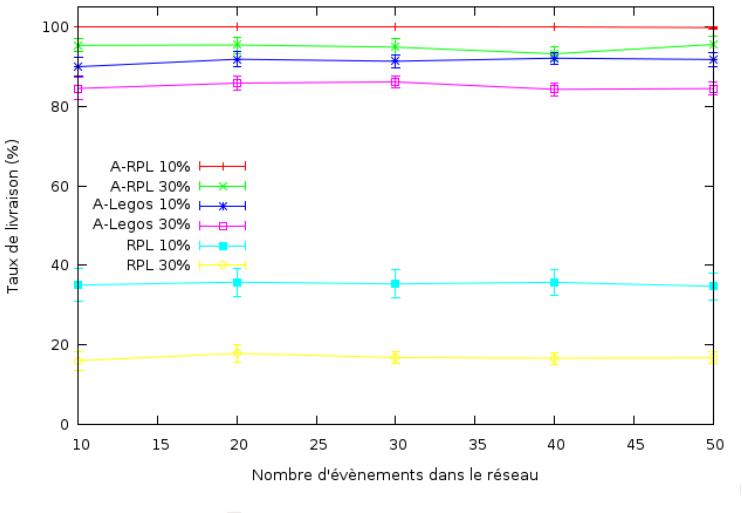
Dans la figure 5.12, nous représentons le taux de livraison des deux adaptations A-RPL et A-Legos et celui des deux protocoles de base RPL et Legos. D'une part, nous remarquons, pour les deux types de topologies, que le taux de livraison des protocoles de base est faible (voir figure 5.12(a) et

5.12(b)). Nous remarquons aussi que lorsque les liens asymétriques dans le réseau augmentent, le taux de livraison diminue parce que les deux protocoles de base ne sont pas conçus pour éviter ou exploiter les liens asymétriques.

D'autre part, nous vérifions que les deux adaptations offrent un taux de livraison élevé, proche de 100%. Avec ce résultat, nous attestons que A-RPL et A-Legos réussissent à éviter les liens asymétriques qui se présentent dans le réseau suite à l'existence de différents niveaux de portées de transmission.



(a) Topologie en grille régulière

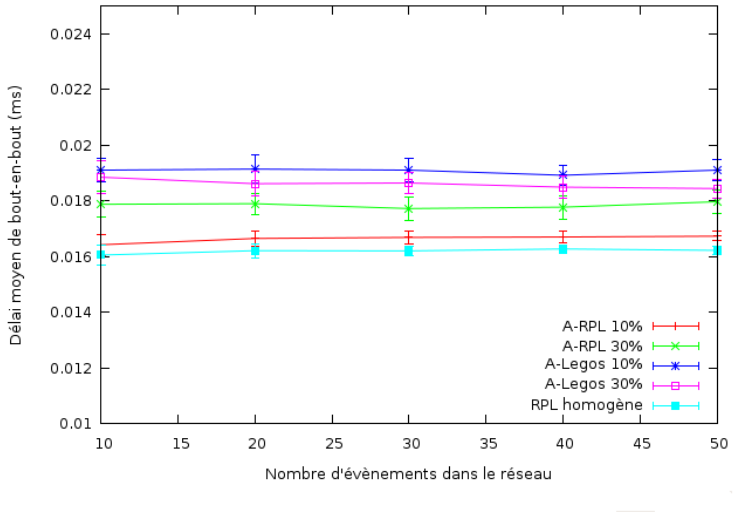


(b) Topologie en grille aléatoire

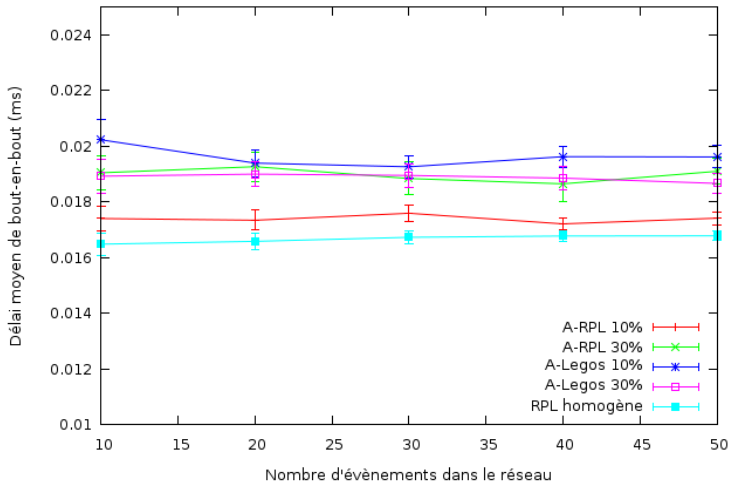
FIGURE 5.12: Evaluation du taux de livraison pour A-RPL et A-Legos

5.5.4 Délai de bout-en-bout

Sur la figure 5.13, nous remarquons que A-RPL offre un délai inférieur à celui offert par A-Legos. Nous remarquons aussi que, lorsque nous augmentons le nombre de super-nœuds, le délai offert par A-RPL augmente alors que celui de A-Legos diminue. En effet, dans ce cas, A-RPL augmente le diamètre du réseau (tout en augmentant les rangs), alors que A-Legos n'augmente pas ce diamètre puisque le nombre de nœuds *Leaders* est borné indépendamment du nombre de super-nœuds.



(a) Topologie en grille régulière

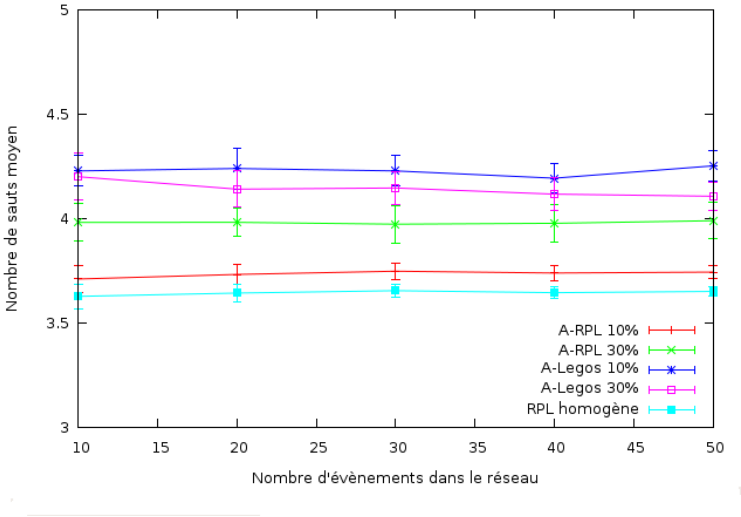


(b) Topologie en grille aléatoire

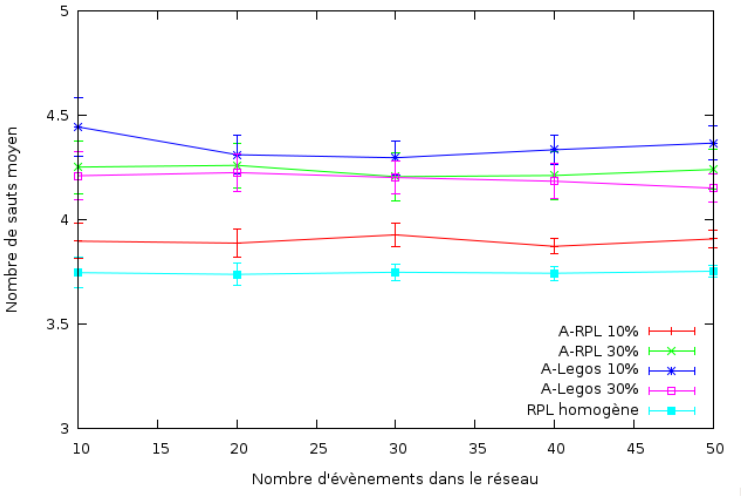
FIGURE 5.13: Evaluation du délai de bout-en-bout pour A-RPL et A-Legos

5.5.5 Allongement du chemin

Pour confirmer les résultats de la sous-section précédente, nous représentons, sur la figure 5.14, le nombre de sauts moyen qu'un message de donnée fait avant d'atteindre la destination finale. Pour les deux topologies, nous vérifions que la proposition A-Legos allonge le chemin. En effet, la longueur du chemin qu'offre la proposition A-Legos n'est pas optimale. Cependant, avec la proposition A-RPL, un nœud choisit le meilleur parent qui le rapproche le plus de la destination finale.



(a) Topologie en grille régulière

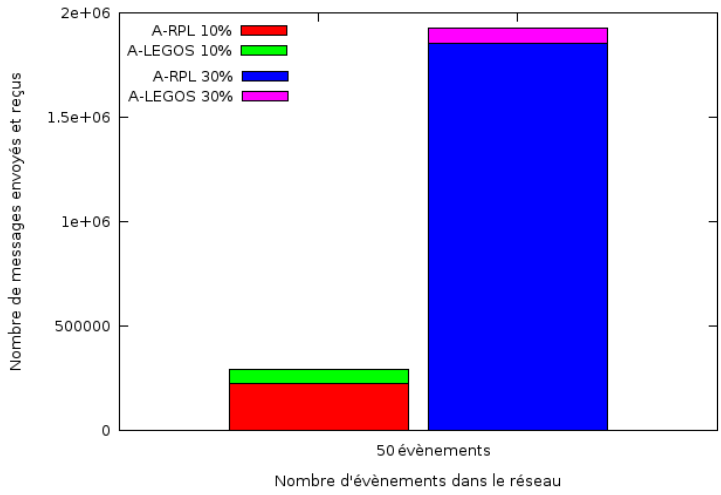


(b) Topologie en grille aléatoire

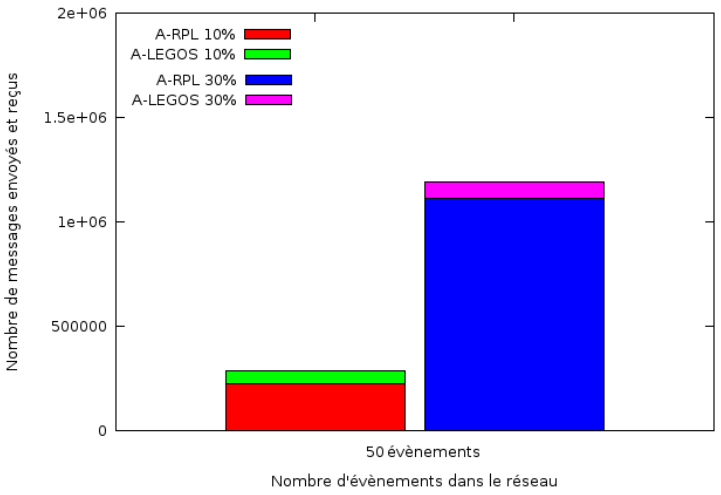
FIGURE 5.14: Nombre de sauts moyen pour A-RPL et A-Legos

5.5.6 Nombre de messages envoyés et reçus

Nous calculons ici le nombre de messages envoyés et reçus par les deux propositions que nous représentons dans la figure 5.15 lorsqu'il y a 50 évènements applicatifs dans le réseau. On voit bien que pour les deux types de topologies, la proposition A-Legos utilise moins de messages. Ceci est dû au fait qu'avec A-RPL, tous les nœuds diffusent des messages DIOs alors qu'avec A-Legos, seuls les nœuds *Leaders* envoient des messages de diffusion.



(a) Topologie en grille régulière



(b) Topologie en grille aléatoire

FIGURE 5.15: Nombre de messages envoyés et reçu pour A-RPL et A-Legos

5.6 Conclusion

Dans ce chapitre, nous avons présenté le protocole RPL en cours de standardisation au sein du groupe du travail ROLL de l'IETF. Nous avons étudié la défaillance de RPL, se basant sur le nombre de saut, en présence de liens asymétriques causés par l'hétérogénéité au niveau des portées de transmission. Pour y remédier, nous avons présenté une adaptation, A-RPL, qui s'appuie sur une méthode de calcul de métrique de rang permettant d'éviter les liens asymétriques. Nous avons également conçu une adaptation du protocole Legos. Cette proposition, A-Legos, permet aussi d'éviter les liens asymétriques tout en tirant profit des liens longs.

Avec des résultats de simulation, nous avons montré que les deux propositions offrent un taux de livraison élevé et que l'existence des liens asymétriques ne détériore pas les performances des deux propositions.

Nous avons attesté aussi que l'adaptation A-RPL permet d'offrir un meilleur délai et un nombre de saut de bout-en-bout moindre comparés à ceux offerts par l'adaptation A-Legos. Cependant, la proposition A-Legos consomme moins d'énergie. Ainsi, la proposition A-Legos peut être utilisée pour les applications critiques en énergie, où la consommation énergétique est de plus haute priorité. L'adaptation A-RPL est dédiée aux applications sensibles au délai, où les messages applicatifs doivent arriver au plus vite à la destination finale tout en sacrifiant de l'énergie au niveau des nœuds capteurs.

Exploitation des liens asymétriques pour le routage convergecast pour les réseaux hétérogènes de capteurs et actionneurs

6

Sommaire

6.1 Introduction

Plusieurs études et approches ont conçu des protocoles de routage pour les réseaux de capteurs sans fil (WSNs) et les réseaux de capteurs et actionneurs (WSANs). Néanmoins, la plupart d'entre eux ont supposé une homogénéité au niveau des liens. En d'autres termes, ils supposent que les liens constituants le réseau sont symétriques, ce qui est en contradiction avec la réalité de ces réseaux. En effet, les liens asymétriques ne peuvent pas être ignorés dans les WSNs et les WSANs, car ils peuvent être prédominants [72]. Ce type de liens peut être causé par l'existence de différents types de nœuds avec des puissances de transmission différentes, par l'environnement où on trouve dans les réseaux des nœuds interférant ou par l'utilisation de différentes antennes pour l'émission et la réception. L'existence des liens asymétriques peut considérablement dégrader les performances des protocoles de routage qui ne considèrent pas ce type de liens. Évidemment, la plupart des protocoles de routage existants supprime les liens asymétriques et maintiennent seulement les liens symétriques, ce qui les rend probablement moins performants et moins adaptés à la réalité de terrain de ces réseaux.

De notre point de vue, les liens asymétriques doivent être considérés car ils peuvent être efficaces dans l'assurance de la connectivité du réseau. Par ailleurs, ils ouvrent de nouvelles opportunités pour améliorer les performances des protocoles de routage. En effet, l'utilisation de ces liens asymétriques, qui sont généralement des liens à longue distance, nous permet de réduire le nombre de sauts pour atteindre la destination finale, d'augmenter ainsi le taux de livraison et d'exploiter de plus la diversité des liens et des routes dans le réseau.

Dans cette perspective et afin de tirer profit de ces liens asymétriques, nous proposons un protocole de routage pour la collecte des données dédiés aux WSANs appelé AsymRP (Asymmetric Routing Protocol). AsymRP est un protocole de routage dédié au trafic de collecte de données ou convergecast qui est basé sur une connaissance de voisinage à 2-sauts combinée avec l'utilisation des messages d'acquittements (ACKs) implicites et une technique de routage de messages ACKs explicites. Notre proposition tire profit des liens asymétriques, permet d'assurer un taux de livraison élevé tout en réduisant significativement le nombre de messages dupliqués et le nombre de sauts de bout-en-bout. Nos résultats de simulation montrent que notre proposition AsymRP surpasse nettement les protocoles de routage traditionnels lors de la présence de liens asymétriques dans le réseau.

Dans ce chapitre, nous commençons par présenter notre proposition AsymRP. Nous évaluons par la suite les performances de notre proposition. Nous discutons à la fin de ce chapitre une extension de AsymRP.

6.2 Motivation

L'existence de liens asymétriques ne peut être évitée dans les WSNs et les WSANs. En fait, les études comme dans [58] [82] [132] [165] ont démontré la présence de ces liens asymétriques. Ces liens asymétriques peuvent être causés par l'hétérogénéité du matériel utilisé (l'existence de différentes portées de transmission dans le réseau), par le déploiement réel (présence d'interférences dans le réseau) et par l'environnement (présence de source de bruit par exemple). Dans ce travail, nous nous sommes intéressés à la présence de liens asymétriques quelle que soit la source.

Pour surmonter les contraintes imposées par les liens asymétriques, deux solutions peuvent être envisagées. La première consiste à supprimer tous les liens asymétriques [81] [101] [109] [167]. L'élagage des liens asymétriques peut avoir plusieurs inconvénients : il peut provoquer la perte de la connectivité du réseau ou détériorer les performances des protocoles de routage. La deuxième solution propose d'utiliser les liens asymétriques dans le but de réduire la longueur du chemin et ainsi

limiter les délais de transmission de bout-en-bout [33] [85] [88] [110] [132] [136].

La plupart des travaux précédents considèrent seulement les liens asymétriques causés par l'hétérogénéité au niveau des portées de transmission [85] et [88] ou supposent une densité élevée pour assurer l'exploitation et l'acheminement des données vers la destination finale [33] ou souffrent d'un taux de messages dupliqués élevé au niveau de la destination finale [132].

Dans ce chapitre, nous proposons un nouveau protocole de routage appelé AsymRP (*Asymmetric Routing Protocol*). Nous adressons le problème des liens asymétriques dans un WSN ou un WSAN connecté sans aucune contrainte sur la densité du réseau ni sur l'origine de l'asymétrie du lien. Avec AsymRP, un nœud recevant un message à transmettre vers une destination finale décide de participer ou non dans le processus de la collecte des données. Cette décision sera prise sur la base des informations contenues dans le message reçu et en fonction d'une connaissance locale du voisinage.

Avec la présence de liens asymétriques, le premier défi est de savoir comment détecter la présence de cette asymétrie. Pour relever ce défi, nous présenterons un mécanisme simple basé sur l'échange de la table de voisinage en utilisant des messages *Hello*. Un autre défi consiste à savoir comment exploiter ces liens pour transmettre un message de données vers la destination finale et en même temps comment assurer un retour pour acquitter la bonne réception du message de données saut par saut afin d'éviter les retransmissions inutiles et réduire les messages dupliqués. Pour relever ce défi, nous présenterons un mécanisme basé sur des acquittements implicites et explicites.

Notre proposition est en mesure de livrer les messages à partir des nœuds source vers le nœud de destination indépendamment de la topologie et la densité du réseau. Pour évaluer le bénéfice de l'exploitation des liens asymétriques dans le réseau, nous compterons le nombre de ce type de liens utilisé dans chaque chemin trouvé pour livrer un message de données.

6.3 AsymRP : Un protocole de routage pour le trafic convergecast exploitant les liens asymétriques

Dans ce chapitre, nous proposons un protocole de routage convergecast dédié aux réseaux connectés ayant des liens asymétriques dans le réseau. Comme nous l'avons mentionné dans l'introduction, nous ne nous intéressons pas à la cause qui a provoquée l'apparition de ces liens asymétriques. Notre proposition, appelée AsymRP, tire profit de l'existence des liens asymétriques pour la collecte des données tout en évitant les messages redondants et tout en réduisant le nombre de sauts à partir des nœuds sources vers les nœuds de destination. AsymRP peut être diviser en deux phases : une phase de découverte de voisinage et une phase de collecte de données.

6.3.1 Hypothèses

Nous considérons un réseau avec un grand nombre de nœuds source et un seul nœud de destination. Le réseau est supposé stable et statique. Nous supposons qu'il existe plusieurs liens asymétriques dans le réseau. À $t = 0$, nous supposons que tous les nœuds sont déployés et actifs.

Aucune information géographique réelle n'est disponible pour les différents nœuds du réseau. Mais nous supposons que les capteurs ont des informations sur leur rang par rapport au nœud destination finale (le nœud puits). Ces rangs seront utilisés pour la collecte de données. Ces rangs ont la propriété d'être strictement décroissant le long de toute route en se rapprochant du nœud puits. Nous supposons que les nœuds les plus proches de la destination ont un plus petit rang. La disposition des rangs peut être obtenue comme décrit dans [14] ou dans [127].

6.3.2 Première phase : Phase de découverte de voisinage

Cette phase peut être divisée en deux sous-étapes. En supposant que les nœuds du réseau ont un rang, chaque nœud diffuse deux messages.

1. **D'abord**, chaque nœud envoie un message de découverte de voisinage appelé *Hello_ Msg*. Dans ce message, chaque nœud met son propre identifiant ID et son rang (qui est égal à 0 pour le nœud de destination). A la réception de ce message, chaque nœud ajoute à sa table de voisinage le nœud qui vient de s'annoncer.
2. **Ensuite**, chaque nœud diffuse un second message, à travers lequel il annoncera la liste des voisins détectés, nommé *Heard_ Neighbor_ Msg*. Ce message contient l'ID du nœud, son rang et la liste des IDs des voisins dont il a connaissance. À la fin de cette phase, chaque nœud aura une connaissance sur son 1-voisinage ouvert [1] : il aura construit une table de voisinage complète contenant la liste de ses voisins, leurs rangs et la liste de ses voisins à deux sauts (les voisins de ses voisins).

6.3.3 Deuxième phase : Phase de collecte de données

Dans cette phase, quand un nœud a des données à envoyer vers la destination, il diffuse le message de données dans son voisinage. Dans l'en-tête de ce message, le nœud émetteur met son ID, son rang et sa table de voisinage construite lors de la première phase.

Chaque nœud émetteur, que nous appelons *Source*, démarre un *Timer*. Durant ce délai, que nous appelons *Timeout_ avant_ réémission*, ce nœud *Source* attend un message d'acquittement (ACK) implicite ou explicite. Le message ACK implicite consiste en ce que ce nœud *Source* entend son message relayé par un autre nœud. Le message ACK explicite est un message envoyé explicitement de la part du nœud relayeur vers le nœud *Source*. Si le délai du *Timer* est écoulé alors que le nœud *Source* n'est pas informé que son message a été relayé, il tente d'envoyer son message de données une nouvelle fois. Le calcul de ce *Timer* sera discuté dans la section 6.3.5.

Lors de la réception d'un message diffusé, un nœud récepteur applique l'algorithme décrit dans la figure 6.1 : ce nœud récepteur vérifie d'abord s'il possède un rang plus petit que celui du nœud source, c'est-à-dire s'il est plus « proche »de la destination que le nœud source. Si c'est le cas, alors ce nœud récepteur est appelé un nœud *Candidat*. Dans le cas contraire, ce nœud ne peut pas participer à la procédure de l'acheminement de ce message, car il possède un rang strictement supérieur à celui du nœud *Source* et il est ainsi plus « loin »de la destination que le nœud *Source*. Ce nœud supprime le message de données reçu.

Si ce nœud est un *Candidat* pour relayer ce message, il calcule un délai d'attente appelé *Timeout_ avant_ relayage* et entre dans une phase de contention. L'objectif de ce délai est de favoriser le nœud ayant le plus petit rang et ainsi le plus proche de la destination. Ce délai est également décrit dans la section 6.3.5.

Tant que le nœud n'entend pas le message relayé par un autre nœud ou il ne reçoit pas un ACK explicite à faire suivre vers le nœud *Source*, ce nœud commence par vérifier la nature du lien qui le relie avec le nœud *Source*. Il peut vérifier la nature du lien en se basant sur la table de voisinage construite dans la première phase.

1. **Dans le cas où le lien est symétrique** : le nœud *Candidat* peut relayer le message de données qui servira en même temps comme un ACK implicite pour le nœud *Source*.

1. Un nœud possédant une connaissance du 1-voisinage ouvert, c'est-à-dire, qu'il a une connaissance des nœuds à 2 sauts connectés à chacun de ses voisins directs

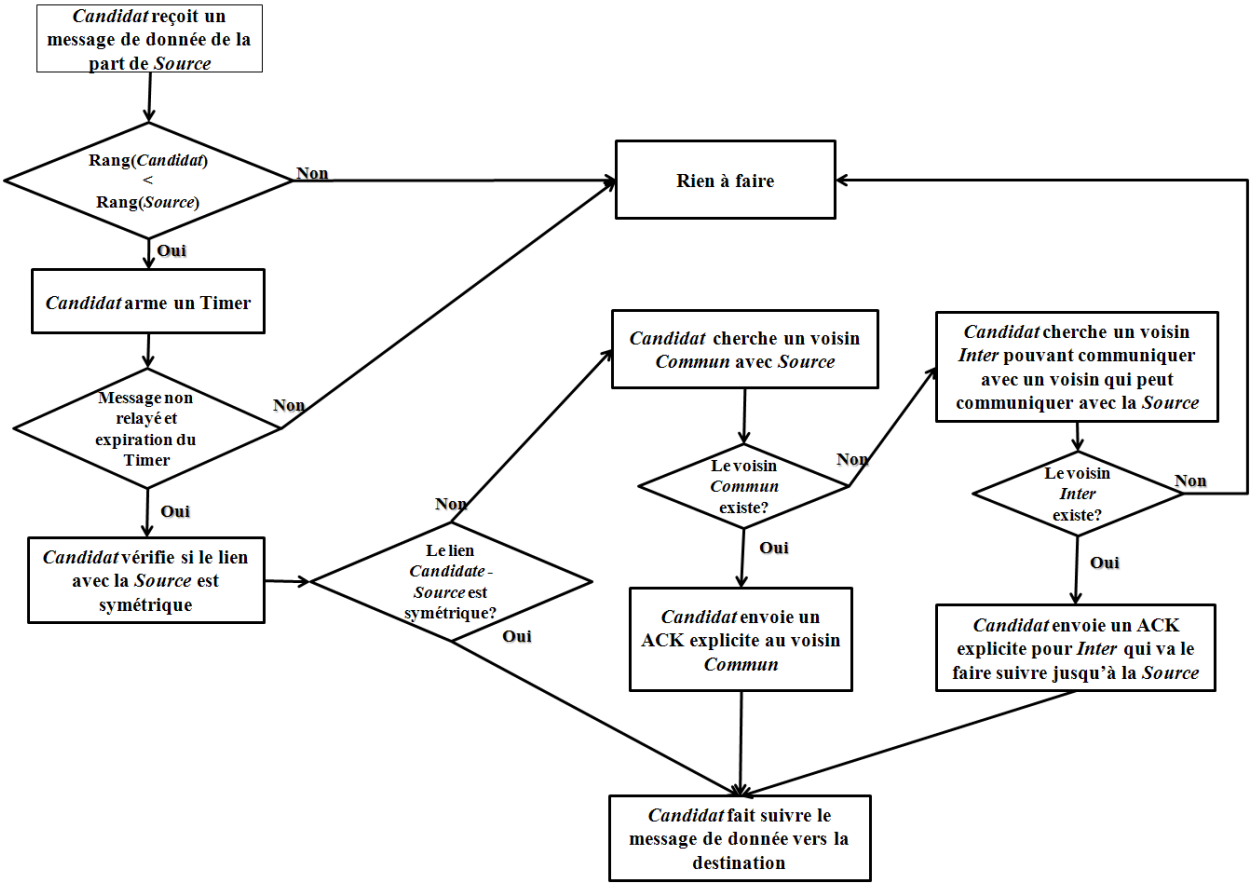


FIGURE 6.1: AsymRP : Principe de la phase de collecte de données.

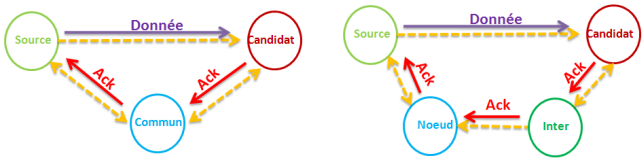
2. **Dans le cas où le lien est asymétrique** : trois cas de figure se présentent :

   a) Le nœud *Candidat* commence par chercher un nœud en commun avec le nœud source. Un exemple de ce nœud, appelé *Commun*, est représenté dans la figure 6.2(a). Ce nœud *Commun* servira de relais pour l'acheminement du message ACK explicite.

   b) Si le premier cas est infructueux, le nœud *Candidat* cherche un nœud intermédiaire à partir de sa table de voisinage ainsi que la table de voisinage du nœud *Source* envoyée dans le message de données. Ce nœud, appelé *Inter*, peut communiquer avec le nœud *Candidat* et avec un voisin du nœud *Source*. Un exemple de ce nœud, est représenté dans la figure 6.2(b). Si ce nœud *Inter* existe, le nœud *Candidat* fait suivre le message de données et envoie un ACK explicite vers le nœud *Inter* qui le fera suivre vers le voisin qui peut communiquer avec le nœud *Source* (voir la figure 6.2(b)).

   Dans ces deux cas, si le nœud *Commun* ou le nœud *Inter* existent, alors le nœud *Candidat* fait suivre le message de données vers la destination finale et envoi un ACK explicite vers le nœud *Commun* ou *Inter* qui le fera suivre respectivement vers le nœud source ou vers un voisin de ce nœud source. Si un nœud *Candidat* détecte que le message a été relayé par un autre nœud, alors il supprime le message de données et ne se considère plus comme étant un nœud *Candidat*.

   c) Si les deux cas précédents sont infructueux, le nœud ne participe pas à la procédure



(a) Détection d'un voisin *Commun*.

(b) Détection d'un voisin *Inter*.

FIGURE 6.2: Utilisation des voisins *Commun* et *Inter* pour envoyer un message ACK explicite.

d'acheminement des données. Une extension est proposée dans la section 6.5 pour traiter où le cas a) et b) ne sont pas satisfaits.

6.3.4 Exemple

Prenons l'exemple illustré par la figure 6.3. Nous supposons que le réseau est composé de sept nœuds (*Src*, *A*, *B*, *C*, *D*, *E* et *F*) et un seul nœud de destination (*Dst*). Chaque nœud a un rang qui détermine son gradient par rapport au nœud de destination (*Dst*). Ce rang est représenté par le nombre écrit en dessous de chaque nœud dans la figure 6.3. Après la première phase de découverte de voisinage, chaque nœud construit sa table de voisinage.

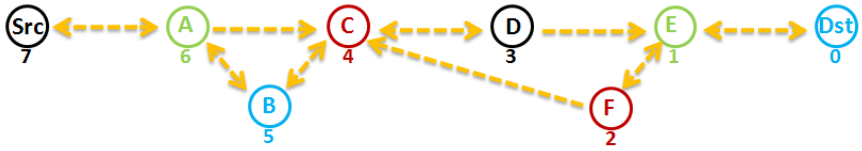


FIGURE 6.3: Exemple d'une topologie avec des liens asymétriques : le nœud *Src* envoie un message de données vers le nœud *Dst*.

1. **Le nœud *Src* envoie un message de données** : Le nœud source *Src* a un message de données à envoyer vers le nœud de destination *Dst*. Le nœud *Src* diffuse son message de données en mettant dans l'entête son identifiant (ID), son rang et sa table de voisinage (voir figure 6.4).
2. **Le nœud *A* fait suivre le message de données** : Le message est reçu par le nœud *A*. Ce dernier est un nœud candidat pour le relayer, car il a un plus petit rang. Le nœud *A* lance un *Timer Timeout_avant_relayage*. Après l'expiration de ce délai, le nœud *A* commence par vérifier la nature du lien entre lui et le nœud *Src* parce qu'il n'a pas entendu ce message relayé par un autre nœud. En consultant sa table de voisinage, le nœud *A* déduit que le lien entre lui et le nœud *Src* est symétrique, ainsi il diffuse ce message de données dans son voisinage (voir figure 6.5) en mettant à jour les informations contenues dans l'en-tête (le nœud *A* met son

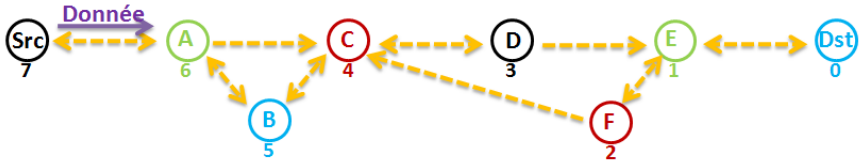


FIGURE 6.4: Le nœud *Src* diffuse son message de données

propre ID, son rang et sa table de voisinage). Quand le nœud *Src* entend son message relayé par le nœud A, il met alors fin à son délai d'attente *Timeout_ avant_ réémission* et considère ce message entendu comme un ACK implicite.

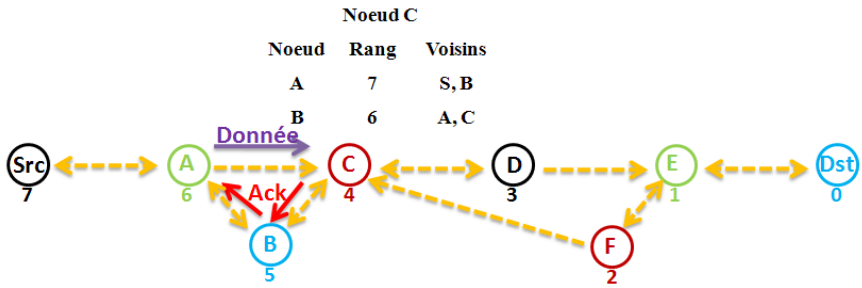


FIGURE 6.5: Le nœud A fait suivre le message de données et reçoit un ACK explicite

3. **Les nœuds *B* et *C* sont candidats pour faire suivre le message de données** : Le message relayé par A sera également reçu par les nœuds B et C (voir figure 6.5). Chacun d'eux va lancer un délai *Timeout_ avant_ relayage*. Le délai du nœud C s'écoule le premier puisque le nœud C a le plus petit rang (C a un rang égal à 4 tandis que le rang de B est 5). Le nœud C constate que le lien avec le nœud A est asymétrique, alors il vérifie s'il possède avec ce nœud émetteur un voisin en commun. Le nœud C déduit à partir de sa table de voisinage que le nœud B est un voisin commun avec le nœud A (figure 6.5). Ainsi le nœud C diffuse le message de données dans son voisinage et envoie un ACK explicite au nœud B qui à son tour le fait suivre au nœud A. En recevant ce message ACK, le nœud B, qui est en phase de contention avec le nœud C, supprime son *Timer* et le message reçu à partir du nœud A (voir figure 6.5).
4. **Le nœud *D* fait suivre le message de données** : Le message de données relayé par le nœud C sera reçu par le nœud D. Ce dernier déduit à partir de sa table de voisinage que le lien avec le nœud C est symétrique, il diffuse alors à son tour le message de données en mettant à jour son entête (voir figure 6.6). Ce message relayé sera reçu par le nœud C et par le nœud E. Le nœud C considère ce message comme ACK explicite.
5. **Le nœud *E* fait suivre le message de données** : Le nœud E déduit que le lien est asymétrique et qu'il ne possède pas un voisin en commun avec le nœud D. Le nœud E cherche alors dans sa table de voisinage ainsi que dans celle envoyée par le nœud D s'il y a un voisin de E qui peut communiquer avec un voisin direct du nœud D. Le nœud E déduit que le nœud F

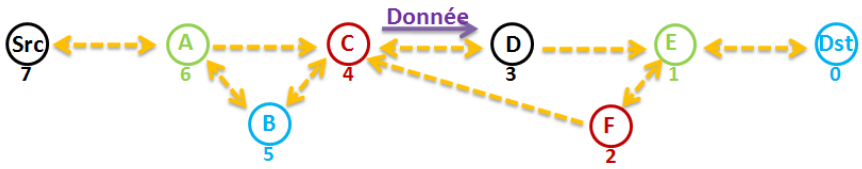


FIGURE 6.6: Le nœud C fait suivre le message de données et reçoit un ACK implicite

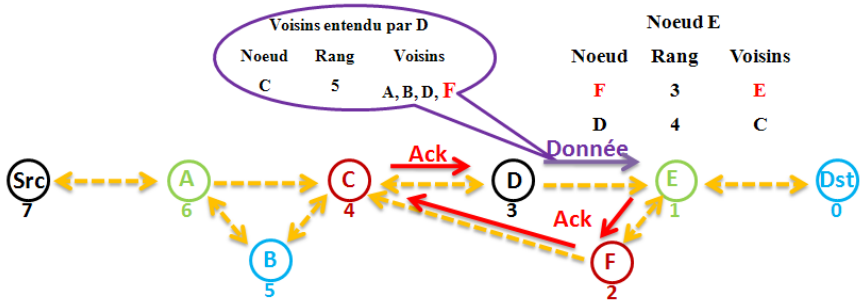


FIGURE 6.7: Le nœud D fait suivre le message de données et reçoit un ACK explicite

peut communiquer avec le nœud C qui est un voisin direct du nœud émetteur D (voir figure 6.7). D'où, le nœud E fait suivre le message de données vers la destination finale et envoie un ACK explicite vers le nœud F qui le fera suivre au nœud C, qui à son tour l'envoie vers le nœud D comme le montre la figure 6.7. Le nœud E mentionne dans l'entête du message ACK explicite la route que doit suivre ce message. Le message de données relayé par le nœud E sera reçu par le nœud Dst et le nœud F (voir figure 6.8). Le nœud F supprime ce message car il possède un rang supérieur à celui du nœud émetteur (car le rang du nœud E est égal à 1 et celui du nœud F est égal à 2).

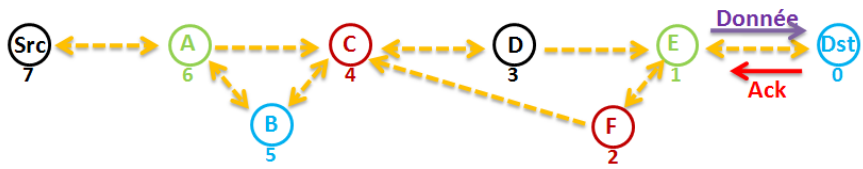


FIGURE 6.8: Le nœud E fait suivre le message de données et reçoit un ACK explicite

6. **A la réception du message de données, le nœud destination finale *Dst* diffuse un ACK** : Le nœud destination finale *Dst*, à la réception du message de données, diffuse un ACK pour que le dernier nœud relayeur arrête son *Timer Timeout_ avant_ réémission* et aussi pour

que les nœuds qui sont en phase de contention pour ce message de données arrêtent aussi leur *Timers Timeout_ avant_ relayage* (voir figure 6.8).

6.3.5 Calcul des Timeouts

Notre proposition définit deux *Timers* : le premier, appelé *Timeout_ avant_ relayage*, est calculé par les nœuds candidats à relayer les messages de données vers la destination finale. Le second, appelé *Timeout_ avant_ réémission*, est calculé par les nœuds émetteurs.

a. Délai d'attente avant la retransmission du message en direction de la destination : *Timeout_ avant_ relayage*

Le *Timeout_ avant_ relayage* est calculé par les nœuds candidats capables de relayer le message de données et qui entrent dans une phase de contention pour décider d'une manière distribuée lequel d'entre eux va relayer le message.

L'objectif de ce *Timer* est d'introduire des priorités entre les nœuds candidats. Plus le délai est petit plus le nœud est prioritaire pour relayer le message de données. Le but est de favoriser les nœuds les plus proches de la destination (ayant le plus petit rang) pour réduire le nombre de sauts. Ainsi, le délai calculé est proportionnel au rang du nœud candidat : plus le rang est petit, plus le délai est court. Le délai est calculé comme représenté dans l'équation 6.1.

$$\text{Timeout\_avant\_relayage} = \alpha * Rang + \beta \tag{6.1}$$

Dans l'équation 6.1, α et β sont deux paramètres de notre proposition AsymRP. α est une constante et β est une variable aléatoire.

Plus la valeur de la constante α est grande, plus la taille de la phase de contention est grande et ainsi le temps d'attente avant la participation au relayage du message de données est grand. Si l'objectif de l'application est de minimiser le délai de bout-en-bout des messages de données, la valeur de α doit être petite. Mais en contre partie le nombre de collisions et le nombre de messages dupliqués va augmenter quand la valeur de α est trop petite.

En ce qui concerne la variable aléatoire β, elle permet d'introduire une priorité entre les voisins ayant le même rang. Elle est liée à l'état de connectivité du nœud avec son voisinage : plus le degré du nœud est élevé, plus la valeur de β est faible. Cette varible permet aussi de diversifier les routes choisies lors de la phase de collecte de données.

b. Délai d'attente avant la réémission du même message : *Timeout_ avant_ réémission*

Ce *Timer* est initié par les nœuds émetteurs. Il est utilisé pour s'assurer que le message de données est relayé par un autre nœud vers la destination.

Timeout_ avant_ réémission est la somme de deux composantes comme le montre l'équation 6.2. La première est la limite supérieure de *Timeout_ avant_ relayage*, la seconde composante, appelée γ, représente une estimation du temps que doit attendre un nœud avant de recevoir un message ACK implicite ou explicite. En effet, la durée maximale qu'un nœud devrait attendre avant d'entendre son message relayé par un voisin direct est égale à la limite supérieure du temps d'attente de ce voisin avant de relayer le message. Si le message est relayé par un nœud que le nœud émetteur ne peut pas entendre, ce dernier doit en plus attendre la réception d'un ACK explicite avant de tenter d'envoyer son message de données une seconde fois.

$$\text{Timeout\_avant\_réémission} = Max(Timeout\_avant\_relayage) + \gamma \quad (6.2)$$

6.4 Evaluation des performances

Cette section est divisée en deux parties : nous commençons par une étude théorique dans laquelle nous évaluons la consommation énergétique de notre proposition AsymRP et nous la comparons avec celle du protocole TRIF [88]. Ce dernier est un protocole de collecte de données profitant des liens asymétriques présents dans le réseau. Dans la deuxième partie, nous évaluons les performances de AsymRP et de TRIF en utilisant la simulation avec des hypothèses plus réalistes. Pour être équitable entre les deux protocoles, il est à noter que pour TRIF, nous avons ajouté un message d'acquittement (ACK) envoyé par le nœud puits lorsqu'il reçoit un message de données. Cet ACK est utilisé pour éviter les réceptions dupliquées. Sans perte de généralité, dans les deux parties nous considérons que les liens asymétriques sont causés par la présence de différentes puissances de transmission dans le réseau. Dans ce cas, nous définissons deux types de nœuds :

- Les nœuds classiques : qui sont les nœuds capteurs ayant une puissance de transmission basique.
- Les super-nœuds : les nœuds ayant une puissance de transmission élevée.

6.4.1 Etude théorique : évaluation numérique

A. Analyse de complexité

Dans cette section, nous commençons par faire une étude de complexité de notre proposition et la comparer avec le protocole TRIF. Le tableau 6.1 récapitule cette comparaison.

1. **Notation**
   Nous supposons que :
   - n : Nombre de nœuds dans le réseau.
   - x : Coefficient multiplicatif de la portée des super-nœuds par rapport aux nœuds classiques dans le réseau.
   - Δ : Degré maximal des nœuds.
2. **Complexité de message**
   Durant l'exécution de notre proposition AsymRP, le coût d'échange de message pour un nœud afin de découvrir son voisinage est de O(Δ) parce que chaque nœud échange avec ses voisins deux messages de contrôle (un message de découverte de voisinage *Hello_Msg* et un message d'annonce de voisins *Heard_Neighbor_Msg*).
   Pour TRIF, le coût d'échange de message pour un nœud durant la phase de collecte de données est de l'ordre de O(x.Δ). En effet, TRIF ne nécessite pas une connaissance de voisinage, mais chaque nœud, quand il émet des données vers un nœud destination, envoie x messages consécutifs avec des niveaux de puissance de transmission décrémentés à chaque fois.
3. **Complexité de calcul**
   La complexité de calcul pour TRIF est de O(x). Cependant, pour AsymRP la complexité est de O(Δ^3). En effet, durant la phase de collecte de données un nœud exécutant AsymRP doit parcourir sa liste de voisin pour vérifier la symétrie du lien et parcourir la liste de ses voisins directs et la liste de ses voisins à 2-sauts pour trouver un voisin *Commun* ou *Intermédiaire* à travers lequel il pourra envoyer un ACK explicite durant la phase de collecte de données.

Algorithme	Info. réseau	Compl. de calcul	Compl. de message	Compl. de mémoire
TRIF	Locale	O(x)	O(x.Δ)	O(1)
AsymRP	Locale (2-voisinages)	O(Δ^3)	O(Δ)	O(Δ^2)

TABLE 6.1: Etude de complexité de AsymRP et TRIF

Comme nous pouvons voir dans le tableau 6.1, notre proposition AsymRP introduit plus de complexité que TRIF. Il y a un compromis entre la complexité et les performances assurées par notre proposition comparées à celles du protocole TRIF, que nous vérifions dans ce qui suit.

B. Evaluation de consommation énergétique

Dans cette section, nous nous intéressons à l'évaluation de la consommation énergétique de notre proposition AsymRP et de TRIF. AsymRP exige une connaissance du voisinage et il existe un compromis entre le coût de l'énergie pour obtenir cette information et le coût énergétique de la phase de collecte des données. En effet, pour des applications de collecte de données fréquentes, le coût de la découverte du voisinage dans un réseau statique (faite au début du déploiement du réseau) peut être négligeable par rapport au coût de l'envoi de données périodiques au(x) nœud(s) destination(s).

Dans cette section, nous commençons par évaluer le coût de la phase de découverte du voisinage et la phase de collecte des données de la proposition AsymRP. Nous calculons le nombre de messages envoyés et reçus pour chaque phase pour des réseaux à faible et à grande densité. Nous comparons la consommation énergétique de AsymRP et de TRIF à la fin de cette section.

1. **Paramètres et hypothèses**

   Nous supposons un réseau uniformément distribué dans une zone géographique donnée. Nous admettons également que les messages de données et des messages de contrôle ont la même taille. Nous appelons :
   - A : Nombre de nœuds classiques.
   - B : Nombre des super-nœuds.
   - N : Nombre de nœuds dans le réseau ($N = A + B$).
   - V : La densité géographique des nœuds. Nous supposons que la densité géographique est uniforme pour les nœuds classiques et les super-nœuds.
   - R : la portée radio des nœuds classiques. Ainsi le nombre de voisins pour un nœud classique est égal à $R^2.V.\pi$.
   - x : Le coefficient multiplicatif de la portée de transmission des super-nœuds par rapport aux nœuds classiques. Ainsi le nombre de voisins pour un super-nœud est égal à $(x.R)^2.V.\pi$.
   - H : Le nombre moyen de nœuds relayeurs pour atteindre la destination finale.
   - S : Nombre de message de données injectés dans le réseau. Pour simplifier le calcul, nous supposons que le nombre de messages est reparti équitablement entre les nœuds classiques et les super-nœuds.

2. **Coût de la phase de découverte de voisinage**
   Ici, nous calculons le nombre de messages envoyés et reçus pendant la phase de découverte de voisinage :

Nombre de messages envoyés
- N messages de découverte de voisinage *Hello_ Message*
- N messages d'annonce de voisin entendu *Heard_ Neighbor_ Message*

Nombre de messages reçus
- Chaque message de type *Hello_ Message* envoyé par un nœud classique sera reçu par $(R^2.V.\pi)$ nœuds. Puisqu'il y a A nœuds classiques, alors il y aura $((A/N).N)$ messages de type *Hello_ Message* envoyés qui généreront $(A/N).N.R^2.V.\pi$ réceptions de messages de type *Hello_ Message*.
- Chaque message de type *Heard_ Neighbor_ Message* envoyé par un nœud classique sera reçu par $(R^2.V.\pi)$ nœuds. Puisqu'il y a A nœuds classiques, alors il y aura $((A/N).N)$ messages de type *Heard_ Neighbor_ Message* envoyés qui généreront $(A/N).N.R^2.V.\pi$ réceptions de messages de type *Heard_ Neighbor_ Message*.
- Chaque message de type *Hello_ Message* envoyé par un super-nœud sera reçu par $((x.R)^2.V.\pi)$ nœuds. Puisqu'il y a B super-nœuds, alors il y aura $((B/N).N)$ messages de type *Hello_ Message* envoyés qui généreront $(B/N).N.(x.R)^2.V.\pi$ réceptions de messages de type *Hello_ Message*.
- Chaque message de type *Heard_ Neighbor_ Message* envoyé par un super-nœud sera reçu par $((x.R)^2.V.\pi)$ nœuds. Puisqu'il y a B super-nœuds, alors il y aura $((B/N).N)$ messages de type *Heard_ Neighbor_ Message* envoyés qui généreront $(B/N).N.(x.R)^2.V.\pi$ réceptions de messages de type *Heard_ Neighbor_ Message*.

Ainsi le nombre de messages émis et reçus pendant la phase de découverte de voisinage pour AsymRP est représenté par l'équation 6.3 :

$$\text{Coût\_découverte\_voisinage} = 2.(N + A.R^2.V.\pi + B.(x.R)^2.V.\pi) \tag{6.3}$$

3. **Coût de la phase de collecte de données**

Nous nous intéressons maintenant à la consommation énergétique de la phase de collecte de données. Nous calculons le nombre de messages émis et reçus pendant cette phase :

Nombre de messages envoyés
- S messages de données injectés seront envoyés dans le réseau.
- Pour les S messages de donnée envoyés, et avec un nombre moyen de relais égal à H, alors il y aura H.S messages relayés par les nœuds intermédiaires avant d'arriver au niveau de la destination finale.
- Dans le cas défavorable, chaque super-nœud relayeur va générer des messages ACKs explicites (comme dans l'exemple représenté dans la figure 6.7). Ainsi, dans ce cas, il y aura $3.(B/N).S.H$ messages ACKs transmis.
- Pour une propagation idéale, la destination finale envoie S messages pour acquitter la bonne réception des messages de données envoyés.

Nombre de messages reçus
- Les messages de données générés par les nœuds classiques sont au nombre de $(A/N).S$. Chaque message de données envoyé par un nœud classique sera reçu par $R^2.V.\pi$ nœuds. Ainsi tous les messages de données envoyés par les nœuds classiques vont générer $(A/N).S.R^2.V.\pi$ réceptions.

- Les messages de données générés par les super-nœuds sont au nombre de $(B/N).S$. Chaque message de données envoyé par un super-nœud sera reçu par $(x.R)^2.V.\pi$ nœuds. Ainsi tous les messages de données envoyés par les super-nœuds vont générer $\underline{(B/N).S.(x.R)^2.V.\pi}$ réceptions.
- Il y aura $(A/N).H.S$ messages de données relayés par des nœuds classiques. Ces messages relayés généreront $\underline{(A/N).H.S.R^2.V.\pi}$ réceptions.
- Il y aura $(B/N).H.S$ messages de données relayés par des super-nœuds. Ces messages relayés généreront $\underline{(B/N).H.S.(x.R)^2.V.\pi}$ réceptions.
- Dans le cas défavorable, les $3.(B/N).S.H$ messages ACKs explicite envoyés vont générer $\underline{3.(B/N).S.H.(x.R)^2.V.\pi}$ réceptions.
- Au meilleur des cas, il y a S messages d'acquittement envoyés par la destination finale, chaque message est reçu par $(x.R)^2.V.\pi$. Ainsi les messages ACK vont générer $\underline{S.(x.R)^2.V.\pi}$ réceptions.

D'où le coût de collecte de données (nombre de messages envoyés et reçus) pour la proposition AsymRP est représenté par l'équation 6.4 :

$$\begin{aligned} \text{Coût\_collecte\_donnée} =& (1+H).(S+(A/N).S.R^2.V.\pi) \\ &+ (1+H).(B/N).S.(x.R)^2.V.\pi) \\ &+ S.(1+(x.R)^2.V.\pi) \\ &+ 6.(B/N).S.H \end{aligned} \tag{6.4}$$

4. **Coût énergétique du protocole TRIF**

Ici, nous calculons le nombre de messages envoyés et reçus par le protocole TRIF avec les mêmes hypothèses de transmission utilisées pour le calcul du coût énergétique de la proposition AsymRP :

Nombre de messages envoyés

- Il y aura $(A/N).S$ messages de données générés par les nœuds classiques.
- Il y aura $(B/N).S$ messages de données générés par les super-nœuds. Puisque chaque super-nœud envoie « x »messages, alors le nombre de message envoyés par les super-nœuds sera $\underline{x.(B/N).S}$.
- Il y aura $(A/N).H.S$ messages de données relayés par des nœuds classiques.
- Il y aura $(B/N).H.S$ messages de données relayés par des super-nœuds. Puisqu'un super-nœud envoie « x »messages, alors le nombre de messages relayés par les super-nœuds est égal à $\underline{x.(B/N).H.S}$.

Nombre de messages reçus

- Les $(A/N).S$ messages envoyés par les nœuds classiques vont générer $\underline{(A/N).S.R^2.V.\pi}$ réceptions.
- Chacun des $(B/N).S$ super-nœuds générant un message de données envoie x messages qui génèrent $\underline{(B/N).S.\sum_{i=2}^{x}(i.R)^2.V.\pi}$ réceptions.
- Les $(A/N).H.S$ messages de donnée relayés par des nœuds classiques vont générer (A/N).H.S. R^2.V.π réceptions.
- Chaque $(B/N).H.S$ super-nœuds qui doit relayer un message de données, envoie « x »messages qui va générer $\underline{(B/N).H.S.\sum_{i=2}^{x}(i.R)^2.V.\pi}$ réceptions.

Ainsi le nombre total des messages envoyés et reçus pour le protocole TRIF est représenté dans l'équation 6.5 :

$$\begin{aligned} \text{Coût\_TRIF} =& (A/N).S.(1 + R.V * \pi + H + H.R.V.\pi) \\ &+ (B/N).x.S.(1 + H) \\ &+ (B/N).S.R^2.V.\pi.(x.(x+1).(2.x+1)/6) \\ &+ (B/N).S.H.R^2.V.\pi.(x.(x+1).(2.x+1)/6) \\ &+ S.(1 + V.(x.R)^2.\pi) \end{aligned} \tag{6.5}$$

C. Consommation énergétique de AsymRP

Ici, nous avons fixé le nombre de nœuds dans le réseau à mille ($N = 1000$) et le niveau de puissance de transmission des super-nœuds à 5 fois le niveau de puissance de transmission des nœuds classiques ($x = 5$). Suites aux expérimentations faites dans [153] pour le projet ARESA [2], le nombre de sauts moyens est fixé à 5 (H=5 sauts). Nous évaluons le nombre de messages envoyés et reçus pour un degré faible et élevé (4 et 20 voisins par nœud) dans un réseau. Figure 6.9(a) et la figure 6.9(b) représentent le nombre total de messages envoyés et reçus par AsymRP pour un réseau à faible densité et un réseau à haute densité, respectivement. Nous constatons que quelle que soit la densité du réseau, lorsque le nombre de messages de données dépasse 1/3 du nombre total des nœuds dans le réseau, le coût de la phase de découverte de voisinage est couvert par la phase de collecte des données. En augmentant le nombre de messages de données, le coût de la phase de découverte de voisinage dans un réseau statique est négligeable par rapport au coût de la phase de collecte de données.

D. Comparaison AsymRP et TRIF

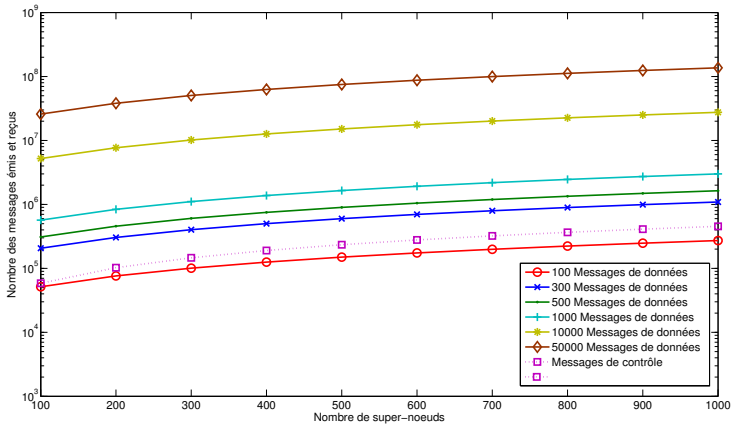
Ici nous comparons la quantité de messages envoyés et reçus pour AsymRP et TRIF dans les deux cas où la densité est faible et où elle est élevée. Figure 6.10(a) et Figure 6.10(b) représentent respectivement le nombre total de messages envoyés et reçus dans un réseau à faible et à haute densité. Dans les deux cas, AsymRP utilise moins de messages que TRIF et consomme donc moins d'énergie. Nous voyons aussi que la différence entre les deux courbes représentant AsymRP et TRIF augmente lorsque nous augmentons le nombre de message de données généré dans le réseau. AsymRP consomme moins d'énergie que TRIF car l'envoi d'un nouveau message de données peut générer au plus un envoi et $\underline{(x.R)^2.V.\pi}$ réceptions. Cependant, avec TRIF l'envoi d'un nouveau message de données peut générer au plus $\underline{\text{x}}$ envois et $\underline{\sum_{i=2}^{x}(i.R)^2.V.\pi}$ réceptions.

6.4.2 Etude par simulation

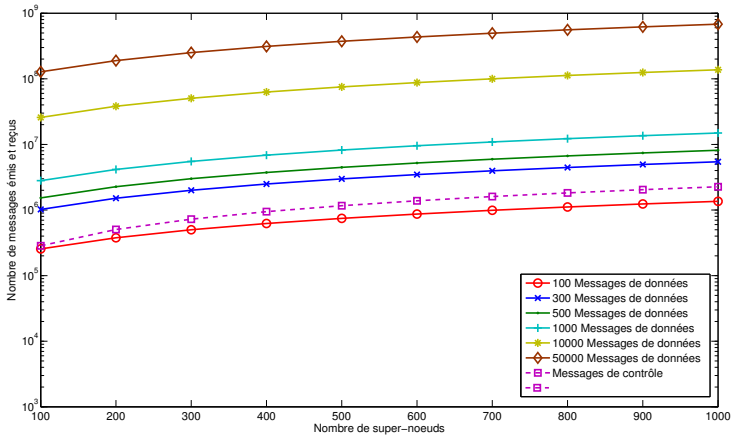
A. Paramètres de simulation

Nous considérons une topologie en grille régulière à maille carrée composée de 120 nœuds capteurs et un seul nœud puits situé au centre du réseau. Dans les simulations, nous supposons que les liens asymétriques sont causés par la présence de différentes portées de transmission des nœuds. Ainsi, nous supposons qu'il y a trois types de nœuds déployés dans chaque réseau : des nœuds classiques

2. ARESA : http ://aresa-project.insa-lyon.fr



(a) Densité faible



(b) Densité élevée

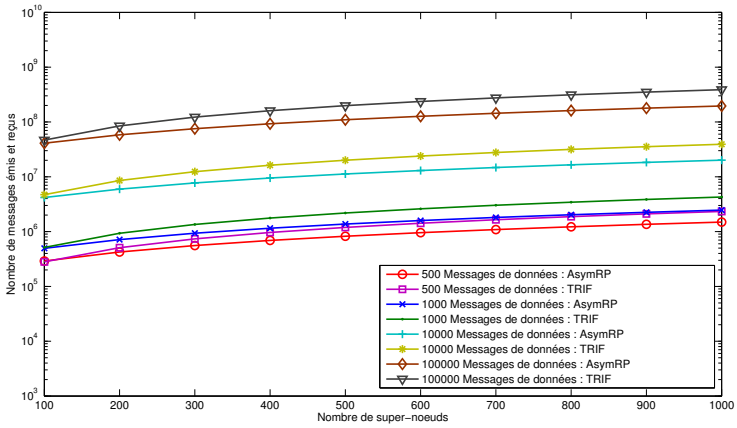
FIGURE 6.9: Coût de la phase de découverte de voisinage et de la phase de collecte de données pour AsymRP

ayant un niveau de portée de transmission unitaire et des super-nœuds ayant un niveau de portée de transmission égal à 3 et 6 fois celui des nœuds classiques.

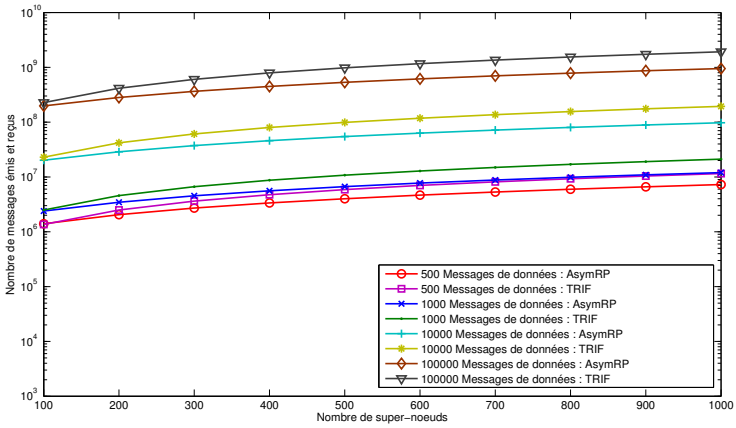
La couche MAC utilisée est le mode CSMA du standard 802.15.4. Chaque nœud est considéré comme fixe et son identité est déterminée par un identifiant unique.

Suivant une loi aléatoire et uniforme, nous choisissons les super-nœuds dans le réseau et nous déclenchons plusieurs évènements applicatifs qui seront détectés chacun par un seul nœud et remontés périodiquement vers le nœud puits situé au centre de la topologie. Les résultats sont représentés avec un intervalle de confiance de 95%.

Le tableau 6.2 résume les principales caractéristiques du réseau et les paramètres de simulation.



(a) Densité faible



(b) Densité élevée

FIGURE 6.10: Comparaison de la consommation énergétique AsymRP et TRIF

B. Taux de messages dupliqués

La figure 6.11 représente la quantité de messages dupliqués reçus par le nœud puits pour AsymRP et TRIF, quand il y a 50 évènements à reporter vers le nœud puits dans les différents scénarii : avec 10%, 30% et 50% de super-nœuds déployés aléatoirement dans le réseau. La figure 6.11, montre que le taux de messages dupliqués reçus avec AsymRP est significativement inférieur à celui du protocole TRIF, et ceci pour les trois scénarii. Nous constatons aussi que lorsque nous augmentons le nombre de super-nœuds déployés dans le réseau, le taux de messages dupliqués pour TRIF augmente aussi parce que chaque super-nœud envoie le message de données à plusieurs reprises avec des niveaux de portée de transmission décrémentés. Pour AsymRP, le taux de messages dupliqués reçu par le nœud de destination reste très faible par rapport à TRIF et il est aussi insensible au nombre des super-nœuds dans le réseau. Ceci est du au fait qu'avec AsymRP, chaque nœud envoie seulement

Paramètre	Valeur
Topologies	Grille régulière
Nombre de nœuds	121 nœuds
Portée des nœuds	1x, 3x et 6x portées des nœuds classiques
Nœuds source de données	10 .. 50
Protocole MAC	802.15.4 (CSMA)
Propagation	Modèle à deux rayons
Nombre de sauts maximum autorisé	16 sauts
Intervalle de confiance	95%
Simulateur	WSNet [67]

TABLE 6.2: Paramètres de simulation communs pour AsymRP et TRIF

un seul message en utilisant son propre niveau de portée de transmission. De plus, l'utilisation du message d'acquittement envoyé par le nœud puits évite la retransmission d'un même message de données par les autres nœuds en contention pour relayer ce même message.

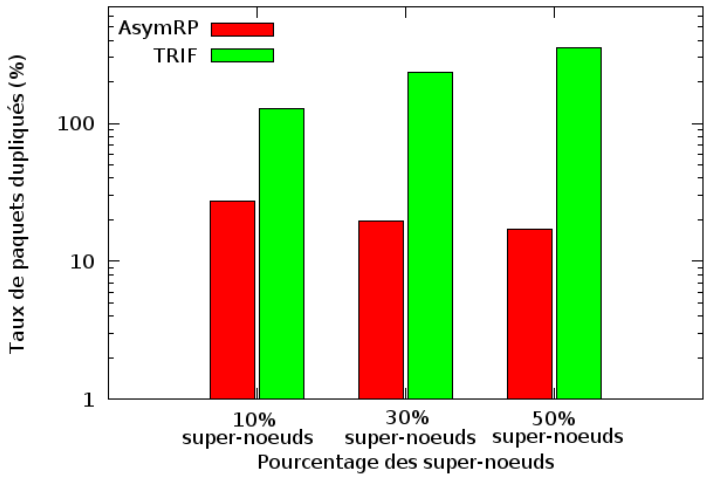


FIGURE 6.11: Comparaison du taux de messages dupliqués pour AsymRP et TRIF

C. Taux de livraison

Dans la figure 6.12, nous évaluons la distribution du taux de livraison de notre proposition AsymRP et TRIF en fonction du rang. La figure 6.12 montre que AsymRP surpasse les performances de TRIF. En effet, plus le chemin est long plus le taux de livraison diminue. De plus, puisque TRIF ne prend pas en compte les liens asymétriques, son taux de livraison atteint la valeur zéro pour les nœuds ayant un rang supérieur à 8 (voir figure 6.12). Nous remarquons aussi que, quand nous augmentons le nombre de liens asymétriques dans le réseau (en augmentant le nombre de super-nœuds), le taux de livraison de notre proposition AsymRP augmente. En effet, avec AsymRP, nous exploitons ces liens asymétriques, ce qui va réduire le nombre de sauts avant d'atteindre la destination finale. Nous allons vérifier cette induction dans le paragraphe suivant.

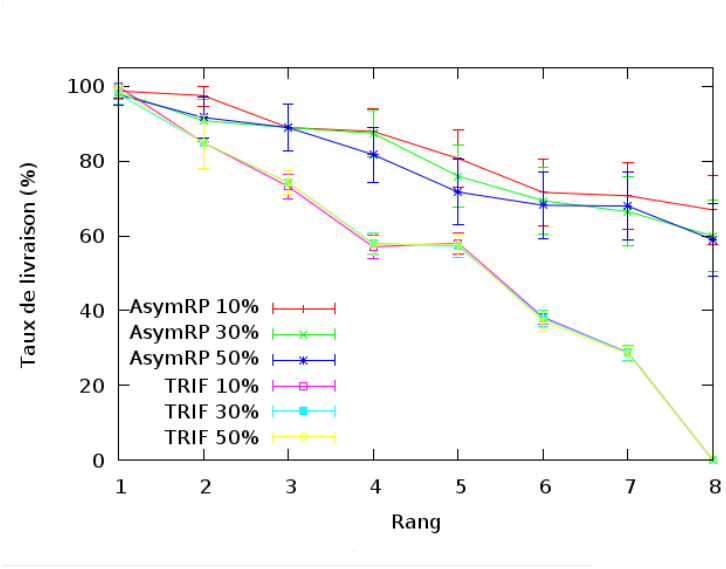


FIGURE 6.12: Comparaison de la distribution du taux de livraison pour AsymRP et TRIF

D. Nombre de sauts moyen

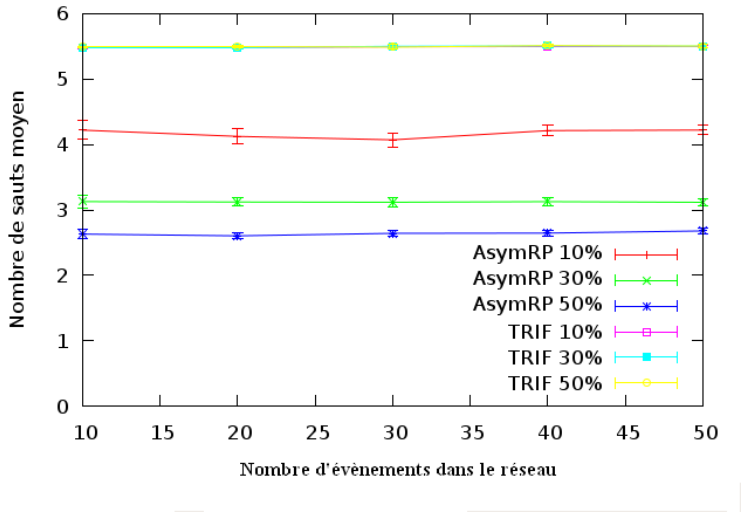
Enfin, nous évaluons le nombre de sauts moyen que fait un message de données avant d'atteindre la destination finale. La figure 6.13(a) représente la distribution du nombre de sauts moyen pour AsymRP et TRIF pour différents scénarii. Comme le montre la figure 6.13(a), un message de données avec AsymRP fait moins de sauts pour atteindre la destination finale qu'avec TRIF. Cela est dû au fait que, dans AsymRP, nous exploitons les liens asymétriques durant la phase de collecte des données. En effet, avec AsymRP, le message de données est transmis via des liens asymétriques ainsi le nombre de sauts de bout-en-bout est plus faible que celui offert par TRIF puisque, avec ce dernier, seulement les liens symétriques sont utilisés. Nous remarquons aussi, que lorsque nous augmentons le nombre de super-nœuds dans le réseau, le nombre de sauts avec AsymRP diminue.

Dans la figure 6.13(b), nous évaluons le pourcentage des liens asymétriques utilisés par notre proposition AsymRP. Le taux de liens asymétriques utilisés en moyenne pour un chemin varie entre 20% et 45%. Ce taux augmente quand nous augmentons le nombre de liens asymétriques présents dans le réseau.

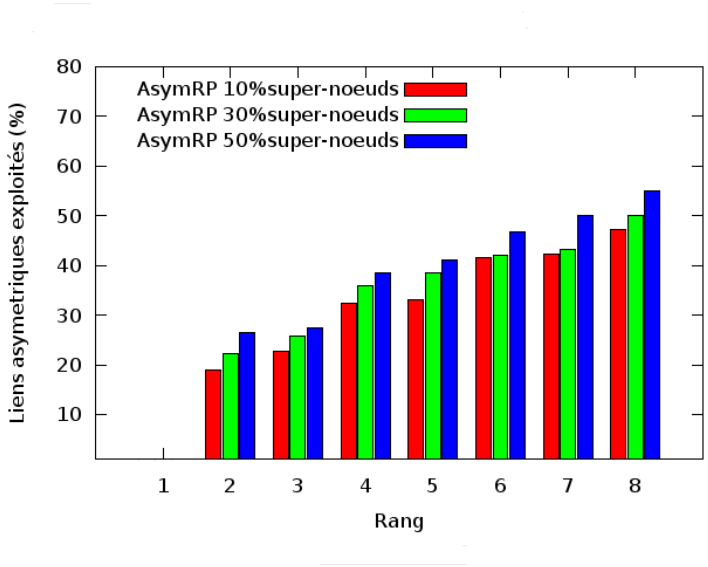
Ainsi, nous avons pu vérifier que notre proposition AsymRP exploite les liens asymétriques dans le réseau tout en augmentant le taux de livraison et en réduisant le nombre de sauts de bout-en-bout et le taux de messages dupliqués reçus au niveau du nœud puits.

6.5 Discussions

Dans cette section, nous discutons une extension possible de notre proposition AsymRP. Nous considérons le problème présenté dans la topologie de la figure 6.14. Même si cette topologie est connectée, le nœud E ne peut pas trouver un chemin pour envoyer un message ACK explicite vers le nœud source D. Dans ce cas particulier, le message de données est perdu quand il arrive au nœud D. Tous les messages de données se perdent au niveau de ce nœud comme si la topologie n'était pas connexe. Ici, nous proposons une extension de notre proposition AsymRP pour répondre à ce problème, appelée O-AsymRP (*Optimized AsymRP*). Cette extension permet de trouver un chemin



(a) Nombre de sauts



(b) Taux moyen de liens asymétriques dans un chemin

FIGURE 6.13: Evaluation du nombre de sauts pour AsymRP et TRIF et du taux des liens asymétriques exploités par AsymRP

pour acheminer un ACK explicite vers le nœud source.

6.5.1 Routage par la source pour le message ACK explicite : O-AsymRP

L'idée de cette extension est d'utiliser une technique basée sur un routage par la source pour envoyer le message ACK explicite. Une première variante peut être résumée comme suit : Chaque

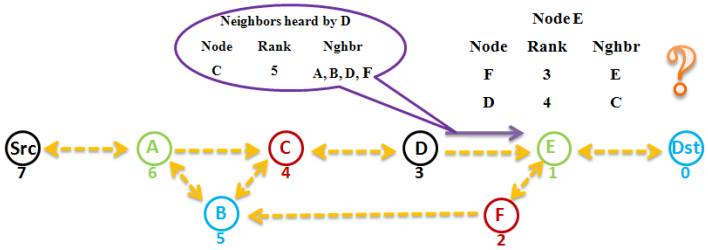


FIGURE 6.14: Problème d'acheminement de message ACK explicite

nœud candidat, ne trouvant pas un nœud en commun ou un nœud intermédiaire vers lequel il peut envoyer un ACK explicite, initie une découverte de chemin pour envoyer son message ACK explicite. Pour éviter l'inondation de cette découverte de route, nous proposons d'utiliser des techniques d'optimisation comme dans [116] ou [142].

Un exemple de cette première variante basée sur le routage par inondation de message ACK explicite est montré dans la figure 6.15. La figure 6.15 montre comment le nœud candidat *E* initie une recherche de chemin pour envoyer un ACK explicite au nœud *D*. Le message ACK trouve le chemin *F*-*B*-*C* jusqu'à ce qu'il atteigne la destination finale, le nœud *D*.

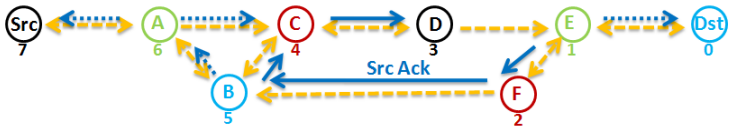


FIGURE 6.15: Première extension : routage par la source pour le message ACK explicite

Notre proposition, O-AsymRP, permet d'éviter l'inondation de messages ACKs explicites dans le réseau et permet aussi d'éviter l'ajout de nouvelles techniques couplées avec l'inondation qui pourront augmenter la complexité de notre proposition.

A. Description de O-AsymRP

Notre optimisation concerne le cas où un nœud *Candidat*, ne trouvant pas un nœuds *Commun* ou un nœud *Intermédiaire* pour envoyer un ACK explicite vers le nœud *Source*. Avec O-AsymRP, le nœud *Candidat* fait suivre le message de données et diffuse un ACK explicite contenant la liste des voisins du nœud *Source*. Les nœuds recevant ce message ACK diffusé vont appliquer l'algorithme décrit dans la figure 6.16. Chaque nœud recevant un ACK en diffusion commence par vérifier s'il peut trouver un nœud *Commun* ou *Intermédiaire* avec le nœud *Source*. Si un tel nœud existe alors ce nœud envoie cet ACK à son voisin *Commun* ou *Intermédiaire* qui le fera suivre vers le nœud *Source*. Si ce nœud en réception ne trouve pas un voisin qui pourra faire suivre le message ACK vers le nœud *Source*, alors il vérifie si le message ACK a atteint la valeur maximale du nombre de sauts autorisé. Si ce n'est pas le cas alors il diffuse ce message à son tour dans son voisinage tout en incrémentant le nombre de sauts parcourus par ce message ACK envoyé en diffusion. Dans le cas où le nombre de sauts autorisé est atteint, le nœud en réception supprime ce message ACK.

1. **Principe de O-AsymRP**
   A la réception d'un message de données, et n'ayant pas trouvé un nœud auquel il peut envoyer un ACK explicite, nous proposons, dans O-AsymRP, que ce *Candidat* fait suivre le message

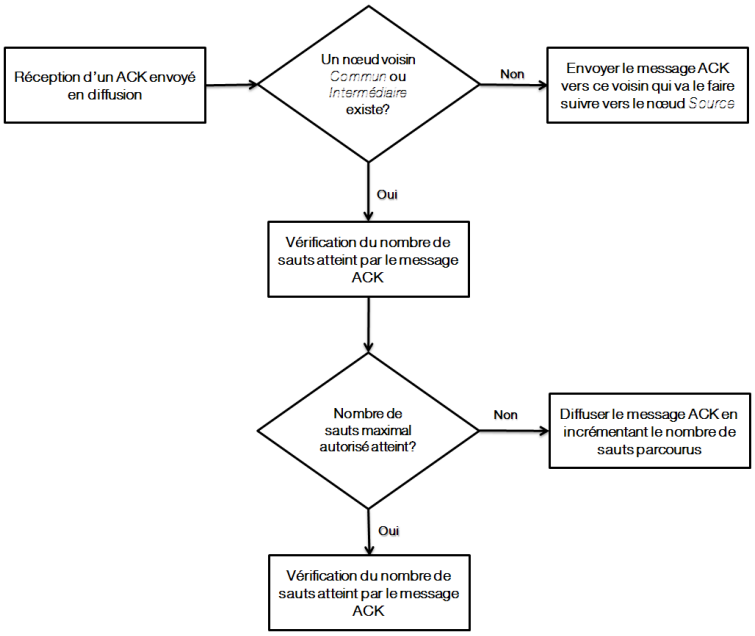


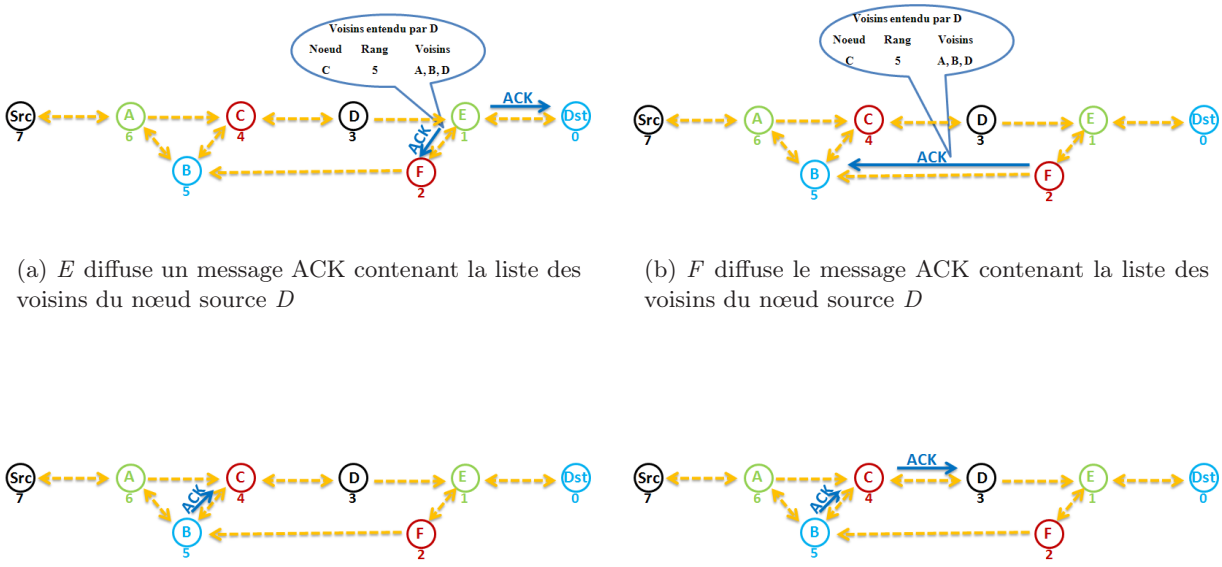
FIGURE 6.16: Principe d'acheminement de message ACK explicite avec O-AsymRP

de données comme décris précédemment pour AsymRP. Ce nœud *Candidat* diffuse aussi dans son voisinage un message ACK pour tenter de trouver un chemin retour vers le nœud *Source*. Dans l'entête de ce message ACK, le nœud *Candidat* ajoute la liste des voisins envoyée par le nœud *Source* dans son message de données. Comme c'est décrit dans l'algorithme de la figure 6.16, les voisins, du nœud *Candidat*, recevant le message ACK envoyé en diffusion commencent par vérifier s'ils ont un voisin *Commun* ou *Intermédiaire* avec le nœud *Source*. Chaque nœud peut vérifier s'il possède un tel voisin en se basant sur sa table de voisinage et sur la liste des voisins du nœud *Source* (envoyée par le nœud *Candidat* dans le message ACK). Si ce nœud *Commun* ou *Intermédiaire* existe, le nœud en réception envoie le message ACK à ce nœud *Commun* ou *Intermédiaire*. Dans le cas échéant, les voisins rediffusent le message ACK envoyé par le nœud *Candidat* si le nombre de sauts autorisé n'est pas atteint.

2. **Exemple**
   Prenons l'exemple de la figure 6.17. Le nœud *Candidat*, *E*, ne trouvant pas un nœud *Commun* ou *Intermédiaire* auquel il peut envoyer un ACK explicite à faire suivre au nœud *D*, diffuse dans son voisinage un message ACK contenant la liste des voisins du nœud *D* (voir figure 6.17(a)). Cet ACK diffusé par le nœud *E* sera reçu par *Dst* et *F*. Ces deux nœuds, ne trouvant pas un voisin auquel ils peuvent envoyer cet ACK, vont le diffuser chacun dans son voisinage (figure 6.17(b)). Ce message diffusé sera reçu par le nœud *B*. Ce dernier conclut à partir de sa table de voisinage et de la liste des voisins du nœud *D* envoyé dans le message ACK, qu'il possède un voisin *Commun* qui est le nœud *C*. Ainsi, *D* envoi un ACK au nœud *C* (voir figure 6.17(c)). Ce dernier fait suivre le message ACK vers la destination finale *D* (voir figure 6.17(d)).

B. Evaluation de performance

Dans cette section, nous évaluons les performances de l'extension O-AsymRP et nous la comparons à celles de la proposition de base.



(a) E diffuse un message ACK contenant la liste des voisins du nœud source D

(b) F diffuse le message ACK contenant la liste des voisins du nœud source D

(c) B envoie le message ACK explicite vers le nœud C

(d) C fait suivre le message ACK explicite vers le nœud source D

FIGURE 6.17: Principe du routage par la source d'un message ACK explicite avec O-AsymRP

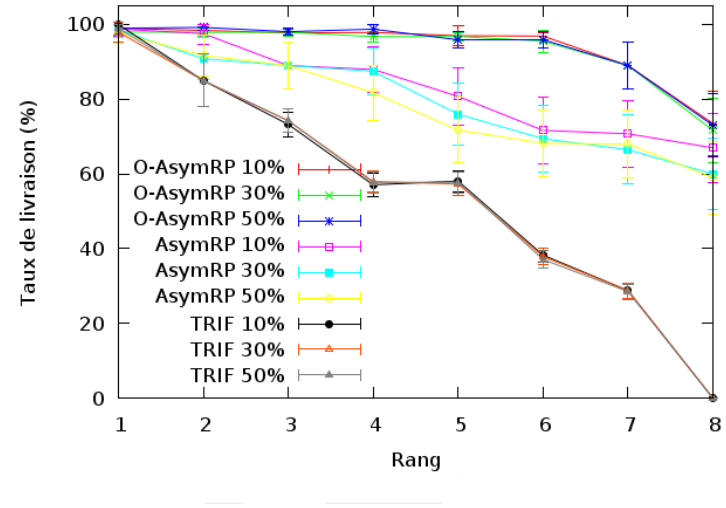
1. **Paramètres de simulation**
   Dans cette section, nous considérons une topologie en grille et un topologie aléatoire de 120 nœuds capteurs et un seul nœud puits situé au centre du réseau. Comme précédemment, dans les simulations, nous supposons que les liens asymétriques sont causés par la présence de différentes portées de transmission des nœuds.
   Le tableau 6.3 résume les principales caractéristiques du réseau et les paramètres de simulation.

Paramètre	Valeur
Topologies	Topologie en grille et topologie aléatoire
Nombre de nœuds	121 nœuds
Portée des nœuds	1x, 3x et 6x portées des nœuds classiques
Nœuds source de données	10 .. 50
Protocole MAC	802.15.4 (CSMA)
Propagation	Modèle à deux rayons
Nombre de sauts autorisés pour les ACKs	3 sauts
Intervalle de confiance	95%
Simulateur	WSNet [67]

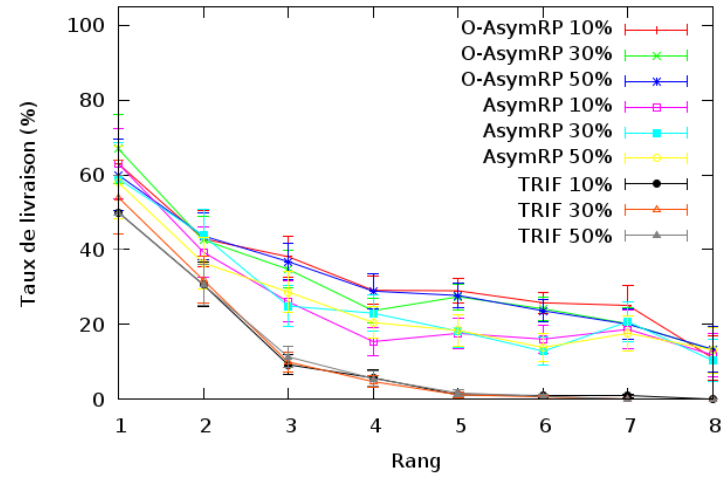
TABLE 6.3: Paramètres de simulation pour O-AsymRP et AsymRP

2. **Evaluation du taux de livraison**
   Dans cette section, nous évaluons le taux de livraison de O-AsymRP et nous le comparons à celui de la proposition de base AsymRP et de TRIF pour une topologie en grille et une topologie aléatoire. La figure 6.18 montre que le taux de livraison de O-AsymRP est dans les deux cas supérieur à celui de AsymRP et de TRIF. En effet, avec O-AsymRP, la phase

de routage de message ACK par la source permet d'augmenter le taux de livraison dans un réseau connexe.



(a) Topologie en grille

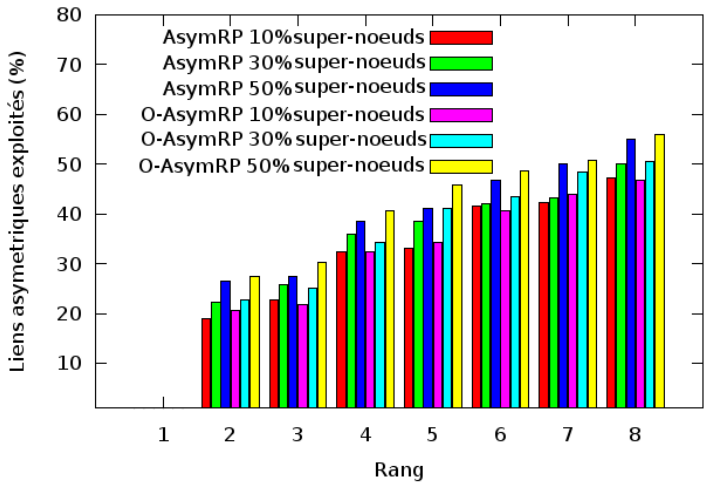


(b) Topologie aléatoire

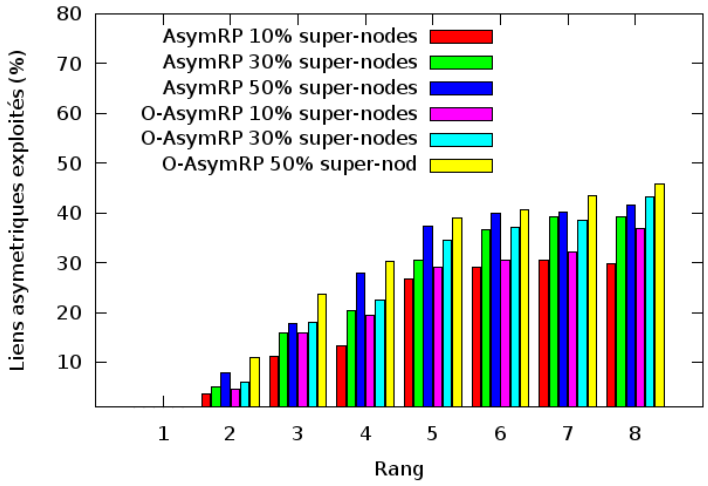
FIGURE 6.18: Evaluation du taux de livraison pour AsymRP, O-AsymRP et TRIF

3. **Evaluation des liens asymétriques exploités**
   Nous évaluons le pourcentage des liens asymétriques utilisés par notre proposition AsymRP et de l'optimisation O-AsymRP (voir figure 6.19). Le taux de liens asymétriques utilisés en moyenne augmente quand nous augmentons le nombre de liens asymétriques présents dans le réseau. En plus, le nombre de liens asymétriques exploités par O-AsymRP est légèrement su-

périeur à celui la proposition de base vu qu'avec cette extension, nous privilégions l'utilisation des liens asymétriques.



(a) Topologie en grille

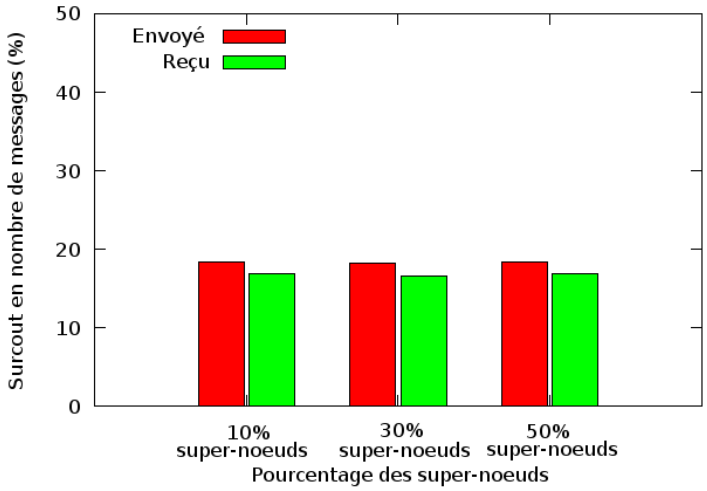


(b) Topologie aléatoire

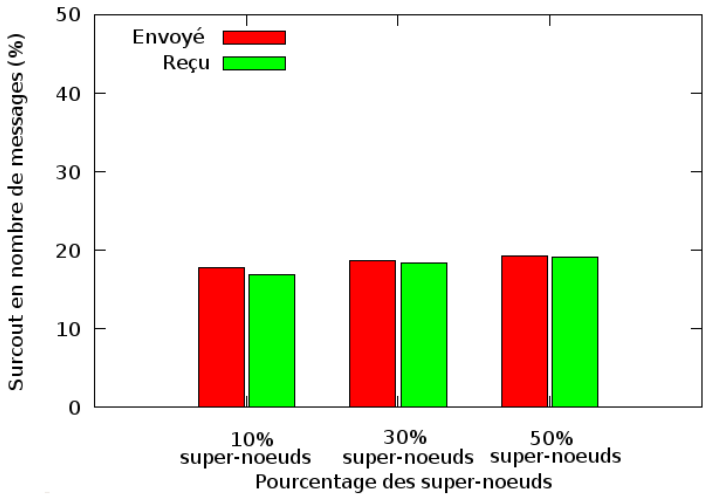
FIGURE 6.19: Evaluation du taux de liens asymétriques exploités pour AsymRP et O-AsymRP

4. **Evaluation du surcoût protocolaire**
   Nous finissons par évaluer, pour les deux topologies, le surcoût protocolaire introduit par O-AsymRP. Nous représentons ainsi dans la figure 6.20, le pourcentage des messages envoyés et reçu pour assurer l'acheminement des messages ACKs utilisés par O-AsymRP par rapport au nombre total de messages envoyés et reçus. La figure 6.20 montre que le surcoût des messages

envoyés et reçus ne dépasse pas les 20%. Ce surcoût dépendra du nombre de sauts autorisé pour les messages ACKs explicites envoyés en diffusion : plus le nombre de sauts autorisé est élevé plus le taux de livraison et le surcoût sont élevé et inversement.



(a) Topologie en grille



(b) Topologie aléatoire

FIGURE 6.20: Evaluation du surcoût protocolaire de O-AsymRP

6.6 Conclusion

Dans ce chapitre, nous avons proposé un algorithme de collecte de données appelé AsymRP. Notre proposition exploite les liens asymétriques pendant la phase de collecte de données.

Les simulations mettent en évidence que notre proposition répond aux exigences de fournir un taux de livraison élevé, un taux de messages dupliqués faible et un nombre de saut faible par rapport au protocole TRIF. Nous avons étudié et évalué la consommation d'énergie de la phase de découverte de voisinage et la phase de collecte de données. Notre proposition nécessite une connaissance du voisinage et il existe un compromis entre le coût de l'énergie pour se procurer cette connaissance et le coût de l'énergie pour la phase de collecte de données. En effet, pour les applications de collecte de données périodiques, le coût de la découverte de voisinage dans un réseau statique peut être négligeable par rapport au coût de l'envoi de données périodiques au nœud de destination. Par conséquent, nous avons comparé la consommation d'énergie de notre proposition AsymRP et de TRIF en calculant la quantité de messages envoyés et reçus. Il a été montré que AsymRP consomme moins d'énergie que TRIF dès que le nombre de message de données dépasse le 1/3 du nombre total des nœuds dans le réseau.

Nous avons fini ce chapitre avec une description d'une extension de AsymRP, appelée O-AsymRP. Cette extension a pour objectif d'assurer l'exploitation des liens asymétriques sur n'importe quelle topologie connectée. Cette extension permet d'améliorer le taux de livraison de la proposition de base mais elle introduit un coût supplémentaire occasionné par l'acheminement des messages ACKs explicites. Nous avons montré par simulation que le nombre de messages introduits par cette extension ne dépasse pas 20% pour les messages envoyés et reçus.

7 Conclusion

Sommaire

Les réseaux de capteurs et les réseaux de capteurs actionneurs ont un large potentiel avec diverses applications pratiques et utiles. Cependant, il y a encore beaucoup de problèmes qui doivent être abordés pour un fonctionnement efficace de ces réseaux dans des applications réelles. Parmi les problèmes fondamentaux et importants dans ces réseaux de capteurs et actionneurs, nous pouvons mentionner l'auto-organisation et le routage. Dans cette thèse, nous nous sommes intéressés aux réseaux urbains considérés par le projet ANR ARESA2 VERSO 2009-017 qui sont principalement des réseaux hétérogènes et dynamiques. L'hétérogénéité est causée par la coexistence des nœuds capteurs à faibles ressources et des nœuds actionneurs riches en ressources. Ces derniers devraient être utilisés de manière différenciée par le réseau. C'est dans ce contexte que se déroule cette thèse dans laquelle nous avons étudié des algorithmes d'auto-organisations et de routage s'appuyant sur l'hétérogénéité. Ces travaux ont fait l'objet de plusieurs publications qui sont présentés dans l'annexe Publications.

7.1 Bilan

7.1.1 Exploitation de l'hétérogénéité pour la structuration du réseau

Dans un premier lieu, nous nous sommes intéressés à la première problématique dans les réseaux de capteurs et actionneurs qui est l'auto-organisation dans un contexte hétérogène. Se basant sur l'idée que la plupart des protocoles d'auto-organisation existants ne sont pas conçus pour exploiter les ressources des nœuds riches (actionneurs) afin de réduire la charge de communication au niveau des nœuds à faibles ressources (capteurs), nous avons proposé, dans le chapitre IV, un protocole d'auto-organisation pour les réseaux hétérogènes WSANs profitant des ressources disponibles au niveau des nœuds actionneurs. Dans cette nouvelle proposition, appelée Far-Legos, les nœuds riches en ressources initient et construisent une topologie logique. Far-Legos permet de profiter de la puissance d'émission des nœuds actionneur pour apporter une information de gradient au niveau des nœuds capteurs. Cette information de gradient, appelé rang, permet de diminuer l'allongement des chemins dans les zones couvertes. En effet, nous avons montré, en utilisant le simulateur WSNET, que l'allongement des chemins de notre proposition par rapport à l'algorithme du plus court chemin ne dépasse pas les 8% sur des topologies en grille régulière et aléatoire. Nous avons montré par simulation aussi que notre proposition offre un délai de bout-en-bout faible et un taux de livraison élevé dans les petites et larges zones non-couvertes dans un réseau stable. Ensuite, nous avons présenté deux variantes de notre proposition pour répondre au problème de la transmission aléatoire durant la phase de collecte des données : Clustered-FAR et Oriented-FAR. Les deux variantes présentent des performances améliorées. Elles conduisent à des performances proches en termes de taux de livraison et le nombre de sauts.

7.1.2 Evitement de l'hétérogénéité pour la collecte de données

Dans un second lieu, nous nous sommes intéressés à la problématique de collecte de données dans un contexte hétérogène. En effet, la présence de différents types de nœuds avec différentes portées de transmission peut provoquer l'apparition de liens asymétriques. Ces derniers peuvent détériorer les performances des protocoles de routage qui ne tiennent pas compte de ce type de liens. Nous nous sommes intéressés au protocole RPL que nous avons analysé dans un contexte hétérogène pour introduire une nouvelle métrique de calcul de rang. Celle-ci sera utile pour détecter et éviter les liens asymétriques au niveau de la couche réseau. Nous avons présenté également une adaptation du protocole Legos, appelé A-Legos, pour tenir compte des liens asymétriques. Avec les résultats

de simulation, nous avons montré, dans le chapitre V, que les deux propositions offrent un taux de livraison élevé et que l'existence des liens asymétriques ne détériore pas leurs performances. Nous avons attesté aussi que l'adaptation A-RPL permet d'offrir un meilleur délai et un nombre de saut de bout-en-bout minimisé comparés à ceux offerts par l'adaptation A-Legos. Cependant, la proposition A-Legos consomme moins d'énergie. Ainsi, la proposition A-Legos peut être utilisée pour les applications critiques en énergie, où la consommation énergétique est de plus haute priorité. L'adaptation A-RPL est dédiée aux applications sensibles aux délais, où les messages applicatifs doivent arriver au plus vite à la destination finale tout en sacrifiant de l'énergie au niveau des nœuds capteurs.

7.1.3 Exploitation de l'hétérogénéité pour la collecte de données

Enfin, nous nous sommes intéressés à l'exploitation des liens asymétriques qui ouvrent de nouvelles opportunités pour améliorer les performances des protocoles de routage. En effet, nous avons proposé, dans le chapitre VI, un protocole de routage pour la collecte des données dédiés aux WSANs appelé AsymRP. AsymRP est un protocole de routage dédié au trafic de collecte de données basé sur une connaissance de voisinage à 2-sauts combinée avec l'utilisation des messages d'acquittements (ACKs) implicites et une technique de routage de messages ACKs explicites. Notre proposition tire profit des liens asymétriques, permet d'assurer un taux de livraison élevé tout en réduisant significativement le nombre de messages dupliqués et le nombre de sauts de bout-en-bout. Nos résultats de simulation montrent que notre proposition AsymRP surpasse nettement les protocoles de routage traditionnels lors de la présence de liens asymétriques dans le réseau. Nous avons proposé une extension de AsymRP, appelée O-AsymRP dans le même chapitre. Cette extension a pour objectif d'assurer l'exploitation des liens asymétriques sur n'importe quelle topologie connectée. Cette extension permet d'améliorer le taux de livraison de la proposition de base mais elle introduit un coût supplémentaire occasionné par l'acheminement des messages ACKs explicites. Nous avons montré par simulation que le nombre de messages introduits par cette extension ne dépasse pas 20% pour les messages envoyés et reçus.

7.2 Perspectives

Les thématiques que nous avons abordées dans cette thèse nous ouvrent plusieurs perspectives de recherche. Elles se situent dans l'extension des travaux réalisés que nous structurons dans les différents points suivants :

7.2.1 Stratégie de collaboration et de coordination entre les actionneurs

Comme nous n'avons pas traité les défis de coordination et de collaborations entre les nœuds actionneurs dans les travaux présentés dans cette thèse, nous proposons d'étudier des techniques permettant d'assurer la collaboration et la coordination entre les actionneurs. En effet, quand un événement est remonté par des nœuds capteurs, les actionneurs doivent collaborer afin de décider l'attribution des tâches : la tâche peut exiger un seul actionneur (comment le sélectionner ?) ou plusieurs actionneurs (comment décider le nombre optimum d'actionneurs pour effectuer l'action ?).

7.2.2 Stratégie de déploiement

Nous avons montré avec Far-Legos et ses deux optimisations, dans le chapitre IV, que dans les zones couvertes par les nœuds riches en ressources le taux de livraison est élevé et que le délai de bout-en-bout est faible. Nous envisageons la possibilité d'étudier les stratégies de déploiement pour les nœuds riches en ressource et de mener des expériences pratiques avec les réseaux de capteurs hétérogènes. L'objectif étant d'optimiser le déploiement des nœuds riche en ressource pour minimiser les zones non-couvertes par ces nœuds actionneurs tout en optimisant le nombre de ces nœuds riches en ressource à déployer dans le réseau.

7.2.3 Expérimentation

Durant nos travaux de thèse nous avons pu valider nos propositions avec le simulateur WSNET. Même si nous avons utilisé des paramètres de simulations proches de la réalité (avec une propagation réaliste, avec des interférences, etc.), les résultats obtenus peuvent varier avec un déploiement réel. De plus, une implémentation réelle nous permettra d'évaluer l'apport réel de nos propositions et en particulier la contribution de notre proposition AsymRP. En effet, par simulation, on a pu créer les liens asymétriques, pour valider AsymRP, en supposant qu'il y a différents types de nœuds avec des puissances de transmission différentes. Or comme nous l'avons mentionné dans le chapitre VI, ce type de liens peut être causé par l'environnement et le déploiement réel ou par l'utilisation de différentes antennes pour l'émission et la réception. Avec l'expérimentation, nous pouvons valider le comportement de notre proposition avec ces deux derniers facteurs favorisant l'existence des liens asymétriques.

7.2.4 Exploitation d'autres types d'hétérogénéité

Nous avons identifié plusieurs types d'hétérogénéité au niveau des réseaux de capteurs et actionneurs :

- L'hétérogénéité au niveau des capacités de calcul et de mémoire où certains nœuds ont des puissances de calcul et de mémoire supérieures à celles des nœuds classiques déployés dans le réseau.
- L'hétérogénéité au niveau des liens où certains liens sont des liens asymétriques et/ou sont des liens à longues distances.
- L'hétérogénéité au niveau de l'énergie où certains nœuds ont des ressources énergétiques illimitées.

Nous nous sommes intéressés à l'hétérogénéité au niveau des liens et aussi à l'hétérogénéité au niveau des puissances de transmission des nœuds constituant le réseau. Les autres types d'hétérogénéités ouvrent aussi des opportunités et méritent d'être étudiés. Les nœuds avec des ressources énergétiques illimitées et des capacités de calcul et de mémoire importantes doivent être plus sollicités. Ces nœuds riches en ressources peuvent jouer le rôle de points de collecte locale. Ils peuvent aussi jouer le rôle de nœuds assurant la structuration et la configuration des nœuds dans le voisinage. Ainsi les nœuds riches en ressources peuvent jouer le rôle de nœuds permettant de structurer et de configurer les nœuds dans leurs voisinages.

7.2.5 Routage

A part le trafic de dissémination (P2MP) et de collecte de données (MP2P), le trafic de type point-à-point (P2P) ouvre des nouvelles pistes de travail. Dans un réseau de capteurs et actionneurs, pour

ce dernier type de trafic, l'utilisation des tables de routage est envisageable au niveau des nœuds riches en ressources. En effet, il y a besoin d'une table de routage qui doit être construite au niveau chaque nœud et un mécanisme pour remplir ces routes afin d'assurer le trafic P2P. Ceci peut être accompli par les nœuds riches en ressource déployé dans le réseau. Ces nœuds peuvent jouer le rôle de serveur DHCP local (*Dynamic Host Configuration Protocol*). Ils peuvent affecter des adresses aux nœuds capteurs dans leur voisinage et en plus ils peuvent sauvegarder les routes menant vers ces nœuds. Pour assurer l'acheminement des données vers ces capteurs, les nœuds riches en ressource jouent aussi le rôle d'un serveur NAT local (*Network Adress Translation*). Ainsi quand un nœud veut communiquer avec un autre nœud du réseau, il envoie une requête vers le nœud riche en ressource auquel il est attaché. Ce dernier, cherche le chemin à partir de sa table de routage locale vers la destination finale.

7.2.6 Sécurité

Les réseaux urbains auxquels nous nous intéressons dans le projet ARESA2 peuvent manipuler des données sensibles et/ou confidentielles. Or, une caractéristique des réseaux de capteurs et actionneurs est qu'ils sont déployés dans des environnements ouverts. Cette caractéristique oblige l'utilisation des techniques pour garantir l'intégrité des données circulant dans le réseau. Cependant, les contraintes matérielles des nœuds constituant le réseau limitent les techniques de sécurité utilisées dans ces réseaux. En effet, les algorithmes de cryptographie traditionnels consomment beaucoup d'énergie et nécessitent une grande puissance de calcul. Ainsi, il faut s'appuyer sur les nœuds riches en ressource dans les réseaux de capteurs et actionneurs pour servir comme point d'exécution d'algorithme de sécurité complexe. Ces nœuds riches en ressources peuvent jouer le rôle de distributeur de clé de sécurité et peuvent aussi appliquer des algorithmes de cryptographie asymétrique.

Liste des publications

– **Conférences et workshops internationaux :**

1. *Exploiting Asymmetric Links in a Convergecast Routing Protocol for Wireless Sensor Networks*. Bilel Romdhani, Dominique Barthel and Fabrice Valois. The International Conference on Ad Hoc Networks and Wireless (AdHocNow'2012), Belgrade, Serbia.
2. *Routing for Data-Collection in Heterogeneous Wireless Sensor Networks*. Bilel Romdhani, Dominique Barthel and Fabrice Valois. The 73rd Vehicular Technology Conference (VTC'2011-Spring), Budapest, Hungary.
3. *Strategy of self-organization in Sensors and Actuators Networks* . Bilel Romdhani, Dominique Barthel and Fabrice Valois. The 6th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob'2010), Niagara Falls, Canada.

– **Conférences nationales et colloques :**

1. *Routage P2P dans les réseaux WSNs : De la recherche académique à la standardisation*. Bilel Romdhani, Dominique Barthel and Fabrice Valois. 3éme journées nationales IP Capteurs, Lyon, France, 2011.
2. *Stratégie d'auto-organisation pour les réseaux de capteurs et actionneurs*. Bilel Romdhani, Dominique Barthel and Fabrice Valois. Les Journés Doctorales en Informatique et Réseaux, Sophia Antipolis, France 2010.
3. *Stratégie d'auto-organisation pour les réseaux de capteurs et actionneurs*. Bilel Romdhani, Dominique Barthel and Fabrice Valois. Présentation d'un poster aux premières journées du projet SEMBA aux Balcons du Lac d'Annecy, France 2009

– **Rapports de recherche :**

1. *Routing for Data-Collection in Heterogeneous Wireless Sensor Networks*. Bilel Romdhani, Dominique Barthel and Fabrice Valois. Rapport de recherche INRIA RR-7586, March 2011.
2. *Self-organization in Sensor and Actuators Networks : Strategies and their optimizations*. Bilel Romdhani, Dominique Barthel and Fabrice Valois. Rapport de recherche INRIA RR-7440, October 2010.

Bibliographie

[1] A. Abbasi and M. Younis. A survey on clustering algorithms for wireless sensor networks. *Journal of Computer Communications, Special Issue on Network Coverage and Routing Schemes for Wireless Sensor Networks*, 30 :2826–2841, 2007.

[2] C. Adjih, P. Jacquet, and L. Viennot. Computing connected dominated sets with multipoint relays. *Ad Hoc & Sensor Wireless Networks*, 1(1-2) :27–39, 2005.

[3] A. Adya, P. Bahl, J. Padhye, A. Wolman, and L. Zhou. A multi-radio unification protocol for ieee 802.11 wireless networks. In *Proceeding of the 1st Annual International Conference on Broadband Networks, BROADNETS04*, pages 344–354, San jose, California, USA, 2004.

[4] K. Akkaya and M. Younis. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3 :325–349, 2005.

[5] K. Akkaya, M. Younis, and W. Youssef. Positioning of base stations in wireless sensor networks. *IEEE Communications Magazine*, 45(4) :96–102, 2007.

[6] I. F. Akyildiz and I. H. Kasimoglu. Wireless sensor and actor networks : research challenges. *Ad Hoc Networks*, 2(4) :351–367, 2004.

[7] I. F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks : a survey. *IEEE Computer Networks*, 47(4) :445–487, 2005.

[8] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks : a survey. *IEEE Computer Networks*, 38(4) :393–422, 2002.

[9] J. N. Al-Karaki and A. E. Kamal. Routing techniques in wireless sensor networks : a survey. *IEEE Wireless Communications*, 11(6) :6–28, 2004.

[10] C. Alippi, G. Anastasi, M. D. Francesco, and M. Roveri. Energy management in wireless sensor networks with energy-hungry sensors. *IEEE Organization*, 12(2) :16–23, 2009.

[11] I. Amadou and F. Valois. Performance evaluation of distributed self-organization protocols in wireless sensor networks. In *Proceedings of the 7th ACM workshop on Performance evaluation of wireless ad hoc, sensor and ubiquitous networks, PE-WASUN10*, pages 79–86, Bodrum, Turkey, 2010.

[12] I. Amadou and F. Valois. Pizza forwarding : A beaconless routing protocol designed for realistic radio assumptions. In *Proceedings of the 4th International Conference on Sensor Technologies and Applications, SENSORCOMM10*, pages 495–500, Venice, Italy, 2010.

[13] W. R. Ashby. Principles of the self-organizing dynamic system. In *Journal of General Psychology*, volume 37, pages 125–128, 1947.

[14] A. Awang, X. Lagrange, and D. Ros. Rssi-based forwarding for multihop wireless sensor networks. In *Proceedings of the 5th edition of the Eunice workshop, EUNICE09*, pages 138–147, Barcelona, Spain, 2009.

[15] S. Banerjee and S. Khuller. A clustering scheme for hierarchical control in multi-hop wireless networks. In *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM01*, pages 1028–1037, Alaska, USA, 2001.

[16] J. Bang-Jensen and G. Gutin. *Digraphs : theory, algorithms and applications*. Springer monographs in mathematics (2nd edition). Springer, 2008.

[17] R. Bellazreg, M. Hamdi, and N. Boudriga. On the impact of irregular radio propagation on coverage control and sleep scheduling in wireless sensor networks. In *Proceedings of the ACS/IEEE International Conference on Computer Systems and Applications, AICCSA10*, pages 1–8, Hammamet, Tunisia, 2010.

[18] F. Benbadis, T. Friedman, M. D. de Amorim, and S. Fdida. Gps-free-free positioning system for wireless sensor networks. In *Proceedings of the 2nd IFIP International Conference on Wireless and Optical Communications Networks, WOCN05*, pages 541–545, Dubai, United Arab Emirates, 2005.

[19] C. Berge. *Théorie des graphes et ses applications*. Collection Univesitaire des Mathématiques, Dunod, Paris, 1958.

[20] J. Blum, M. Ding, A. Thaeler, and X. Cheng. Connected dominating set in sensor networks and manets. In *Handbook of Combinatorial Optimization*, pages 329–369. Springer US, 2005.

[21] M. Blum, T. He, S. Son, and J. A. Stankovic. Igf : A state-free robust communication protocol for wireless sensor networks, 2003.

[22] P. Bose, P. Morin, I. Stojmenovic, and J. 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, DIALM99*, pages 48–55, New York, NY, USA, 1999.

[23] A. Boukerche, R. Werner Nelem Pazzi, and R. B. Araujo. Fault-tolerant wireless sensor network routing protocols for the supervision of context-aware physical environments. *Journal of Parallel and Distributed Computing*, 66(4) :586–599, 2006.

[24] A. Brandt, J. Buron, and G. Porcu. Home Automation Routing Requirements in Low-Power and Lossy Networks. RFC5826 (Proposed Standard), April 2010.

[25] S. Butenko, X. Cheng, D. Du, and P. M. Pardalos. On the construction of virtual backbone for ad hoc wireless network, 2003.

[26] S. Calomme and G. Leduc. Conception d'un protocole de contrôle de topologie pour les overlays construits sur des réseaux ad hoc. In *Colloque Francophone sur l'Ingénierie des Protocoles - CFIP 2006 Session 4 : Réseaux sans fil multi sauts*, Tozeur, Tunisia, 2006.

[27] M. Cardei, Xia. Cheng, Xiu. Cheng, and D. Du. Connected domination in multihop ad hoc wireless networks. In *Proceedings of the 6th International Conference on Computer Science and Informatics,CS&I02*, pages 251–255, Durham, NC, USA, 2002.

[28] J. Cartigny, F. Ingelrest, and D. Simplot-Ryl andI. Stojmenovic. Localized lmst and rng based minimum-energy broadcast protocols in ad hoc networks. *Ad Hoc Networks*, 3(1) :1–16, 2005.

[29] A. Caruso, S. Chessa, S. De, and A. Urpi. Gps free coordinate assignment and routing in wireless sensor networks. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM05*, pages 150–160, Miami, FL, USA, 2005.

[30] A. Cerpa, J. L. Wong, M. Potkonjak, and D. Estrin. Temporal properties of low power wireless links : modeling and implications on multi-hop routing. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing, MobiHoc05*, pages 414–425, Urbana-Champaign, IL, USA, 2005.

[31] I. Chakeres and C. Perkins. Dynamic manet on-demand (dymo) routing. Work in progress draft-ietf-manet-dymo-21, 2010.

[32] E. Chávez, N. Mitton, and H. Tejeda. Routing in wireless networks with position trees. In *Proceedings of the 6th international conference on Ad-hoc, mobile and wireless networks, ADHOC-NOW07*, pages 32–45, Morelia, Mexico, 2007.

[33] B. Chen, S. Hao, M. Zhang, M. C. Chan, and A. L. Ananda. Deal : discover and exploit asymmetric links in dense wireless sensor networks. In *Proceedings of the 6th Annual IEEE communications society conference on Sensor, Mesh and Ad Hoc Communications and Networks, SECON09*, pages 297–305, Rome, Italy, 2009.

[34] D. Chen, J. Deng, and P. K. Varshney. A state-free data delivery protocol for multihop wireless sensor networks. In *Proceedings of IEEE Wireless Communications and Networking Conference, WCNC05*, pages 1818–1823, New Orleans, Louisiana, USA, 2005.

[35] D. Chen and P. K. Varshney. On-demand geographic forwarding for data delivery in wireless sensor networks. *Computer Communications*, 30(14-15) :2954–2967, 2007.

[36] D. Chen and P. K. Varshney. A survey of void handling techniques for geographic routing in wireless networks. *IEEE Communications Surveys Tutorials*, 9(1) :50–67, 2007.

[37] T. Clausen and P. Jacquet. Optimized link state routing protocol (olsr), 2003.

[38] L. H. Correia, D. F. Macedo, A. L. Dos Santos, A. F. Loureiro, and J. M. Nogueira. Transmission power control techniques for wireless sensor networks. *Computer Networks : The International Journal of Computer and Telecommunications Networking*, 51 :4765–4779, 2007.

[39] R.K. Crane. *Propagation handbook for wireless communication system design.* Electrical engineering and applied signal processing series. CRC Press, 2003.

[40] F. Dai and J. Wu. An extended localized algorithm for connected dominating set formation in ad hoc wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 15(10) :908–920, 2004.

[41] K. Dasgupta, K. Kalpakis, and P. Namjoshi. An efficient clustering-based heuristic for data gathering and aggregation in sensor networks. In *Proceedings of IEEE Wireless Communications and Networking, WCNC 03*, pages 1948–1953, New Orleans, Louisiana, USA, 2003.

[42] G. Destino, J. Saloranta, and G. Abreu. Sensor localization in realistic environment using ultra-wideband radio. In *Proceeding of the 11th International Symposium on Wireless Personal Multimedia Communications, WPMC08*, pages 8–11, Lapland, Finland, 2008.

[43] I. Dietrich and F. Dressler. On the lifetime of wireless sensor networks. *ACM Transactions on Sensor Networks*, 5 :1–39, 2009.

[44] L. Doherty, K. S. J. Pister, and L. El Ghaoui. Convex position estimation in wireless sensor networks. In *Proceedings of the 20th Annual Joint Conference of Computer and Communications Societies, INFOCOM01*, pages 1655–1663, Alaska, USA, 2001.

[45] M. Dohler, T. Watteyne, T. Winter, and D. Barthel. Routing Requirements for Urban Low-Power and Lossy Networks. RFC5548 (Proposed Standard), May 2009.

[46] S. J. Douglas, A. Daniel, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of the 9th International Conference on Mobile Computing and Networking, MobiCom03*, pages 134–146, San Diego, California, USA, 2003.

[47] R. Draves, J. Padhye, and B. Zill. Comparison of routing metrics for static multi-hop wireless networks. In *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications, SIGCOMM04*, pages 133–144, Portland, Oregon, USA, 2004.

[48] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proceedings of the 10th annual international conference on Mobile computing and networking, MobiCom04*, pages 114–128, Philadelphia, PA, USA, 2004.

[49] F. Dressler. A study of self-organization mechanisms in ad hoc and sensor networks. *Computer Communications*, 31(13) :3018–3029, 2008.

[50] E. Ekici, G. Yaoyao, and D. Bozdag. Mobility-based communication in wireless sensor networks. *IEEE Communications Magazine*, 44(7) :56–62, 2006.

[51] E. H. Elhafsi, N. Mitton, and D. Simplot-Ryl. Cost over progress based energy efficient routing over virtual coordinates in wireless sensor networks. *Proceeding of the International Symposium on A World of Wireless, Mobile and Multimedia Networks, WOWMOM07*, pages 1–6, 2007.

[52] E. H. Elhafsi, N. Mitton, and D. Simplot-Ryl. End-to-end energy efficient geographic path discovery with guaranteed delivery in ad hoc and sensor networks. In *Proceedings of the 19th International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC08*, pages 1–5, Cannes, France, 2008.

[53] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica. Beacon vector routing : scalable point-to-point routing in wireless sensornets. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design and Implementation, NSDI05*, pages 329–342, Boston, Massachusetts, USA, 2005.

[54] H. Frey and I. Stojmenovic. *Geographic and Energy-Aware Routing in Sensor Networks*, chapter Chap 12. John Wiley & Sons, 2005.

[55] H. Fubler, J. Widmer, M. Kasemann, M. Mauve, and H. Hartenstein. Contention-based forwarding for mobile ad hoc networks. *Ad Hoc Networks*, 1(4) :351–369, 2003.

[56] R. K. Gabriel and R. R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18(3) :259–278, 1969.

[57] P. Galiotos. Security-aware topology control for wireless ad-hoc networks. In *Proceeding of the IEEE Global Telecommunications Conference, GLOBECOM08*, pages 1–6, New Orleans, LA, USA, 2008.

[58] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. Complex behavior at scale : An experimental study of low-power wireless sensor networks. Technical report, UCLA Computer Science Department, 2002.

[59] P. Ghaffariyan. An effective data aggregation mechanism for wireless sensor networks. In *Proceedings of the 6th International Conference on Wireless Communications Networking and Mobile Computing, WiCOM10*, pages 1–4, Wuhan, China, 2010.

[60] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection tree protocol. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, SenSys09*, pages 1–14, Berkeley, California, 2009.

[61] J. Gomez and A. T. Campbell. A case for variable-range transmission power control in wireless multihop networks. In *Proceedings of 23rd Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM04*, pages 1425–1436, Hong Kong, China, 2004.

[62] M. Goyal. Reactive discovery of point-to-point routes in low power and lossy networks. Draft, IETF-ROLL, 2011.

[63] M. Goyal, E. Baccelli, A. Brandt, R. Cragie, and J. Martocci. Reactive discovery of point-to-point routes in low power and lossy networks, May 2011.

[64] J. Guerin, M. Portmann, and A. Pirzada. Routing metrics for multi-radio wireless mesh networks. In *Proceeding of Telecommunication Networks and Applications Conference, ATNAC07*, pages 343–348, Christchurch, New Zealand, 2007.

[65] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2) :388–404, 2000.

[66] M. Hamdi, N. Essaddi, and N. Boudriga. Energy-efficient routing in wireless sensor networks using probabilistic strategies. In *Proceeding of the Wireless Communications and Networking Conference, WCNC08*, pages 2567–2572, Las Vegas, USA, 2008.

[67] E. Ben Hamida, G. Chelius, and J. M. Gorce. Scalable versus accurate physical layer modeling in wireless network simulations. In *Proceedings of the 22nd Workshop on Principles of Advanced and Distributed Simulation, PADS08*, pages 127–134, Roma, Italy, 2008.

[68] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan. An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, 1(4) :660–670, 2002.

[69] M. Heissenbüttel, T. Braun, T. Bernoulli, and M. Wälchli. Blr : beacon-less routing algorithm for mobile ad hoc networks. *Computer Communications*, 27(11) :1076–1086, 2004.

[70] K. Heurtefeux and F. Valois. Distributed qualitative localization for wireless sensor networks. In *Proceedings of the 7th international conference on Ad-hoc, Mobile and Wireless Networks, ADHOC-NOW08*, pages 218–229, Sophia-Antipolis, France, 2008.

[71] K. Heurtefeux and F. Valois. Distributed localization protocol for routing in a noisy wireless sensor network. In *International Workshop on Localized Communication and Topology Protocols for Ad hoc Networks, LOCAN09*, pages 223–230, Wu Yi Shan, China, 2009.

[72] K. Heurtefeux and F. Valois. Is rssi a good choice for localization in wireless sensor network ? In *Proceedings of the 26th IEEE International Conference on Advanced Information Networking and Applications, AINA12*, pages 732–739, Fukuoka, Japan, 2012.

[73] Y. T. Hou, Y. Shi, H. D. Sherali, and S. F. Midkiff. On energy provisioning and relay node placement for wireless sensor networks. *IEEE Transactions on Wireless Communications*, 4(5) :2579–2590, 2005.

[74] F. Ingelrest, G. Barrenetxea, and M. Vetterli. SensorScope, un système clef en main de surveillance de l'environnement. In *Colloque Francophone sur l'Ingénierie des Protocoles (CFIP)*, Les Arcs, France, 2008.

[75] F. Ingelrest, D. Simplot-ryl, and I. Stojmenovic. A dominating sets and target radius based localized activity scheduling and minimum energy broadcast protocol for ad hoc and sensor networks. In *Prococeedings of the Mediterranean Ad Hoc Networking Workshop, Med-Hoc-Net 04*, pages 351–359, Bodrum, Turkey, 2004.

[76] F. Ingelrest, D. Simplot-ryl, and I. Stojmenovic. Routing and broadcasting in hybrid ad hoc and sensor networks. In *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks*, 2006.

[77] D. B. Johnson, D. A. Maltz, and J. Broch. Ad hoc networking. chapter DSR : the dynamic source routing protocol for multihop wireless ad hoc networks, pages 139–172. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[78] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next Century Challenges : Mobile Networking for "Smart Dust". In *Proceedings of the International Conference on Mobile Computing and Networking, MOBICOM99*, pages 271–278, Seattle, Washington, USA, 1999.

[79] B. Karp and H. T. Kung. Gpsr : greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking, MobiCom00*, pages 243–254, Boston, MA, USA, 2000.

[80] V. Katiyar, N. Chand, and S. Soni. Clustering algorithms for heterogeneous wireless sensor network : A survey. *International Journal Advanced Networking and Applications*, 2 :745–754, 2011.

[81] V. Kawadia and P.R. Kumar. Power control and clustering in ad hoc networks. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications, INFOCOM03*, pages 459–469, San Francisco, USA, 2003.

[82] K. Kim and K. G. Shin. On accurate measurement of link quality in multi-hop wireless mesh networks. In *Proceedings of the 12th annual international conference on Mobile computing and networking, MobiCom06*, pages 38–49, Los Angeles, CA, USA, 2006.

[83] Y. Kim, R. Govindan, B. Karp, and S. Shenker. Geographic routing made practical. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design and Implementation, NSDI05*, pages 217–230, Boston, MA, USA, 2005.

[84] L. M. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. *Theoretical Computer Science*, 243 :289–305, 2000.

[85] Y. Ko, S. Lee, and J. Lee. Ad hoc routing with early unidirectionality detection and avoidance,personal wireless communications. In *Prococeedings of International Conference on Personal Wireless Communications , PWC04*, pages 132–146, Delft, The Netherlands, 2004.

[86] P. Kulakowski, E. Calle, and J. L. Marzo. Sensors-actuators cooperation in wsans for firefighting applications. In *Proceedings of the 6th International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob10*, pages 726–732, Niagara Falls, Canada, 2010.

[87] Q. Lampin, D. Barthel, and F. Valois. Efficient route redundancy in dag-based wireless sensor networks. In *IEEE Wireless Communications and Networking Conference, WCNC10*, pages 1–6, Sydney, Australia, 2010.

[88] T. Le, P. Sinha, and D. Xuan. Turning heterogeneity into an advantage in wireless ad-hoc network routing. *Ad Hoc Network*, 8 :108–118, 2010.

[89] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle : a self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation, NSDI04*, pages 15–28, San Francisco, California, 2004.

[90] P. Levis, A. Tavakoli, and S. Dawson-Haggerty. Overview of existing routing protocols for low power and lossy networks. Draft, IETF-ROLL, 2009.

[91] N. Li and J. C. Hou. Topology control in heterogeneous wireless networks : problems and solutions. In *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM04*, pages 232–243, Hong Kong, China, 2004.

[92] N. Li, J. C. Hou, and L. Sha. Design and analysis of an mst-based topology control algorithm. *IEEE Transactions on Wireless Communications*, 4(3) :1195–1206, 2005.

[93] H. Liu, A. Nayak, and I. Stojmenovic. *Energy Efficient Backbones and Broadcasting in Sensor and Actuator Networks*, chapter Chap 2. John Wiley & Sons, 2010.

[94] T. Liu and W. Liao. Capacity-aware routing in multi-channel multi-rate wireless mesh networks. In *Proceeding of the IEEE International Conference on Communications, ICC06*, pages 1971–1976, Istanbul, Turkey, 2006.

[95] J. L. Lu. *Impacts of Self-organized Mechanisms in Wireless Sensors Networks*. PhD thesis, Ph.D. dissertation, INSA Lyon, May 2008.

[96] J. L. Lu and F. Valois. On the data dissemination in wsns. In *Proceedings of the 3rd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, WiMOB07*, pages 58–63, White Plains, New York, USA, 2007.

[97] J. L. Lu, F. Valois, and D. Barthel. Low-energy self-organization scheme for wireless ad hoc sensor networks. In *Proceedings of the 4th Annual Conference on Wireless on Demand Network Systems and Services, WONS07*, pages 138–145, Obergurgl, Tyrol, Austria, 2007.

[98] G. Malkin. Rip version 2. RFC2453 (Proposed Standard), 1998.

[99] A. Manjeshwar and D. P. Agrawal. Teen : a routing protocol for enhanced efficiency in wireless sensor networks. In *Proceedings of the 15th International Parallel and Distributed Processing Symposium, IPDPS01*, pages 2009–2015, San Francisco, USA, 2001.

[100] A. Manjeshwar and D. P. Agrawal. Apteen : A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. In *Proceedings of the 16th International Parallel and Distributed Processing Symposium, IPDPS02*, pages 195–202, Fort Lauderdale, Florida, USA, 2002.

[101] M. K. Marina and S. R. Das. Routing performance in the presence of unidirectional links in multihop wireless networks. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing, MobiHoc02*, pages 12–23, Lausanne, Switzerland, 2002.

[102] D. Martinez, F. Blanes, J. Simo, and A. Crespo. Wireless sensor and actuator networks : Charecterization and case study for confined spaces healthcare applications. In *International Multiconference on Computer Science and Information Technology, IMCSIT08*, pages 687–693, Wisla, Poland, 2008.

[103] J. Martocci, P. De Mil, N. Riou, and W. Vermeylen. Building Automation Routing Requirements in Low-Power and Lossy Networks. RFC5867 (Proposed Standard), April 2010.

[104] T. Melodia, D. Pompili, V. C. Gungor, and I. F. Akyildiz. Communication and coordination in wireless sensor and actor networks. *IEEE Transactions on Mobile Computing*, 6 :1116–1129, 2007.

[105] V. Mhatre and C. Rosenberg. Homogeneous vs heterogeneous clustered sensor networks : a comparative study. In *Proceedings of the IEEE International Conference on Communications, ICC04*, pages 3646–3651, Paris, France, 2004.

[106] N. Mitton, T. Razafindralambo, D. Simplot-Ryl, and I. Stojmenovic. Hector is an energy efficient tree-based optimized routing protocol for wireless networks. In *Proceedings of the 4th International Conference on Mobile Ad-hoc and Sensor Networks, MSN08*, pages 31–38, Wuhan, China, 2008.

[107] J. P. Monks and V. Bharghavanand W. M. W. Hwu. Transmission power control for multiple access wireless packet networks. In *Proceedings of the 25th Annual IEEE Conference on Local Computer Networks, LCN00*, pages 12–21, Tampa, Florida, USA, 2000.

[108] J. Moy. Ospf version 2. RFC2328 (Proposed Standard), 1998.

[109] S. Narayanaswamy, V. Kawadia, R. S. Sreenivas, and P. R. Kumar. Power control in ad-hoc networks : Theory, architecture, algorithm and implementation of the compow protocol. In *Proceedings of European Wireless Conference, EW02*, pages 156–162, Florence, Italy, 2002.

[110] S. Nesargi and R. Prakash. A tunneling approach to routing with unidirectional links in mobile ad-hoc networks. In *Proceedings of the 9th International Conference on Computer Communications and Networks, ICCCN00*, pages 522–527, Las Vegas, NV, USA, 2000.

[111] D. Niculescu and N. Badri. Ad hoc positioning system (aps) using aoa. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM03*, pages 1734–1743, San Francisco, USA, 2003.

[112] R. Ogier, F. Templin, and M. Lewis. Topology dissemination based on reverse-path forwarding (tbrpf), 2004.

[113] J. Park and S. Sahni. Power assignment for symmetric communication in wireless sensor networks. *International Journal of Distributed Sensor Networks*, 5 :185–200, 2009.

[114] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal. Locating the nodes : cooperative localization in wireless sensor networks. *IEEE Signal Processing Magazine*, 22(4) :54–69, 2005.

[115] N. Patwari, A. O. Hero III, M. Perkins, N. S. Correal, and R. J. O'Dea. Relative location estimation in wireless sensor networks. *IEEE Transactions on Signal Processing*, 51(8) :2137–2148, 2003.

[116] W. Peng and X. C. Lu. On the reduction of broadcast redundancy in mobile ad hoc networks. In *Proceedings of the 1st ACM international symposium on Mobile ad hoc networking & computing, MobiHoc00*, pages 129–130, Boston, Massachusetts, USA, 2000.

[117] W. Peng-Jun, K. M. Alzoubi, and O. Frieder. Distributed construction of connected dominating set in wireless ad hoc networks. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM02*, pages 1597–1604, New York, USA, 2002.

[118] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing, 2003.

[119] K. Pister and T. Phinney P. Thubert. Industrial Routing Requirements in Low-Power and Lossy Networks. RFC5673 (Proposed Standard), October 2009.

[120] S. Plancoulaine, A. Bachir, and D. Barthel. Wsn node energy dissipation, July 2006. France Telecom R&D, Technical Report.

[121] C. Prehofer and C. Bettstetter. Self-organization in communication networks : principles and design paradigms. *IEEE Communications Magazine*, 43(7) :78–85, 2005.

[122] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint relaying for flooding broadcast messages in mobile wireless networks. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences, HICSS02*, pages 3866–3875, Hawaii, USA, 2002.

[123] C. Qing and T. Abdelzaher. A scalable logical coordinates framework for routing in wireless sensor networks. In *Proceedings of the 25th International Real-Time Systems Symposium, RTSS04*, pages 349–358, Miami, FL, USA, 2004.

[124] V. Raghunathan, C. Schurgers, S. Park, M. Srivastava, and B. Shaw. Energy-aware wireless microsensor networks. 19 :40–50, 2002.

[125] R. Ramanathan and R. Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In *Proceedings of the 9th Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM00*, pages 404–413, Tel-Aviv, Israel, 2000.

[126] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *Proceedings of the 9th annual international conference on Mobile computing and networking, MobiCom03*, pages 96–108, San Diego, CA, USA, 2003.

[127] B. Romdhani, D. Barthel, and F. Valois. Strategy of self-organization in sensors and actuators networks. In *Proceeding of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob10*, pages 414–420, Niagara Falls, Canada, 2010.

[128] M. Rossi, M. Zorzi, and R. R. Rao. Cost efficient routing strategies over virtual coordinates for wireless sensor networks. In *Proceeding of the Global Telecommunications Conference, GLOBECOM05*, pages 2980–2986, Saint Louis, MO, USA, 2005.

[129] S. Roundy, E. S. Leland, J. Baker, E. Carleton, E. Reilly, E. Lai, B. Otis, J. M. Rabaey, P. K. Wright, and V. Sundararajan. Improving power output for vibration-based energy scavengers. *Pervasive Computing*, 4(1) :28–36, 2005.

[130] J. Sanchez, P. Ruiz, and R. Marin-Perez. Beacon-less geographic routing made practical : challenges, design guidelines, and protocols. *IEEE Communications Magazine*, 47(8) :85–91, 2009.

[131] J. A. Sanchez, R. Marin-Perez, and P. M. Ruiz. Boss : Beacon-less on demand strategy for geographic routing inwireless sensor networks. In *Proceedings of the 4th IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems, MASS07*, pages 1–10, Pisa, Italy, 2007.

[132] L. Sang, A. Arora, and H. Zhang. On exploiting asymmetric wireless links via one-way estimation. In *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing, MobiHoc07*, pages 11–21, Montreal, Quebec, Canada, 2007.

[133] C. Schurgers and M. B. Srivastava. Energy efficient routing in wireless sensor networks. In *Proceedings of the IEEE Military Communications Conference, MILCOM01*, pages 357–361, McLean, VA, USA, 2001.

[134] K. Seada, M. Zuniga, A. Helmy, and B. Krishnamachari. Energy-efficient forwarding strategies for geographic routing in lossy wireless sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems, SenSys04*, pages 108–121, Baltimore, MD, USA, 2004.

[135] R. C. Shah and J. M. Rabaey. Energy aware routing for low energy ad hoc sensor networks. In *Proceedings of the Wireless Communications and Networking Conference, WCNC2002*, pages 350–355, Orlando, FL, USA, 2002.

[136] V. Shah and S. Krishnamurthy. Handling asymmetry in power heterogeneous ad hoc networks : A cross layer approach. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems, ICDCS05*, pages 749–759, Columbus, Ohio, USA, 2005.

[137] IEEE Computer Society. Part15.4 : Wireless medium access control (mac) and physical layer (phy) specifications for low rate wireless personal area networks (wpans) amendment 1 : Add alternate phys. 2007.

[138] G. De Sousa, J. P. Chanet, A. Jacquot, D. Boffety, G. André, and K. M. Hou. Data collection and management solution for wireless sensor networks. In *The International Conference on Agricultural Engineering, AgEng10*, page 2010, Clermont Ferrand, France, 2010.

[139] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis. An empirical study of low-power wireless. *ACM Transaction Sensors Network*, 6 :1–49, 2010.

[140] S. Staab, F. Heylighen, C. Gershenson, G W. Flake, D. M. Pennock, D. C. Fain, D. De Roure, K. Aberer, W. M. Shen, O. Dousse, and P. Thiran. Neurons, viscose fluids, freshwater polyp hydra-and self-organizing information systems. *IEEE Intelligent Systems*, 18 :72–86, 2003.

[141] I. Stojmenovic. Energy conservation in sensor and sensor actuator networks. In Shih Lin Wu Yu Chee Tseng, editor, *Wireless Ad Hoc Networking : Personal Area, Local Area, and Sensory Area Networks*, pages 107–133, 2007.

[142] I. Stojmenovic, M. Seddigh, and J. Zunic. Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks. *IEEE Transactions on Parallel and Distributed Systems,*, 13(1) :14–25, 2002.

[143] F. Theoleyre and F. Valois. About the self-stabilization of a virtual topology for self-organization in ad hoc networks. In Sébastien Tixeuil and Ted Herman, editors, *Self-Stabilizing Systems*, volume 3764 of *Lecture Notes in Computer Science*, pages 214–228. Springer Berlin Heidelberg, 2005.

[144] F. Theoleyre and F. Valois. A self-organization structure for hybrid networks. *Ad Hoc Network*, 6 :393–407, 2008.

[145] P. Thubert. Rpl objective function zero. Draft, IETF, 2011.

[146] G. Tolle and D. Culler. Design of an application-cooperative management system for wireless sensor networks. In *Proceeedings of the 2nd European Workshop on Wireless Sensor Networks, EWSN05*, pages 121–132, Istanbul, Turkey, 2005.

[147] G. T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12(4) :261–268, 1980.

[148] Y. C. Tseng, S. Y. Ni, Y. S. Chen, and J. P. Sheu. The broadcast storm problem in a mobile ad hoc network. *Wireless Network*, 8(2/3) :153–167, 2002.

[149] JP. Vasseur, M. Kim, K. Pister, N. Dejean, and D. Barthel. Routing metrics used for path calculation in low power and lossy networks. Draft, IETF, 2011.

[150] R. Verdone, D. Dardari, G. Mazzini, and A. Conti. *Wireless Sensor and Actuator Networks : Technologies, Analysis and Design*. Academic Press, 2008.

[151] A. C. Viana, M. D. de Amorim, S. Fdida, and J. F. de Rezende. Self-organization in spontaneous networks : the approach of dht-based routing protocols. *Ad Hoc Network*, 3 :589–606, 2005.

[152] T. Watteyne, I. Auge-Blum, M. Dohler, and D. Barthel. Geographic forwarding in wireless sensor networks with loose position-awareness. In *Proceedings of the 18th International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC07*, pages 1–5, Athens, Greece, 2007.

[153] T. Watteyne, D. Barthel, M. Dohler, and I. Auge-Blum. Sense and sensitivity : a large scale experimental study of reactive gradient routing. *Measurement Science and Technology, Special issue on Internet of Things*, 21(12) :pp9, 2010.

[154] T. Watteyne, A. Molinaro, M. Richichi, and M. Dohler. From manet to ietf roll standardization : A paradigm shift in wsn routing protocols. *Communications Surveys Tutorials, IEEE*, 13 :688–707, 2010.

[155] T. Winter and P. Thubert. RPL : IPv6 routing protocol for low power and lossy networks. RFC6550 (Proposed Standard), Mars 2012.

[156] J. Wu and F. Dai. Distributed dominant pruning in ad hoc networks. In *Proceedings of the IEEE International Conference on Communications, ICC03*, pages 353–357, Alaska, USA, 2003.

[157] F. Xia. Qos challenges and opportunities in wireless sensor/actuator networks. *Sensors*, 8(2) :1099–1110, 2008.

[158] Y. Xu and W. C. Lee. Psgr : priority-based stateless geo-routing in wireless sensor networks. In *Proceedings of the 2nd IEEE Conference Mobile Ad-hoc and Sensor Systems, MASS05*, pages 7–10, Washington, DC, USA, 2005.

[159] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh. Exploiting heterogeneity in sensor networks. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM05*, pages 878–890, Miami, USA, 2005.

[160] F. Ye, G. Zhong, S. Lu, and L. Zhang. Gradient broadcast : a robust data delivery protocol for large scale sensor networks. *Wireless Network*, 11 :285–298, 2005.

[161] M. Younis, M. Youssef, and K. Arisha. Energy-aware management for cluster-based sensor networks. *Computer Network*, 43 :649–668, 2003.

[162] O. Younis and S. Fahmy. Heed : A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Transactions on Mobile Computing*, 3 :366–379, 2004.

[163] O. Younis, M. Krunz, and S. Ramasubramanian. Node clustering in wireless sensor networks : recent developments and deployment challenges. *IEEE Network*, 20(3) :20–25, 2006.

[164] Y. Zhang and L. Cheng. Flossiping : a new routing protocol for wireless sensor networks. In *Proceedings of the IEEE International Conference on Networking, Sensing and Control, ICNSC04*, pages 1218–1223, Taipei, Taiwan, 2004.

[165] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems, SenSys03*, pages 1–13, Los Angeles, California, USA, 2003.

[166] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic. Impact of radio irregularity on wireless sensor networks. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services, MobiSys04*, pages 125–138, Boston, MA, USA, 2004.

[167] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic. Models and solutions for radio irregularity in wireless sensor networks. *ACM Transaction Sensor Network*, 2 :221–262, 2006.

[168] Z. Zhou, S. Zhou, S. Cui, and J. H. Cui. Energy-efficient cooperative communication in a clustered wireless sensor network. *IEEE Transactions on Vehicular Technology*, 57(6) :3618–3628, 2008.