République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed El Bachir El Ibrahimi, Bordj bou Arréridj

Faculté des Mathématiques et d'Informatique

Mémoire

en vue de l'obtention du diplôme de

Master en Informatique

Option : Réseaux et Multimédia

Titre

Routage à délai borné et à basse consommation énergétique pour les réseaux de capteurs et d'actionneurs sans fil

Présenté Par

Med Amin TRIRAT & Hakim TOUBACHE

Soutenu le 26 Septembre 2021

Devant le jury composé de :

Président	Makhlouf NAILI	MCB, Université de BBA
Rapporteur	Nadjib BENAOUDA	MCB, Université de BBA
Examinateur	Djaafar ZOUACHE	MCA, Université de BBA

Promotion 2020-2021

Remerciements

Nous tenons à remercier, tout d'abord, « Allah », le tout puissant, de nous avoir donné le courage et la patience afin d'accomplir, au mieux, ce présent travail.

Nous tenons à exprimer tous nous sincères remerciements à notre encadrant, Mr. Nadjib BENAOUDA, pour nous avoir suivi au cours de la réalisation de ce mémoire. Ses encouragements dans nos moments de doute et la confiance qu'il nous a accordé nous ont permis de mener ce projet à terme. Nous tenons encore à lui exprimer notre profonde gratitude pour son aide et tous ses précieux conseils.

Nous tenons à exprimer, également, notre gratitude aux membres du jury pour avoir accepté de lire notre mémoire et évaluer notre travail.

Un grand merci à tous nos collègues en Master 2 R&M et à tous les enseignants du département Informatique, qui nous ont offert un environnement estudiantin extrêmement agréable.

Un merci pudique : à nos familles, pour leur soutien qui nous a poussé à chercher au fond de nous la volonté de faire toujours beaucoup plus, à nos amis et tous ceux qui ont contribué de près ou de loin à l'aboutissement de ce travail.

Dédicaces

Je dédie ce modeste travail à l'âme de mon père.

Hakim

Merci à Dieu.

Je dédie ce modeste travail à tous ceux qui me sont chers à mon cœur :

À mes chers parents

À mes chères sœurs

À mes respectables professeurs

À tous ceux qui m'aiment et ceux que j'aime.

Enfin je tente à exprimer mon gratitude à toute personne ayant contribué de près ou de loin à l'élaboration de ce mémoire.

Que la paix d'ALLAH soit avec tous...Que nous réunisse dans son vaste paradis INSHALLAH.

Mohammed Amin

Résumé

Dans ce travail, nous nous sommes intéressés aux réseaux de capteurs et d'actionneurs sans fil. En particulier, le problème auquel s'adresse notre travail est celui du routage des données captées par les nœuds capteurs vers leurs nœuds actionneurs correspondants. Avec un tel routage, il est nécessaire d'établir des routes économes en énergie, à cause de la capacité énergétique limitée des nœuds capteurs, tout en assurant l'arrivée des données avant l'échéance considérée pour que les nœuds actionneurs puissent réagir rapidement. En fait, comme ces deux objectifs sont contradictoires, il est nécessaire d'assurer une balance, et donc, un compromis entre ces deux besoins de minimisation de la consommation en énergie et de délivrance à temps des données captées. Pour répondre à cette problématique, nous avons proposé dans ce présent travail, un nouveau protocole que nous avons baptisé LDER comme acronyme de *Localized Delay-bounded and Energy-efficient Routing.* LDER a été proposé sur la base d'un autre protocole nommé DEDA que nous avons amélioré en proposant des solutions permettant une conservation plus importante d'énergie. Pour valider LDER, nous nous sommes procédés par simulation à l'aide du simulateur J-sim. Notre protocole avec DEDA ont les deux été intégrés dans le simulateur. Les résultats obtenus sont présentés sous formes de courbes et d'instantanés, et montrent la capacité de LDER à assurer la balance désirée et aussi sa supériorité en comparaison avec DEDA.

Abstract

In this present work, we are interested in wireless sensor and actor networks (WSANs). In particular, the problem addressed is that of routing of the sensed data by sensor nodes to their corresponding actor nodes. With such routing, it is necessary to establish energy-efficient routes, because of the limited energy of sensor nodes, while ensuring the arrival of data before the given deadline so that, the actor nodes can react quickly. As these two objectives are contradictory, it is necessary to ensure a balance, and therefore, a tradeoff between these two needs of energy-consumption minimization and the quick delivering of the sensed data. To answer this question, we have proposed in this work, a new protocol that we have called LDER as an acronym for *Localized Delay-bounded and Energy-efficient Routing*. LDER was proposed on the basis of another protocol named DEDA. This last was improved by proposing new solutions allowing a greater conservation of energy. To validate LDER, we proceeded by simulation using J-sim simulator. Our protocol with DEDA have both been integrated into this simulator. The results obtained after the various experiments carried out, are presented in the form of curves and snapshots, and show the capacity of LDER to ensure the desired balance and also its superiority in comparison to DEDA.

ملخص

لقد تمحور هذا البحث حول شبكات المجسات و المحركات اللاسلكية. على وجه الخصوص، المشكلة التي حاولنا معالجتها تتمثل في كيفية نقل البيانات التي يتم استشعارها من طرف المجسات إلى المحركات المناسبة. خلال هذا النقل، من الضروري الحرص على توجيه البيانات من خلال مسارات موفرة للطاقة نظرا لمصدر الطاقة المحدود الذي يميز المجسات، و أيضا الحرص على وصول هته البيانات في ظرف زمني محدد لكي تستطيع المحركات الاستجابة بسرعة. في الواقع ، نظراً لأن هذين الهدفين متناقضان ، فمن الضروري ضمان شكل من التوازن ، وبالتالي ، إيجاد حل وسط بين الحاجة إلى تقليل استهلاك الطاقة من جهة و الحرص على وصول البيانات الملتقطة في الوقت المناسب من جهة أخري. كحل لهته المشكلة، اقترحنا في هذا العمل بروتوكولا جديداً أطلقنا عليه اسم LDER كاختصار لجملة Localized Delay-Bounded and Energy-Efficient Routing. لقد تم اقتراح هذا البروتوكول على أساس بروتوكول آخر يسمى DEDA حيث قمنا بتحسين هذا الأخير من خلال اقتراح جملة من الحلول تسمح باقتصاد قدر أكبر من الطاقة. لإثبات فعالية البروتوكول المقترح قمنا بمحاكات هذا الأخير و أيضا البروتوكول الذي اعتمد عليه عملنا بهدف المقارنة بينهما. النتائج التي تم الحصول عليها أثبتت فعالية الاقتراح الخاص بنا من اجل تحقيق التوازن السالف الذكر. هته النتائج أثبتت أيضا تفوق هذا الاقتراح على DEDA.

Table des matières

Table des figures

Liste des tableaux

Liste des Algorithmes

Introduction générale

Ces dernières années, nous avons assisté à une émergence accrue des réseaux sans fil avec des capacités distinctes, des caractéristiques différentes et des applications cibles variées. Parmi ces réseaux, nous pouvons identifier les réseaux de capteurs sans fil (WSNs [1]) et plus récemment, les réseaux de capteurs et actionneurs sans fil (WSANs [2]) auxquels nous nous intéressons dans ce présent travail. Les WSANs sont des réseaux auto-organisés qui sont constitués d'un grand nombre de nœuds capteurs autonomes à faible ressources (i.e. capacité de calcul, de stockage et de communication limitées) 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 nœuds 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 les WSANs, les nœuds capteurs et à la détection d'évènements, doivent délivrer rapidement leurs données aux nœuds actionneurs appropriés car, ces derniers doivent décider des actions à entreprendre et de réagir, par la suite, rapidement afin d'éviter toute catastrophe pouvant être causée par ces évènements. En fait, cela montre le besoin à des protocoles de communication devant assurés la délivrance en temps opportun des données prélevées. En outre, il est nécessaire d'avoir également des protocoles qui conservent l'énergie des nœuds capteurs à cause de la ressource énergétique limitée de ces derniers nœuds.

La conception et le développement de tels protocoles qui répondent conjointement à ces deux besoins n'est pas facile. En fait, la minimisation de la consommation de l'énergie des nœuds capteurs contredit le besoin d'assurer une arrivée à temps des données prélevées. Si on veut minimiser cette consommation, alors il faut attendre à une latence plus importante car, les routes qui seront construites vont avoir beaucoup de nœuds étant donné que, les nœuds vont émettre sur de courtes distances afin de minimiser cette consommation en énergie. Contrairement, si on veut minimiser cette latence, il faut attendre à une consommation plus importante. À partir de cela, il est nécessaire d'assurer une certaine balance et donc, un compromis entre ces deux besoins de minimisation de la consommation en énergie et de délivrance avant échéance des données captées.

Dans ce présent travail, nous nous intéressons à cette problématique décrite précédemment en proposant comme solution, notre propre protocole que nous baptisons LDER comme acronyme de *« Localized Delay-bounded and Energy-efficient Routing »*. LDER a été proposé sur la base d'un autre protocole nommé DEDA [1] que nous avons amélioré en proposant des solutions permettant une conservation plus importante d'énergie.

Pour valider notre proposition, nous avons procédé par simulation à l'aide du simula-

1. Wireless Sensor Neworks.
2. Wireless Sensor and Actor Networks.

teur réseau J-sim en intégrant LDER et aussi le protcole DEDA pour des fins de comparaison. Différentes expérimentations ont été menées qui ont donné des résultats positifs montrant la capacité de LDER à assurer la balance visée et aussi la supériorité de LDER par rapport à DEDA. Les résultats obtenus sont présentés sous formes d'instantanés et de courbes.

Pour bien décrire le travail présenté dans ce projet de fin d'étude, nous présentons dans ce mémoire toutes les notions nécessaires à la bonne compréhension de notre travail ainsi qu'une présentation détaillée de notre contribution, du processus de sa validation ainsi que les résultats obtenus. Sur la base de cela, notre mémoire est organisé somme suit :

- Le premier chapitre a pour but, de donner une description générale sur les WSANs avec toutes les notions y sont liées,
- Le deuxième chapitre s'intéresse au routage dans les WSNs et les WSANs. Il présente, principalement, les particularités et les défis d'un tel routage avec une classification des protocoles de routage existants. Il présente également quelques solutions connexes à notre travail.
- Le troisième chapitre présente les détails de notre protocole proposé LDER. Il présente, particulièrement, nos motivations derrière cette proposition et une description détaillée de LDER incluant une description des structures de données utilisées, des messages échangés et aussi des étapes de son fonctionnement.
- Le quatrième chapitre décrit le processus de validation suivi. Particulièrement, il présente la méthodologie et les paramètres de simulation utilisés, les résultats de simulation obtenus et une discussion sur ces résultats.
- Ce mémoire se termine par une conclusion générale récapitulant notre travail et nos résultats.

Chapitre 1

Généralités sur les réseaux de capteurs et d'actionneurs sans fil

1.1 Introduction

Dans ce chapitre, nous allons présenter les bases nécessaires à la compréhension des spécificités des réseaux de capteurs et d'actionneurs sans fil (abrégés en WSANs), que ce soit du point de vue architectural ou applicatif. Particulièrement, un rappelle sur les architectures et les modes de communication, les différentes formes de dissipation d'énergie, les principales techniques de conservation de cette énergie, la communication temps-réel et quelques scénarios d'application sera présenté.

1.2 Réseaux de capteurs sans fil

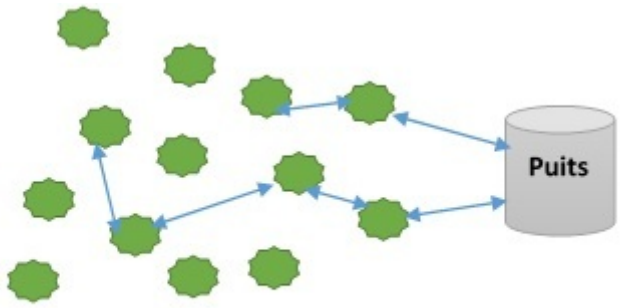


FIGURE 1.1: Réseau de capteurs sans fil WSN.

Les réseaux de capteurs sans fil, abrégés en WSNs, est un ensemble de nœuds capteurs autonomes distribués qui s'auto-organisent en un réseau sans fil multi-sauts (voir la figure [1.2]), leur objectif est de mesurer des paramètres physiques de l'environnement (température, pression, humidité ... etc.), de les traiter et enfin de les envoyer vers un ou plusieurs nœuds spécifiques appelés puits. Un nœud capteur est composé, comme l'illustre la figure (1.2), d'une :

- **Unité de captage :** son rôle est de transformer une grandeur physique récoltée dans l'environnement où il est déployé en une information numérique.
- **Unité de traitement :** un micro-processeur avec une faible puissance de calcul et un faible espace de stockage. Elle est composée de deux interfaces, une interface avec l'unité d'acquisition et une autre avec le module de transmission. Elle contrôle les procédures permettant au nœud de collaborer avec les autres nœuds pour réaliser les tâches d'acquisition, et stocker les données collectées.
- **Unité de transmission :** 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.
- **une source d'énergie :** une batterie limitée partagée par toutes les unités décrites ci-dessus.

Il existe des capteurs qui sont dotés d'autres composants additionnels : les systèmes de localisation comme les GPS (Global Positioning Systems) ou un mobilisateur qui leur permet de se déplacer [2, 3].

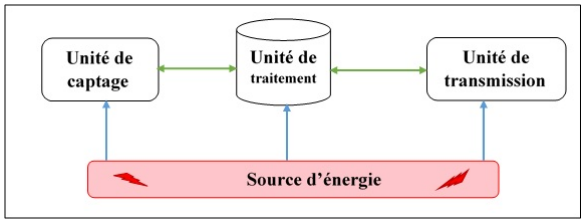


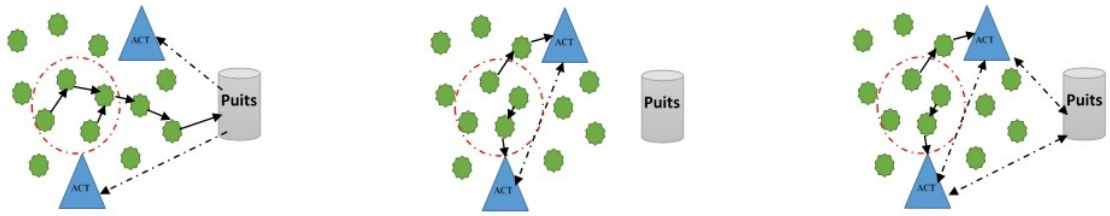
FIGURE 1.2: Architecture d'un noeud capteur sans fil.

1.3 Réseaux de capteurs et d'actionneurs sans fil

Un WSAN est composé généralement d'un nombre important de nœuds capteurs qui sont utilisés pour recueillir les informations de l'environnement, et un nombre moins important de nœuds actionneurs qui sont utilisés pour modifier le comportement de l'environnement. Il y a des liaisons sans fil entre les nœuds capteurs et les actionneurs. Les nœuds capteurs détectent et signalent l'état de l'environnement tandis que les nœuds actionneurs collectent les données provenant des capteurs et sont capables d'agir sur l'environnement. 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. Les nœuds actionneurs possèdent des capacités plus importantes et une source d'énergie abondante comme l'action sur le monde physique exige une capacité très élevée en énergie [3, 4].

1.3.1 Architectures de communication dans les WSANs

L'architecture des WSANs est schématisée en trois types d'architecture de communication comme l'illustre la figure (1.3), deux architecture de base [4] (semi-automatique et automatique) et plus récemment, une troisième appelée architecture coopérative [3].



(a) Architecture semi-automatique.

(b) Architecture automatique

(c) Architecture coopérative

FIGURE 1.3: Architectures de communication dans les WSANs.

- **Architectures semi-automatique :** dans cette architecture les capteurs envoient leurs données vers le nœud puits, ce dernier choisit lui-même l'actionneur le plus approprié pour la réalisation de l'action (figure (1.3a)).
- **Architectures automatique :** Dans cette seconde architecture, les nœuds capteurs envoient leurs données directement vers les actionneurs. Ces derniers, traitent toutes les données réceptionnées pour entamer les actions appropriées et informer ou faire suivre les données vers le nœud puits (figure(1.3b)).
- **Architectures coopérative :** Avec cette architecture, les capteurs transmettent les données vers les nœuds actionneurs on multi-sauts. Les actionneurs analysent les données et peuvent consulter le nœud 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 (1.3c)).

1.3.2 Coordination dans les WSANs

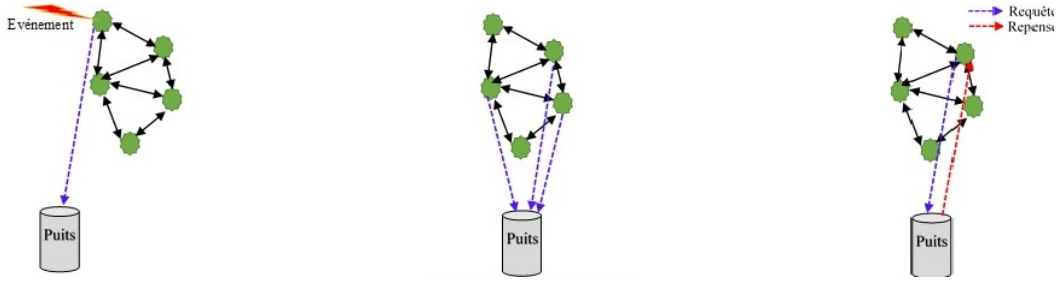
La coordination dans les WSAN est plus compliquée que celle nécessaire dans les WSN. Les deux types de coordinations suivants peuvent être distingués [5] :

- **coordination capteur-actionneur :** après la détection de l'évènement, le nœud capteur traite et transmet les données prélevées vers les nœuds actionneurs appropriés. Le processus de détermination de ces nœuds actionneurs et aussi celui de construction de chemins de données entre le noeud capteur et l'actionneur est appelé coordination capteur-actionneur.
- **coordination actionneur-actionneur :** à la réception des informations sur l'évènement par les nœuds actionneurs, ces nœuds traitent ces informations pour prendre des décisions d'une façon collaborative pour réaliser la tache appropriée (cette tâche peut exiger un seul actionneur ou plusieurs actionneurs), ce processus est appelé coordination Actionneur-Actionneur.

1.4 Modes de communication

La communication dans les WSNs ou les WSANs peut se faire selon différents modes. Particulièrement, les trois modes suivants peuvent être distingués [3] :

- **Communication événementielle :** Ici, un noeud source ne transmet des données que lorsqu'un évènement est détecté dans ses environs. Un évènement est décrit



(a) : Communication événementielle

(b) Communication périodique

(c) Communication à la demande

FIGURE 1.4: Modes de communication.

comme un changement significatif de la valeur mesurée par le capteur dans le temps. Cette technique est utilisée dans des applications de surveillance (e.g. détection d'incendie et d'intrusion). Ce modèle diffère du modèle synchrone parce que le trafic n'est ni constant, ni prévisible, mais défini par l'apparition aléatoire des évènements et leur distribution géographique dans le réseau (voire la figue (1.4a)).

- **Communication périodique :** Dans ce mode, les nœuds qui se chargent de la collection des données transmettent des messages, à intervalles réguliers, vers leurs destinataires. Chaque nœud génère donc des flux intermittents de messages qui lui permettent de passer périodiquement en mode sommeil afin de sauvegarder son énergie (voire la figue (1.4b)). Ce mode est principalement utilisé dans des applications de supervision. Les réseaux périodiques permettent, sous réserve d'un échantillonnage adapté, de rassembler suffisamment d'informations sur une zone donnée pour en déduire les tendances et les comportements. Toutefois, la synchronisation des nœuds nécessaire à ces trafics implique l'échange de paquets de contrôle qui peut engendrer une surcharge réseau non-négligeable.
- **Communication à la demande :** Dans ce cas, les nœuds capteurs ne transmettent de l'information que lorsqu'un autre nœud du réseau (potentiellement la passerelle) en fait explicitement la requête via un message d'intérêt. Ce modèle en requête/réponse est pratique pour un trafic sporadique, toutefois, quand le nombre de requêtes augmente la surcharge induite par celles-ci peut engendrer une dégradation notable des performances du réseau (voire la figue (1.4c)).

1.5 Objectifs et contraintes

Les WSANs présentent un intérêt considérable et une nouvelle étape dans l'évolution des technologies de l'information et de la communication. Cette nouvelle technologie suscite un intérêt croissant vu la diversité de ces applications : santé, environnement industrie et même dans le domaine sportif.

Lors de la conception et du développement de toute solution qui est dédiée à ces réseaux, certains objectifs de conception doivent être considérés. Particulièrement, et en ce qui concerne la communication capteur-actionneur, communication auquel nous nous intéressons dans notre travail, deux objectifs sont à considérés :

- La minimisation de la consommation énergétique des nœuds de capteurs et la prolongation de la durée de vie du réseau.

- La délivrance fiable et en temps opportun des données prélevées aux noeud actionneurs appropriés.

Cette conception doit se faire en tenant en compte de certaines contraintes, particulièrement, les capacités de calcul, de stockage et de communication limitées des nœuds capteurs et aussi leur source énergétique limitée.

1.6 formes de dissipation d'énergie d'un nœud capteur

Dans les WSANs, on distingue entre deux type d'alimentation. Celle des nœuds actionneurs caractérisés par des sources d'énergie importantes et celle des nœuds capteurs qui sont généralement alimentés par des batteries avec une durée de vie limité. Étant donné que o une influence sur la durée de vie du réseau, ils est nécessaire de gérer judicieusement la consommation d'énergie afin de prolonger cette durée de vie du réseau car, le remplacement des batteries n'est pas une option envisageable pour des réseaux avec des milliers de nœuds. Afin de concevoir des solutions efficaces en énergie, il est extrêmement important de faire d'abord une analyse des différents facteurs provoquant la dissipation de l'énergie d'un nœud-capteur. Cette dissipation d'énergie se fait dans l'une des trois opérations suivante : détection, traitement et communication.

1.6.1 Opération de captage (détection)

Il y a plusieurs sources de consommation d'énergie par le module de détection, notamment l'échantillonnage et la conversion des signaux physiques en signaux électriques, ainsi que le conditionnement des signaux et la conversion analogique-numérique. Étant donné la diversité des capteurs, il n'y a pas de valeurs typiques de l'énergie consommée. En revanche, les capteurs passifs (température, sismiques, ...) consomment le plus souvent peu d'énergie par rapport aux autres composants du nœud capteur. Notons, les capteurs actifs tels que les sonars, les capteurs d'images, etc. peuvent consommer beaucoup d'énergie.

1.6.2 Opération traitement

Généralement les unités de traitement possèdent divers modes de fonctionnement : *actif*, *veille* et *Sommeil*, à des fins de gestion d'énergie. Chaque mode est caractérisé par une quantité différente de consommation d'énergie. Toutefois, la transition entre les modes de fonctionnement implique un surplus d'énergie et de latence. Ainsi, les niveaux de consommation d'énergie des différents modes, les coûts de transition entre les modes mais encore le temps passé par l'unité de traitement dans chaque mode ont une incidence importante sur la consommation totale d'énergie d'un nœud-capteur.

1.6.3 Opération de communication

D'après les expériences menées dans [6], la transmission de données est la phase la plus consommatrice en énergie dans le fonctionnement d'un nœud capteur. Le coût d'une transmission d'un bit d'information est approximativement le même que le coût nécessaire au calcul d'un millier d'opérations. La radio opère dans quatre modes de fonctionnement : émission, réception, veille, et sommeil. Une observation importante dans le cas de la plupart des radios est que le mode veille induit une consommation d'énergie significative, presque égale à la consommation en mode réception. Ainsi, il est plus judicieux d'éteindre complètement la radio plutôt que de passer en mode veille quand l'on a ni à émettre

ni à recevoir de données. Un autre facteur déterminant est que, le passage de la radio d'un mode à un autre engendre une dissipation importante due à l'activité des circuits électroniques. Par exemple, quand la radio passe du mode sommeil au mode émission pour envoyer un paquet, une importante quantité d'énergie est consommée pour le démarrage de l'émetteur lui-même.

1.7 Techniques de conservation d'énergie

La durée de vie est sans doute l'indicateur le plus important pour évaluer la performance d'un réseau de capteurs sans fil. Dans le contexte de la disponibilité et de la sécurité des réseaux de capteurs, la durée de vie est également considérée comme un paramètre de base. Maximiser la durée de vie du réseau revient à réduire la consommation énergétique des nœuds. Bien que des progrès aient été réalisés, la longévité de ces appareils alimentés par batterie reste un défi majeur et un facteur clé, et des recherches supplémentaires sur l'efficacité énergétique de la plate-forme et protocole de communication sont nécessaires. Les mesures expérimentales montrent qu'en général, la transmission de données est la plus consommatrice d'énergie et que le calcul lui-même consomme peu d'énergie. La consommation énergétique du module de détection dépend de la spécificité du capteur [7].

Dans de nombreux cas, la consommation de l'unité de détection est négligeable par rapport à celle du module de traitement. Cependant, dans d'autres cas, l'énergie utilisée lors de cette détection peut être égale ou supérieure à l'énergie nécessaire à la transmission des données. D'une manière générale, la technologie d'économie d'énergie se concentre sur deux parties : la partie réseau (c'est-à-dire considérant la gestion de l'énergie en fonctionnement de chaque nœud, et dans la conception du protocole réseau), la partie détection de la technologie est utilisée pour réduire le nombre ou la fréquence d'échantillonnage énergivore). La durée de vie des réseaux de capteurs peut être prolongée grâce à l'application conjointe de différentes technologies. Par exemple, les protocoles d'économie d'énergie sont conçus pour minimiser la consommation d'énergie pendant les activités du réseau. Cependant, les composants du nœud (CPU, radio, etc.) consomment beaucoup d'énergie même s'ils sont inactifs [8].

L'énergie consommée par un capteur est principalement due aux opérations suivantes : la détection, le traitement et la communication. Plusieurs approches sont proposées pour réduire la consommation énergétique dans ces réseaux en économisant la consommation énergétique soit au niveau de captage , au niveau de traitement ou encore au niveau de la communication. D'une manière générale, nous distinguons trois classes de techniques de conservation d'énergie [8] : Duty-cycling, approches orientées données, et mobilité. Nous détaillerons dans ce qui suit chacune de ces classes.

1.7.1 Techniques du Duty-cycling

Cette technique est principalement utilisée dans l'activité réseau. Le moyen le plus efficace pour conserver l'énergie est de mettre la radio de l'émetteur en mode veille (low-power) à chaque fois que la communication n'est pas nécessaire. Idéalement, la radio doit être éteinte dès qu'il n'y a plus de données à envoyer et ou à recevoir, et devrait être prête dès qu'un nouveau paquet de données doit être envoyé ou reçu. Ainsi, les nœuds alternent entre périodes actives et sommeil en fonction de l'activité du réseau. Ce comportement est généralement dénommé Duty-cycling. Un Duty-cycle est défini comme

étant la fraction de temps où les nœuds sont actifs Comme les nœuds-capteurs effectuent des tâches en coopération, ils doivent coordonner leurs dates de sommeil et de réveil. Un algorithme d'ordonnancement Sommeil/Réveil accompagne donc tout plan de Duty-cycling. Il s'agit généralement d'un algorithme distribué reposant sur les dates auxquelles des nœuds décident de passer entre l'état actif et l'état sommeil. Il permet aux nœuds voisins d'être actifs en même temps, ce qui rend possible l'échange de paquets, même si les nœuds ont un faible duty-cycle (i.e., ils dorment la plupart du temps).

Protocoles Sleep/Wakeup

comme mentionné précédemment, un régime sleep/wakeup peut être défini pour un composant donné (i.e. le module Radio) du nœud-capteur. On peut relever les principaux plans sleep/wakeup implantés sous forme de protocoles indépendants au-dessus du protocole MAC (i.e. au niveau de la couche réseau ou de la couche application), les protocoles sleep/wakeup sont divisés en trois grandes catégories :à la demande, rendez-vous programmés, régimes asynchrones. Les protocoles à la demande utilisent l'approche la plus intuitive pour la gestion d'énergie. L'idée de base est qu'un nœud devrait se réveiller seulement quand un autre nœud veut communiquer avec lui. Le problème principal associé aux régimes à la demande est de savoir comment informer un nœud en sommeil qu'un autre nœud est disposé à communiquer avec lui. À cet effet, ces systèmes utilisent généralement plusieurs radios avec différents compromis entre énergie et performances (i.e. une radio à faible débit et à faible consommation pour la signalisation, et une radio à haut débit mais à plus forte consommation pour la communication de données).

Les protocoles du niveau MAC :

plusieurs protocoles MAC pour les réseaux de capteurs sans fil ont été proposés, et de nombreux états de l'art et introductions aux protocoles MAC sont disponibles dans la littérature. Dans ce qui suit, nous nous intéressons aux protocoles MAC qui implémentent des méthodes de duty cyling pour la gestion de l'énergie. Ces protocoles peuvent être classés en trois catégories : protocoles basés sur TDMA, protocoles avec contention et protocoles hybrides.

- **Protocoles MAC basés sur TDMA :** Dans les protocoles MAC basés sur TDMA, le temps est divisé en trames et chaque trame est constituée d'un nombre de slots. A chaque nœud est attribué un ou plusieurs slots par trame, selon un algorithme d'ordonnancement. Pendant ces slots, le nœud capteur peut transmettre/recevoir des paquets à partir d'autres nœuds. Dans de nombreux cas, les nœuds sont regroupés en clusters avec un Cluster-Head chargé d'attribuer les slots de temps pour les nœuds du cluster. Les protocoles TDMA sont par nature économes en énergie, puisque les nœuds n'activent leur radio que pour certains slots de temps et ils s'endorment le reste du temps. Néanmoins, ces protocoles ne sont pas fréquemment utilisés dans les réseaux de capteurs à cause de leurs inconvénients dans la pratique à savoir leurs problèmes de passage à l'échelle, leur besoin de synchronisation et leur sensibilité aux interférences.
- **Protocoles avec contention :** Ces protocoles sont les plus utilisés dans les réseaux de capteurs sans fil grâce à leur robustesse et leur évolutivité. Ils assurent le duty cycling par une intégration étroite des fonctionnalités d'accès au canal avec une approche sleep/wakeup. La seule différence est que, dans ce cas, l'algorithme sleep/wakeup est dépendant du protocole MAC utilisé.

- **Protocoles MAC hybrides :** L'idée de base des protocoles MAC hybrides est l'alternance de fonctionnement entre les protocoles TDMA et CSMA, selon le niveau de contention. Ils combinent les points forts des protocoles MAC basés sur TDMA et ceux avec contention tout en compensant leurs faiblesses [8].

1.7.2 Techniques orientées données

Généralement, les plans Duty-cycling ne tiennent pas compte des données prélevées par les nœuds. Par conséquent, des approches orientées données peuvent être utiles pour améliorer l'efficacité en énergie. En fait, la détection (ou prélèvement de données) affecte la consommation d'énergie de deux manières :

- **Des échantillons inutiles :** les données échantillonnées ont souvent de fortes corrélations spatiales et/ou temporelle il est donc inutile de communiquer les informations redondantes à l'actionneur. Un échantillonnage inutile implique une consommation d'énergie à son tour inutile. En effet, même si le coût de l'échantillonnage est négligeable, cela induit aussi des communications tout le long du chemin qu'emprunte le message.
- **La consommation électrique du module de détection :** réduire la communication ne suffit pas lorsque le capteur est lui-même très consommateur. Des techniques orientées données sont conçues pour réduire la quantité d'échantillonnage de données en garantissant un niveau de précision acceptable dans la détection pour l'application [8].

Réduction des données

Réduire les données en terme de volume ou de nombre de paquets, dans le réseau peut avoir un impact majeur sur la consommation d'énergie due à la communication. Parmi les méthodes de réductions de données, nous trouvons le In-network processing qui consiste à réaliser de l'agrégation de données (par exemple, calculer la moyenne de certaines valeurs) au niveau des nœuds intermédiaires entre la source et l'actionneur. Ainsi, la quantité de données est réduite tout en parcourant le réseau vers l'actionneur. Une agrégation de données appropriée est spécifique à l'application. La compression de données peut être également appliquée pour réduire la quantité d'informations transmises par les nœuds sources [7].

Acquisition de données efficace en énergie

Les techniques d'acquisition de données efficaces en énergie vise à réduire la consommation d'énergie du module de détection. Ceci va à l'encontre de l'hypothèse selon laquelle la détection n'a pas une consommation d'énergie significative. En effet, ce module peut, dans certaines applications, consommer plus d'énergie que la radio ou le reste des composants du nœud capteur. Cela est dû principalement à la consommation excessive d'énergie de certains composants du module de détection tels que les transducteurs, les convertisseurs A/N et les dispositifs de capture actifs ou simplement liés à un temps d'acquisition long. Ces techniques ont pour objectif de réduire le nombre d'acquisitions (échantillons de données). Réduire les données prélevées permet de réduire le nombre de communications et donc d'agir sur la consommation du module de détection lui-même et par conséquent la consommation globale du réseau [8].

1.7.3 Mobilité

Dans certains cas où les nœuds sont mobiles, la mobilité peut être utilisée comme outil pour réduire la consommation d'énergie (au-delà du duty-cycling et des techniques orientées données). Dans un réseau de capteurs statiques, les paquets provenant des nœuds suivent des chemins multi-sauts vers l'actionneur. Ainsi, certains chemins peuvent être chargés (sollicités plus que d'autres), et les nœuds proches d'actionneur relayant plus de paquets et sont plus sujets à l'épuisement prématuré de leurs batteries. Si certains nœuds (éventuellement, l'actionneur) sont mobiles, le trafic peut être modifié si les nœuds mobiles sont chargés de collecter des données directement à partir de nœuds statiques.

Les nœuds ordinaires attendent le passage d'un dispositif mobile pour lui envoyer leurs messages de telle sorte que la communication ait lieu à proximité (directement ou au plus avec un nombre limité de sauts). Par conséquent, les nœuds ordinaires peuvent économiser de l'énergie parce que la longueur du chemin, la contention et les overheads de diffusion sont ainsi réduits. En outre, le dispositif mobile peut visiter le réseau afin de répartir uniformément la consommation d'énergie due à la communication. Lorsque le coût de la mobilité des nœuds de capteurs est prohibitif, l'approche classique consiste à attacher un capteur à des entités qui seront en itinérance dans le champ de détection, comme des autobus ou des animaux [7, 9].

1.8 Les communications temps réel dans les WSANs

Un système temps réel est un système dans lequel l'exactitude des applications ne dépend pas seulement de l'exactitude du résultat mais aussi du temps auquel ce résultat est produit. Si les contraintes temporelles de l'application ne sont pas respectées, on parle de défaillance du système. Il est donc essentiel de pouvoir garantir le respect des contraintes temporelles du système. Ceci nécessite que le système permette un taux d'utilisation élevé, tout en respectant les contraintes temporelles identifiées [6].

Les délais de bout en bout dans les WSANs proviennent essentiellement du temps d'accès au medium et de la longueur des routes. Pour pouvoir minimiser ces délais, il faut :

- Au niveau MAC, borner le temps d'accès au canal qui dépend du protocole utilisé, mais aussi du nombre de nœuds qui ont du trafic à écouler,
- Au niveau du routage, pouvoir contraindre le nombre de sauts nécessaires pour atteindre le nœud actionneur.

On distingue deux types de systèmes temps-réel : les systèmes temps-réel strict ou dur (hard real-time) et les systèmes temps-réel souple ou mou (soft real-time).

1.8.1 Les systèmes temps-réel dur (hard real-time) :

Dans ces systèmes, le non-respect de l'échéance d'un paquet peut avoir des conséquences catastrophiques sur des vies humaines ou sur l'environnement. Par exemple, pour une application de détection de départ de feu de forêt, si l'alarme arrive en retard sur l'échéance, le feu peut alors devenir extrêmement difficile à maîtriser et détruire non-seulement des écosystèmes, mais aussi menacer des constructions et des vies humaines.

1.8.2 Les systèmes temps-réel souple ou mou (soft real-time) :

Dans ces systèmes, une proportion des messages peut rater l'échéance sans conséquences. C'est le cas pour une application de diffusion de vidéo : lors d'une transmission vidéo, des trames contenant des informations sur les images à afficher sont envoyées successivement. Si certaines trames dépassent l'échéance ou sont perdues, la qualité de la vidéo sera dégradée mais l'application pourra continuer à l'afficher. Pour avoir un système-temps réel efficace, on doit minimiser le délai de bout-en-bout.

1.9 Quelques scénarios d'applications

Voici quelques scénarios d'utilisation des WSANs :

- **La télé-relève :** deux types de trafics sont utilisés. Le premier type correspond aux données collectées où les nœuds capteurs envoient périodiquement des mesures vers le puits ou les nœuds actionneurs. Le deuxième type correspond aux messages de contrôle disséminés où un message de contrôle doit être disséminé des actionneurs ou de puits vers les nœuds capteurs.
- **La surveillance :** ce scénario peut servir pour récolter des données périodiques sur l'environnement ou pour détecter des dépassements de seuil d'un paramètre afin de déclencher une alerte. Le trafic dominant consiste en une collecte de données des nœuds capteurs vers les nœuds actionneurs ou le puits
- **La gestion événementielle :** le trafic dominant dans ces types d'applications est la collecte événementielle. Quand les capteurs détectant un évènement, envoient une alerte aux nœuds actionneur ou au puits.

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

1.10 Conclusion

Dans ce chapitre, nous avons procédé à l'étude des réseaux de capteurs et actionneurs sans fil. Nous avons posé les briques de base et présenté quelques notions nécessaires à la compréhension du travail qui sera présenté dans la suite de cette thèse.

Après ce premier chapitre introductif, nous passerons à l'étude du problème du routage dans les WSANs. Cette étude qui fera l'objet du prochain chapitre.

Chapitre 2

Routage dans les réseaux de capteurs et d'actionneurs sans fil

2.1 Introduction

Les protocoles de routage au sein des WSNs et WSAN sont influencés par un facteur déterminant à savoir : la minimisation de la consommation d'énergie sans une perte considérable de l'efficacité. Dans les réseaux de capteurs, chaque nœud joue le rôle de source et de relais. De ce fait, la défaillance énergétique d'un capteur peut changer significativement la topologie du réseau et imposer une réorganisation coûteuse de ce dernier.

Dans ce chapitre, nous allons mettre en évidence les principaux défis de routage avec une présentation d'une certaine classification potentielle des protocoles de routages existants. Quelques travaux qui sont connexes à notre travail sont également présentés.

2.2 Notion de routage

Dans le contexte des WSNs et des WSANs, le routage peut être défini comme un ensemble d'opérations dont l'objectif est de trouver un chemin qui minimise les dépenses énergétiques tout en assurant le transport des données. D'autre part, le routage dans ces réseaux est un routage multi-sauts avec un acheminement des paquets d'une source de données à une destination qui correspond au nœud puits/actionneur. 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.).

Dans ce contexte, une mauvaise politique de routage peut avoir des conséquences graves sur la durée de vie du réseau [10].

2.3 Types de trafics

Nous trouvons différents types de trafics dans les WSNs et les WSANs. En effet et comme l'illustre la figure (2.1), les trois types de trafic suivants peuvent exister [3] :

2.3.1 Trafic Multi-Points à Point ou convergecast (MP2P)

Avec un tel trafic et comme l'illustre la figure (2.1(a)), les données sont récoltées par des nœuds capteurs et sont acheminées vers le(s) nœud(s) puits ou le(s) nœud(s) actionneur(s).

2.3.2 Trafic Point à Multi-Points ou divergecast (P2MP)

Dans ce cas, 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 2.1(b)).

2.3.3 Trafic Point à Point (P2P)

C'est le type de trafic qui correspond à l'échange de données entre deux nœuds du réseau (Figure 2.1(c)). Il se peut qu'un nœud capteur envoie des données vers un actionneur ou vice versa ou bien deux actionneurs qui échangent des données entre eux.

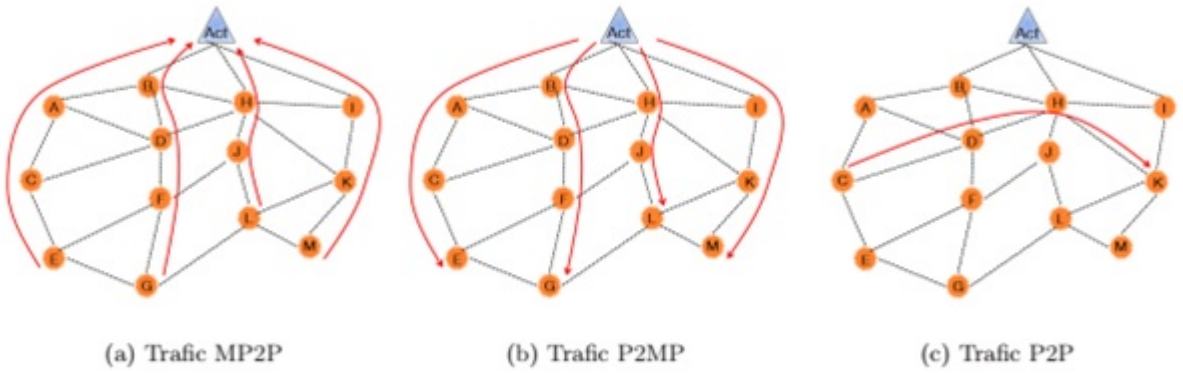


FIGURE 2.1: Différents types de trafic dans les réseaux WSANs

2.4 Défis de routage dans les WSANs

Le routage des données constitue le mécanisme le plus important dans les WSANs. En effet, les capteurs sans fil communiquent par radio à courte portée ce qui rend le routage multi-sauts une nécessité pour l'acheminement des données des nœuds géographiquement éloignés du puits ou du nœud actionneur. Ce mécanisme est responsable de la sélection des chemins pour acheminer les données d'un nœud expéditeur jusqu'à une ou plusieurs puits ou actionneurs. L'objectif principal des protocoles de routage consiste donc à garantir un taux d'acheminement des messages élevé tout en minimisant la consommation d'énergie.

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.

En plus du besoin de la conservation de l'énergie des nœuds, les particularités présentées dans ce qui suit doivent également être considérés.

2.4.1 La dynamicité du réseau

Dans certains cas, les nœuds du réseau sont mobiles rendant la tâche de routage plus complexe. En effet, la mobilité, particulièrement, des nœuds actionneurs entraîne une

modification continue des liens de communication. Les algorithmes de routage doivent par conséquent s'adapter à ce changement de topologie et assurer le routage des paquets.

2.4.2 La qualité de service

Chaque message émis doit arriver à destination dans son intégralité. De plus, dans certaines applications, le délai de transmission est un paramètre de qualité de service important : un message arrivant après un certain délai pourrait être considéré futile.

2.4.3 Le modèle d'émission des données

Il existe plusieurs types de communication entre un noeud capteur et un noeud puits/actionneur. Selon l'application, le modèle d'émission des données peut être périodique, en réponse à des requêtes explicites du noeud puits/actionneur ou orienté événement, c.à.d. les capteurs envoient leurs données seulement si un événement spécifique se produit. Ces modes de communication entre les nœuds et la station de base impliqueront des protocoles de routage totalement distincts.

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 nœuds capteurs recevant ces messages pourront éventuellement les rediffuser ou les ignorer si, par exemple, un actionneur plus proche a été déjà identifié. Ainsi les capteurs transmettent leurs informations vers l'actionneur approprié via des transmissions multi-sauts. Plusieurs scénarios supposent des nœuds actionneurs mobiles, des robots par exemple, qui peuvent se déplacer pour recueillir périodiquement des informations et pour répartir la consommation énergétique des nœuds capteurs [5].

2.5 Routage et types des protocoles

On distingue, principalement, entre le routage des données et le routage des requêtes. Particulièrement :

- **Routage des données :** se fait d'un capteur à un autre capteur, ou à un point de contrôle, ou encore à un utilisateur à travers le réseau.
- **Routage des requêtes :** pouvant provenir d'un capteur, d'un point de contrôle ou d'un utilisateur jusqu'au capteur ayant détecté le stimulus. On parlera aussi d'un routage à priori et d'un routage sur demande. Concernant le routage à priori, il y a un calcul préliminaire des routes dans le réseau, ceci nécessite une maintenance continuelle des routes et par conséquent, l'utilisation de l'énergie du réseau, même si certaines routes ne sont jamais utilisées. Concernant le routage sur demande, la découverte des routes se fait seulement quand il y en a besoin. Les données ne sont pas envoyées tant que la route n'est pas établie. Si les paquets de découverte de route sont perdus, alors le délai augmente.

Un protocole de routage dans un réseau de capteurs doit avoir les propriétés suivantes :

- Il doit pouvoir s'auto-configurer.

- Il doit être robuste dans le cas des pannes de capteurs (de changement de topologie) tout en minimisant la consommation des ressources d'énergie.
- Il devrait prendre en compte la collaboration locale des capteurs pour diminuer la bande passante.
- Il doit supporter le passage à l'échelle (scalability). Les réseaux de capteurs sont destinés à être denses, donc il s'agit d'un problème essentiel.

2.6 Classification des protocoles de routage

Comme le montre la figure 2.2, trois classes de protocoles peuvent être distinguées :

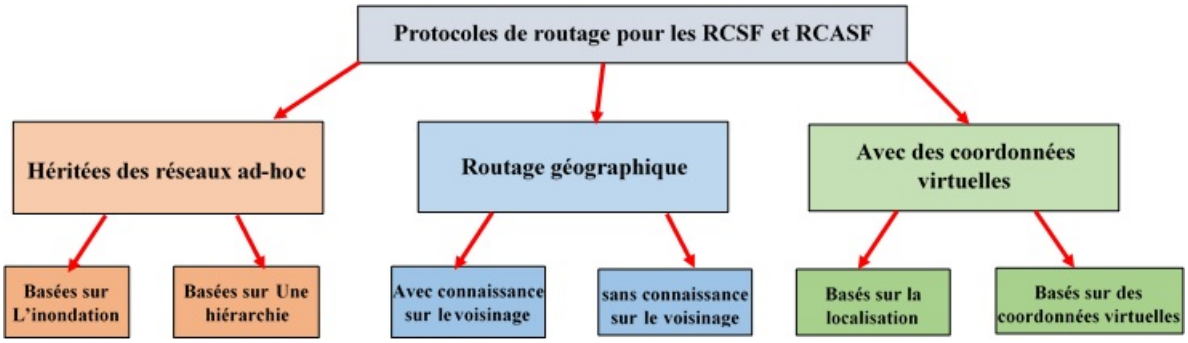


FIGURE 2.2: Classification des protocoles de routage pour les WSNs et les WSANs.

2.6.1 Les Protocoles de routage hérités des réseaux ad-hoc

Les protocoles de routages à plat

Cette technique primordiale permet d'assurer les trois types de trafics MP2P, P2MP et P2P. En effet, les nœuds 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

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 ou 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, sinon un algorithme d'ordonnancement d'activité pour changer les CHs est nécessaire.

2.6.2 Les protocoles de routage géographique

Routage géographique avec connaissance sur le voisinage

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 et /ou actionneurs.

Routage géographique sans connaissance de voisinage

L'envoie des messages vers une destination ce fait autrement, 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 données 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.

2.6.3 Les protocoles de routage avec des coordonnées virtuelles

Protocole basés sur la localisation

Le protocole utilise des techniques pour faire une approximation des coordonnées géographiques réelles.

Protocole basés sur des coordonnées virtuelles

Cette technique 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.

2.7 Travaux connexes

Dans ce qui suit, quelques travaux qui sont connexes à notre travail sont présentés. Particulièrement, quatre travaux de la littérature sont brièvement décrits.

DEPR [11] est l'un des premiers protocoles qui ont été conçus pour les WSANs. C'est un protocole réactif qui permet le partitionnement des nœuds capteurs situant dans la zone d'occurrence de l'évènement en de multiple clusters. Cela est fait en associant chaque noeud capteur détectant l'évènement avec le noeud actionneur le plus proche de lui, ce dernier qui est considéré comme cluster-head. Au sein de chaque cluster, un arbre appelé da-tree (data aggregation tree) est créé dont le noeud racine est le noeud actionneur responsable de ce cluster. Au cours de ce partitionnement, l'établissement de chemins économes en énergie et qui assurent l'arrivée avant échéance des données est visé. Pour assurer cela, les nœuds capteurs changent leurs comportements suivant une certaine réponse qui leur arrive du nœud actionneur. Cette valeur qui reflète la fiabilité des communications en cours est définie comme le ratio entre le nombre de paquets reçus à temps et le nombre total des paquets générés dans un intervalle donné. Suivant cette réponse et lorsque la valeur de fiabilité est au-dessous d'un certain seuil donné, chaque capteur choisit le voisin le plus proche géographiquement de la destination finale. Cela aide à minimiser le nombre de sauts du chemin qui mène vers cette destination, et donc, qui contribue dans la minimisation de la latence des données. Cette minimisation qui se fait au détriment d'une

consommation énergétique plus importante. Dans le cas contraire, lorsque la valeur de fiabilité est au-dessus du seuil prédéterminé, ce qui reflète un excès de fiabilité, chaque nœud capteur achemine les données reçues vers son voisin le plus proche afin d'utiliser une petite puissance de transmission. L'agrégation des données est également utilisée en agrégeant les données prélevées lorsqu'elles se rencontrent au niveau des mêmes nœuds capteurs.

Dans [12, 13], les auteurs proposent une approche qui permet d'associer chaque nœud capteur avec le nœud actionneur le plus proche de lui. Cette association permet la formation préalable de clusters autour des nœuds actionneurs existants et qui sont considérés comme cluster-head. Leur approche procède selon trois phases. La première, d'apprentissage, exécutée au déploiement initial, et assurée par un protocole baptisé ADP (Actuator-discovery Protocol), donne l'opportunité à chaque nœud capteur de procéder à un attachement à un nœud actionneur approprié. La deuxième phase de coordination permet à chaque nœud capteur de router les évènements qui se produisent au nœud actionneur auquel il est attaché, et cela, en utilisant le chemin établi lors de la première phase. Dernièrement, la troisième phase maintient la clusterisation initiale et permet le ré-établissement des chemins qui mènent vers chaque nœud actionneur lorsqu'ils deviennent obsolètes. En ce qui concerne la QoS assurée, l'approche proposée garantit une latence courte en considérant le plus court chemin, en termes du nombre de sauts, qui sépare chaque nœud capteur de son nœud actionneur correspondant. Pour ce qui est de l'efficacité énergétique, l'approche favorise la modification des chemins d'acheminement des données au cours du fonctionnement du réseau sur la base de l'état énergétique des différents nœuds situant sur les différents chemins. En outre, elle permet aux nœuds actionneurs, par une forme d'une supervision centrale, de contrôler les duty cycling de leurs nœuds capteurs correspondants.

CCR [14] est un autre protocole qui est dédié à ces réseaux de capteurs et d'actionneurs sans fil. Avec ce protocole, les nœuds capteurs s'auto-organisent en un ensemble de clusters. Le cluster-head de chacun de ces clusters est ensuite associé avec le nœud actionneur qui peut couvrir la zone géographique de son cluster. Si plusieurs actionneurs couvrent cette zone, le noeud actionneur le plus proche de ce cluster-head est sélectionné. CCR tire profit de l'agrégation de données en autorisant chaque cluster-head à agréger les données de ses membres. Cependant, Cette agrégation doit être menée sans que la délivrance avant la date limite des données soit violée.

DEDA [1] s'intéresse au problème de construction d'un arbre d'agrégation avec des chemins bornés en délais et économes en énergie. DEDA établit, dans un premier temps, et en se basant sur l'algorithme LMST, une nouvelle topologie éparse du graphe original du réseau. Ensuite, un arbre qui connecte chaque nœud actionneur avec les nœuds capteurs les plus proches de lui est construit au-dessus de cette nouvelle topologie. Cet arbre est utilisé, sans modification, si la date limite imposée n'est pas violée. Dans le cas contraire, cet arbre est ajusté en remplaçant certains chemins de la topologie éparse par d'autres liens du graphe original. Cet ajustement se fait sur la base d'une certaine valeur qui est calculée au niveau de chaque nœud et qui reflète l'avancement désiré sur la topologie éparse afin que la date limite soit respectée. Cette valeur est calculée par la formule suivante : $DEP(DEsiredProgress) = \lceil \frac{LD(w)}{DL-MED} \rceil l(w)$. $LD(w)$ représente la distance qui sépare le nœud w (le noeud courant) du nœud actionneur au niveau de la topologie éparse. DL est la date limite qui est spécifiée par l'utilisateur. MED est le plus grand délai déjà expérimenté par tous les rapports reçus à partir des voisins de w. $l(w)$ est défini suivant l'intensité du trafic réseau. Il est égal à 1 si cette intensité n'est pas élevée. Sinon, il correspond au degré du nœud w. Étant donné que la latence des données est affectée par

le nombre de nœuds qui concurrent pour l'accès au canal sans fil, DEDA route les données prélevées, dans le cas de la présence d'un trafic réseau important, le long du chemin dont la somme des degrés de ses nœuds est minimale. Sinon, le chemin qui assure un nombre de sauts minimal est sélectionné.

2.8 Conclusion

Dans ce chapitre, nous avons sommes intéressés au problème de routage dans les WSNs et les WSANs. Nous avons présenté, principalement, les principaux défis du routage avec une classification possible des protocoles de routages existants. Quelques travaux connexes à notre travail ont été également présentés.

Après cette illustration de cette notion de routage au sein des WSNs et WSANs, nous passons à la présentation des détails propres à notre solution de routage proposée, cette présentation qui sera détaillée dans le prochain chapitre.

Chapitre 3

Contribution

3.1 Introduction

Dans ce chapitre, nous présentons tous les détails de notre contribution. En particulier, nous avons proposé un protocole de routage réactif que nous avons baptisé LDER comme acronyme de *Localized Delay-bounded and Energy-efficient Routing.* Ce protocole proposé est dédié au WSANs. Il vise à établir, à l'occurrence d'évènements, des routes économes en énergie qui assurent l'arrivée des données avant une certaine échéance donnée. Ce chapitre présente dans premier temps, le modèle de notre réseau et une brève description de la problématique auquel notre travail essaye de répondre. Quelques remarques qui concernent le protocole DEDA [1] sont ensuite présentées. DEDA est le protocole sur lequel nous nous sommes basés pour proposer notre protocole LDER. Après ces remarques, nous passons à une description brève de LDER et une autre plus détaillée qui présente les structures de données utilisées, les messages échangés et les étapes de fonctionnement de LDER.

3.2 Modèle réseau

Nous assumons notre modèle réseau comme suit :

0. Notre réseau est constitué d'un ensemble de nœuds capteurs statiques et également un seul nœud actionneur qui est aussi statique. Dans notre travail et pour des raisons de simplicité, on se focalise sur le scénario d'un seul nœud actionneur avec les nœuds capteurs qui sont associés avec lui. En fait, avec les WSANs, chaque nœud capteur est généralement associé avec le nœud actionneur le plus proche de lui. De cette façon, le réseau global est divisé en clusters où chaque nœud actionneur avec ses nœuds capteurs membres forment un cluster et chaque nœud capteur émet ses données vers son nœud actionneur. C'est pour cette raison et afin de bien clarifier notre travail, nous nous concentrer uniquement sur ce scénario d'un seul nœud actionneur avec ses nœuds capteurs associés.
0. Chaque noeud i du réseau est conscient de ses propres coordonnées.
0. Les nœuds capteurs peuvent ajuster leurs portées de communication en ajustant leurs niveaux de puissance de transmission sans dépasser une puissance maximale P_{max} qui correspond à une portée maximale R_{max}.
0. Soit $G = \{V, E\}$ le graphe qui modélise la topologie de notre WSN où :
   (0) $V = \{v_1, v_2, ..., v_n\}$ est l'ensemble des sommets représentant les différents nœuds du réseau tel que $|V| = l$ et v_1 est le nœud actionneur.

(0) Et E l'ensemble d'arcs (liens) qui relient ces nœuds et qui est défini comme suit :

$$E = \{(u, v) | u \in V, v \in V, \delta_{u,v} \leq R_{max}\}$$

sachant que :

$$\delta_{u,v} = \sqrt{(x_u - x_v)^2 + (y_x - y_v)^2}$$

représente la distance euclidienne entre les deux nœuds u et v, avec $u = (x_u; y_u)$ et $v = (x_v; y_v)$.

0. Soit $S = \{s_1, s_2, ..., s_m\}$ l'ensemble de nœuds sources qui captent et collectent les données de l'environnement, tel que $|S| = m$ et $S \subseteq (V - \{v1\})$.

0. Similairement à [15], la puissance de transmission nécessaire pour que le nœud i communique avec j est calculée en assumant la formule :

$$power_{i,j} = \delta_{i,j}^{\alpha} \tag{3.1}$$

où α représente l'exposant d'affaiblissement de propagation « path loss ».

3.3 Définition du problème

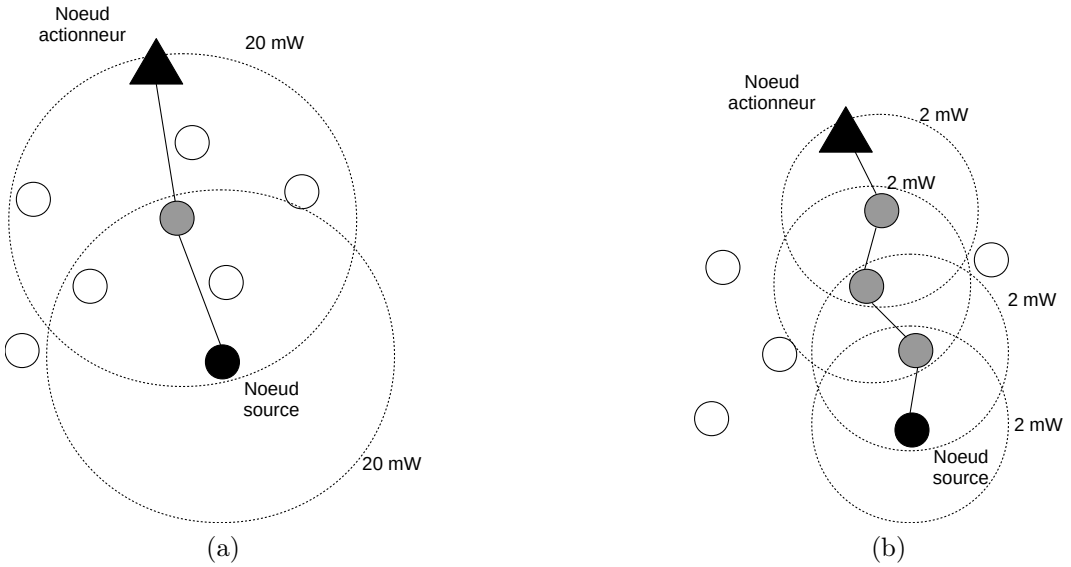
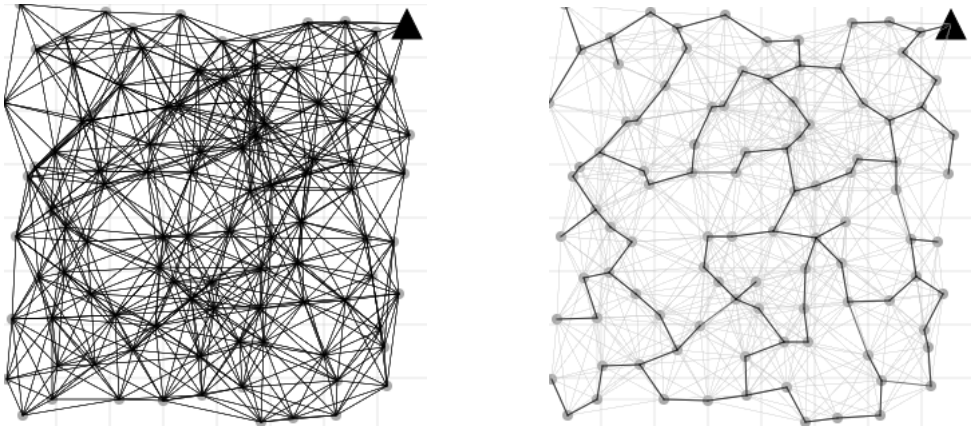


FIGURE 3.1: Deux routes possibles entre le nœud source et le nœud actionneur : (a) une route avec une puissance de transmission totale qui est égale à 40 mW et un nombre de sauts de 2 sauts, (b) une route avec une puissance de 8 mW et un nombre de sauts de 4 sauts.

Dans ce travail, nous nous intéressons aux réseaux de capteurs et d'actionneurs sans fil, en particulier, nous nous intéressons au routage des données prélevées à partir des nœuds capteurs vers les nœuds actionneurs. En fait, cette communication doit se faire, d'un côté, de manière énergétiquement efficace à cause de la source énergétique restreinte des nœuds de capteurs, et d'un autre côté, la communication des données prélevées ne doit pas dépassée un temps de latence donné pour que les nœuds actionneurs puissent entreprendre

les actions appropriées à temps. Pouvoir assurer ce routage n'est pas une chose facile. En fait, ces deux objectifs sont contradictoires. Si nous voulons minimiser au maximum l'énergie consommée lors du routage alors, des chemins avec des liens courts doivent être considérés. Cela va donner à son tour des routes qui ont un grand nombre de sauts, ce qui va augmenter le temps de latence de ces données. Pour illustrer davantage ce problème, prenons la figure (3.1), adaptée de [16], qui illustre deux routes possibles qui connectent les deux nœuds source et actionneur. Avec la figure (3.1a), une route de deux sauts est établie avec une puissance de transmission totale de 40 mW. En contre partie, avec la figure (3.1b), une deuxième route plus énergétiquement efficace avec une puissance de transmission totale de 8 mW est construite mais, avec un nombre de sauts total de quatre. Donc, pouvoir répondre à ce problème de communication capteur-actionneur revient à pouvoir trouver une solution qui assure un compromis entre ces deux objectifs de minimisation de la puissance de transmission totale de la route établie tout en assurant des communications avec des délais qui ne dépassent pas la borne temporelle imposée. En fait, c'est à ce problème que nous essayons de répondre dans ce travail.

3.4 Nos remarques sur le protocole DEDA



(a) (b)

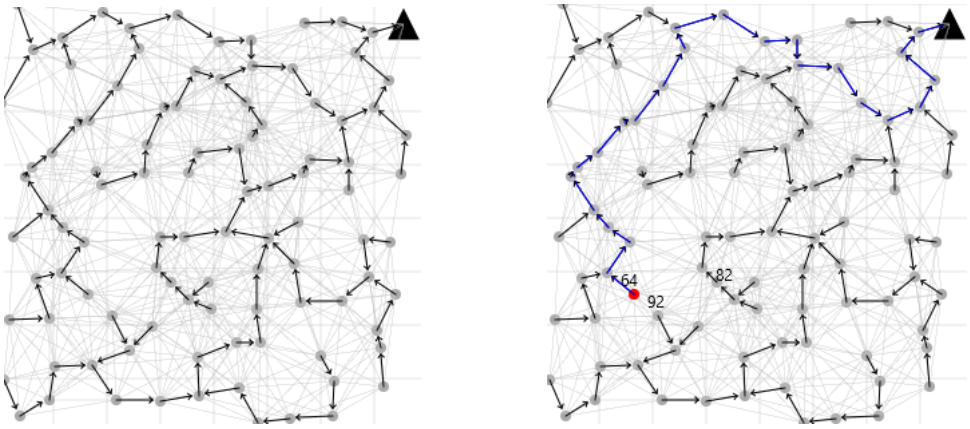
FIGURE 3.2: Exemple de topologies d'un WSAN : (a) topologie initiale générée en utilisant la puissance maximale des nœuds du réseau, (b) et sa topologie éparse générée en appliquant l'algorithme LMST.

Le protocole DEDA fonctionne selon le principe suivant [1] : en appliquant l'algorithme LMST, une nouvelle topologie éparse (figure (3.2b)) est créée à partir de la topologie initiale qui est générée en utilisant les puissances maximales des nœuds du réseau (figure (3.2a)). Sur la base de cette nouvelle topologie, un arbre du plus court chemin qui est enraciné au niveau du nœud actionneur et qui couvre tous les autres nœuds du réseau est construit (figure (3.3a)). Cet nouvel arbre est utilisé sans aucune modification si les routes de cet arbre génèrent des délais qui obéissent à la borne temporelle exigée. Sinon, des raccourcis seront ajoutés en empruntant des liens de la topologie initiale.

En fait, DEDA a été conçu spécialement pour collecter les données de tous les nœuds associés à un nœud actionneur donné suite à la requête de ce dernier. Dans ce cas, tous les nœuds émettent leurs données et les agrègent le long des différents chemins de l'arbre d'agrégation établi préalablement. Ces données sont donc routées selon des chemins qui

sont énergétiquement efficaces car, ce sont des chemins qui ont été construits au-dessus de la topologie LMST.

Cependant, dans le cas des réseaux événementiels avec lesquels, les nœuds capteurs qui détectent des événements sont les seuls qui émettent leurs données aux nœuds actionneurs, DEDA ne procède pas efficacement. Pour illustrer cela, prenons l'exemple de la figure (3.3b). En fait, comme l'illustre cette figure, le nœud source 92 (le nœud en rouge) veut envoyer ses données au nœud actionneur avec lequel il est associé. Si on suppose qu'il n'y a aucune contrainte temporelle, le nœud 92 va utilisé sa route dans l'arbre d'agrégation qui a été construit préalablement (figure (3.3a)) sans aucune modification. Comme nous pouvons le constater à partir de la figure (3.3b), cette route (la route en couleur bleu) est très longue et donc, a un coût énergétique important même si elle est constituée que de liens courts (i.e. des nœuds avec des puissances de transmission minimales). Dans ce cas, il est préférable d'emprunter d'autres chemins qui ont des coûts énergétiques totaux moins importants, par exemple, il est préférable que le nœud 92 passe par son voisin 82, en empruntant le lien 92-82 de la topologie initiale, comme l'illustre la figure (3.4) au lieu d'aller à travers son nœud parent dans l'arbre d'agrégation.



(a) (b)

FIGURE 3.3: Routes de routage avec le protocole DEDA : (a) établissement d'un arbre d'agrégation du plus court chemin au-dessus de la topologie LMST, (b) et la route du nœud source 92 (en bleu) utilisée pour acheminer ses données en supposant une borne temporelle qui est égale à ∞.

La question qui se pose ici *est comment peut-on détecter ce genre de voisins et emprunter des raccourcis dans la topologie initiale pouvant nous donner des chemins avec des coûts énergétiques moins importants*. En fait, c'est dans ce sens qu'aille notre contribution en améliorant DEDA pour le faire adapter aux réseaux de capteurs et d'actionneurs événementiels et chercher des raccourcis en empruntant des liens de la topologie initiale, non pas uniquement pour satisfaire la borne temporelle imposée, mais aussi pour aboutir à des routes qui ont des coûts énergétiques plus bas. Nous baptisons notre protocole *LDER* comme acronyme de *Localized Delay-bounded and Energy-efficient Routing*. Tous ses détails sont décrits dans les sections suivantes.

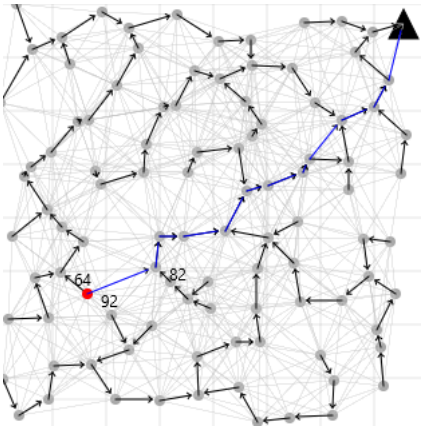


FIGURE 3.4: Solution visée : faire router les données du nœud source 92 via son voisin 82, qui ne fait partie de ses voisins dans la topologie éparse, mais qui assure un chemin vers le nœud actionneur avec un coût énergétique plus basse.

3.5 Vue d'ensemble de notre protocole LDER

LDER est un protocole de routage réactif qui établit toute route nécessaire entre le noeud capteur source et son noeud actionneur correspondant à l'occurrence d'évènements. Comme nous avons déjà mentionné, l'objectif est d'établir des routes économes en énergie qui assurent l'arrivée des données avant leur échéance. Notre protocole LDER opère selon les étapes suivantes :

0. Dans la première étape, chaque noeud du réseau collecte les informations de ses nœuds voisins, en particulier, leurs coordonnées et leurs identifiants. Cette collection se fait à l'aide des messages *HELLO* que chaque noeud diffuse à son voisinage.
0. Sur la base de ces informations collectées, une nouvelle topologie éparse est construite au-dessus de la topologie initiale du réseau en se basant sur l'algorithme LMST. L'utilisation d'une telle topologie éparse présente l'avantage de travailler, dès le départ, avec une topologie économe en énergie car, cette dernière garde uniquement les liens qui consomment la moindre d'énergie.
0. Après l'établissement de cette nouvelle topologie, un arbre du plus court chemin avec noeud actionneur comme racine est construit au-dessus de cette topologie éparse. Le nombre de sauts des routes parcourues est utilisé comme distance pour établir cet arbre. Durant cette étape, chaque noeud détermine son noeud parent initial et collecte les informations propres à tous ses voisins, même à ceux qui ne sont pas dans l'ensemble de ses voisins dans la topologie éparse. En fait, c'est sur la base de ces informations que chaque noeud peut établir des raccourcis dans la topologie initiale et décider de son nouveau parent en cas de dépassement de l'échéance donnée.
0. Une fois cet arbre construit, l'acheminement des données prélevées du noeud source vers le noeud actionneur peut ainsi être effectué à chaque détection d'évènement. Chaque noeud achemine les messages qu'il reçoit en les envoyant vers son nœud parent initial déterminé dans l'étape précédente si :
   (0) Ce parent assure la délivrance à temps de ces messages ;
   (0) Et aucun autre voisin existe, parmi tous ses voisins dans la topologie initiale, qui assure également cette délivrance à temps et qui propose un chemin plus économe en énergie vers le noeud actionneur.

Dans le cas contraire, un raccourci dans la topologie initiale est adoptée en choisissant un nouveau noeud parent parmi tous les voisins du noeud courant dans cette topologie initiale.

Algorithme 1 : Processus de formation de la topologie éparse et de l'établissement de l'arbre du plus court chemin au-dessus de cette topologie.

```
1  pour chaque noeud i faire
2  |  Le noeud i diffuse un message HELLO à son voisinage;
3  |  Le noeud i collecte les messages HELLO de ses voisins dans l'intervalle t_hello;
4  |  Le noeud i calcule son voisinage dans la nouvelle topologie éparse en
   |   appliquant l'algorithme LMST;
   |  // Un message RD (ROUTE_DISCOVERY) sert à établir l'arbre du plus
   |     court chemin au dessus de la topologie éparse. Il est diffusé
   |     initialement par le noeud actionneur
5  |  pour chaque écoute d'un message RD faire
6  |  |  Le noeud i met à jour les informations propores au voisin source du
   |  |   message;
7  |  |  si RD propose un meilleur chemin alors
8  |  |  |  Le noeud i met à jour les structures de données nécessaires, en
   |  |  |   particulier, son parent et sa distance du noeud actionneur;
9  |  |  |  Le noeud i met à jour le message RD et le reddifuse;
10 |  |  fin
11 |  fin
12 fin
```

L'algorithme (1) illustre le principe général de la formation de la topologie éparse et la construction de l'arbre de plus court chemin au-dessus de la nouvelle topologie. Comme l'illustre cet algorithme, chaque noeud i du réseau diffuse initialement un message HELLO contenant son identifiant et ses coordonnées à son voisinage en utilisant sa puissance de transmission maximale. Après l'écoulement de l'intervalle t_{hello} [1], chaque noeud i procède, sur la base des informations de ses voisins reçues préalablement, au calcul de ses nouveaux voisins dans la nouvelle topologie éparse en utilisant l'algorithme LMST. Après cette construction, l'établissement de l'arbre du plus court chemin peut être effectué. Ce processus de construction est initié par le noeud actionneur qui diffuse un message ROUTE_DISCOVERY (RD) à son voisinage. Chaque noeud i qui écoute ce message RD met à jour les informations propres au voisin source du message. Ce même noeud met à jour d'autres structures de données locales, en particulier, l'information de son parent et sa distance du noeud actionneur si ce message propose un meilleur chemin.

À la fin de ce processus, chaque noeud capteur possède les informations nécessaires au déroulement de l'étape 4 cité ci-dessus.

1. t_{hello} est l'intervalle du temps dans lequel chaque noeud reste dans l'attente des messages HELLOs de ses voisins.

3.6 Description détaillée de LDER

Dans cette section, nous présentons en détails notre protocole. Particulièrement, les structures de données utilisées au niveau de chaque noeud du réseau, la liste et format des messages échangés et les étapes de son fonctionnement sont présentées.

3.6.1 Structures de données utilisées

Notre protocole opère sur la base de certaines structures de données qui différent selon le type du noeud, capteur ou actionneur.

Structures de données utilisées au niveau du nœud capteur

Les structures de données suivantes sont utilisées au niveau de chaque noeud capteur i :

- **$\mathbf{Id_i}$:** correspond à l'identifiant du noeud capteur i,
- **$\mathbf{ActorId_i}$:** correspond à l'identifiant du noeud actionneur auquel le noeud i est associé,
- **$\mathbf{Coordinate_i}$:** ce sont les coordonnées du noeud i,
- **$\mathbf{Parent_i}$:** représente l'identifiant du noeud parent du noeud i,
- **$\mathbf{HopToActor_i}$:** c'est le nombre de saut du meilleur chemin qui connecte i avec le noeud actionneur,
- **DL (Delay Limit) :** c'est la borne temporelle qu'il ne faut pas dépassée.
- **$\mathbf{N}_i^{LMST}$:** c'est l'ensemble de nœuds voisins du noeud i dans la topologie éparse (LMST),
- **$\mathbf{N}_i^{MAX}$:** c'est l'ensemble de nœuds voisins du noeud i dans la topologie initiale. Chaque élément de cette ensemble comporte l'identifiant du noeud voisin, la distance (nombre de sauts) du chemin qui connecte ce voisin avec le noeud actionneur dans la topologie éparse et dernièrement le coût énergétique de ce chemin.

Structures de données utilisées au niveau d'un nœud actionneur

Le noeud actionneur participe au processus d'établissement de la topologie éparse et c'est lui qui initie la construction de l'arbre du plus court chemin. Un noeud actionneur i se base sur les structures de données suivantes qui sont similaires à celles du noeud capteur : Id_i, $Coordinate_i$, DL, N_i^{LMST} et N_i^{MAX}.

3.6.2 Liste et formats des messages

Le déroulement de notre protocole se base sur l'échange de certain messages de contrôle qui sont présentés dans cette section.

Message HELLO

Ce message est émis initialement par chaque noeud du réseau. Il permet aux différents nœuds d'être au courant de leurs voisins et de leurs informations. Un message HELLO qui est émis par un noeud i se compose des champs suivants :

- **$\mathbf{Id_i}$:** représente l'identifiant du noeud qui a diffusé le message,

- **Coordinate$_i$:** ce sont les coordonnées de i. Cette information est indispensable à l'application de l'algorithme LMST.

Message ROUTE_DISCOVERY (RD)

Ce deuxième message sert à établir l'arbre du plus court chemin au-dessus de la topologie éparse. Il est initialement diffusé par le noeud actionneur. Un tel message RD qui est émis par un noeud i se compose des champs suivants :

- **Sender :** correspond à Id_i , l'identifiant du noeud qui a diffusé le message,
- **ActorId :** c'est l'identifiant du noeud actionneur qui a initié la diffusion de ce message RD,
- **HopToActor :** représente la distance (nombre de sauts) sur la topologie éparse parcouru par le message RD depuis sa diffusion par le nœud actionneur,
- **DL :** ce champ stocke la borne temporelle qu'il ne faut pas dépassée.
- **Cost :** correspond au coût énergétique accumulé depuis la diffusion initiale du message.
- **N$_{LMST}$:** l'ensemble des nœuds voisins du noeud i dans la topologie éparse. L'utilité de cet ensemble sera présenté par la suite.

Message PARENT_DECISION_MESSAGE (PDM)

Ce message permet de déterminer le chemin approprié qui assure l'arrivée des données prélevées avant leur échéance. Il est émis par un noeud source à chaque détection d'un évènement. Un message PDM qui est émis par un noeud i se compose des champs suivants :

- **Sender :** correspond à Id_i, l'identifiant du noeud qui a diffusé le message,
- **Parent :** représente le prochain saut du message PDM, et donc, l'identifiant du noeud parent du noeud qui a diffusé le message (Parent de i),
- **ERD :** correspond au temps de transmission du message, déjà expérimenté, depuis son initiale émission par le noeud source.

3.6.3 Étapes de fonctionnement

Dans cette section, nous présentons en détail chacune des quatre étapes de fonctionnement de notre protocole LDER présentées dans la section (3.5).

Collection des informations des nœuds voisins

Le fonctionnement de notre protocole débute, et comme l'illustre l'algorithme (3.5), par la collection de chaque noeud du réseau, des informations propres à ses voisins. Pour cela, chaque noeud crée un message HELLO qui contient son identifiant et ses coordonnées et le diffuse en utilisant sa puissance de transmission maximale. Cette collection dure un intervalle de t_{hello} durant lequel chaque noeud émet son message et reste dans l'attente de l'arrivée des autres.

La formation de la topologie éparse via l'application de l'algorithme LMST

Après l'écoulement de l'intervalle t_{hello}, chaque noeud capteur possédera les informations de tous ses voisins dans la topologie initiale. La disponibilité de ces informations permet l'application de l'algorithme LMST. Cela est fait comme suit [17] : chaque nœud et

sur la base des coordonnées de ses voisins, calcule localement son arbre couvrant de poids minimal (MST) en appliquant l'algorithme de Prim par exemple. Les nouveaux voisins de ce noeud dans la nouvelle topologie sont ensuite sélectionnés en considérant uniquement les voisins qui sont à un saut dans ce nouvel arbre créé. La nouvelle topologie générée en considérant les nouveaux voisins de tous les nœuds du réseau est un graphe orienté. Pour la convertir en graphe non orienté, l'une des deux méthodes suivantes peut être appliquée. Avec la première, tout arc entre les deux nœuds i et j est inclut dans la nouvelle topologie uniquement si les deux arcs $e_{i,j}$ et $e_{j,i}$ font respectivement partie des deux MSTs des deux nœuds i et j. La nouvelle topologie dans ce cas est baptisée LMST$^-$. Avec la deuxième méthode, un nouveau arc qui connecte i à j est ajouté seulement si uniquement l'un des deux arcs $e_{i,j}$ ou $e_{j,i}$ est présent. Dans ce deuxième cas, la nouvelle topologie est baptisée LMST$^+$.

Dans ce présent travail, la topologie éparse LMST a été générée à l'aide de l'algorithme de Prim. Les poids des arêtes ont été calculés selon la formule (3.1). Cette dernière permet de calculer la puissance de transmission nécessaire à la communication entre deux nœuds donnés. Comme variation de LMST, nous avons adopté LMST$^-$. Cela a été fait en permettant à chaque noeud de partager ses nouveaux voisins calculés à l'aide des messages RD durant le processus d'établissement de l'arbre du plus court chemin.

Le calcul de l'arbre du plus court chemin au-dessus de la nouvelle topologie

Lorsque la formation de la topologie éparse est terminée, l'établissement de l'arbre du plus court chemin qui connecte les nœuds capteurs avec leur noeud actionneur correspondant peut commencer. Cette construction est débutée comme le montre la ligne 1 de l'algorithme (2), par le noeud actionneur qui diffuse un message *RD* à son voisinage. Les champs *Sender, ActorId, LD, Cost, DL* et N_{LMST} du message *RD* sont, respectivement, initialisés par les valeurs Id_a, Id_a, *0, 0, la borne temporelle donnée et* N_a^{LMST}. Chaque noeud i recevant ce message met à jour l'élément de son ensemble N_i^{MAX} qui correspond à *RD.Sender* par les valeurs *RD.Cost* et *RD.HopToActor*. Il supprime également *RD.Sender* de son ensemble N_i^{LMST} si ce dernier appartient à N_i^{LMST} et Id_i n'appartient pas à l'ensemble $RD.N_{LMST}$. Le noeud i procède de cette façon pour travailler avec la topologie LMST$^-$. Ensuite, le noeud i vérifie si *RD* arrive avec un meilleur chemin. Si cela est le cas alors il met à jour ses informations locales et les champs du message *RD* comme l'illustre, respectivement, les lignes 7-11 et 12-14 de l'algorithme (2). Il rediffuse ensuite le message *RD* en utilisant sa puissance de transmission maximale.

À la fin de cette étape, chaque noeud i connaît la distance et le coût du meilleur chemin qui connecte chacun de ses voisins dans l'ensemble N_i^{MAX} avec le noeud actionneur. Ces informations sont indispensable au bon fonctionnement de la prochaine étape.

Le calcul de la route appropriée lorsqu'un évènement arrive

Après la terminaison de l'établissement de l'arbre du plus court chemin, les nœuds capteurs se mettent dans l'attente de l'occurrence des évènements. Nous rappelons que les données prélevées par chaque noeud capteur doivent arriver avant leur échéance. Pour cette raison et à chaque détection d'un évènement, le noeud source crée un message PARENT_DECISION_MESSAGE (PDM) dont le rôle est de déterminer un chemin qui assure cette délivrance à temps des données et aussi si possible, un chemin qui économise

Algorithme 2 : Processus détaillé de la construction de l'arbre du plus court chemin au-dessus de la topologie éparse.

Le noeud actionneur a diffuse un message RD (ROUTE_DISCOVERY) avec $Sender = Id_a$, $ActorId = Id_a$, $LD = 0$, $Cost = 0$ et $N_{LMST} = N_a^{LMST}$;
// R_{RD} est l'ensemble de noeuds recevant un message RD
pour chaque *noeud* $i \in R_{RD}$ **faire**
    Mettre à jour l'élément qui correspond à RD.Sender dans N_i^{MAX} avec RD.HopToActor et RD.Cost;
    // Test Propre à la variation LMST-
    si $Id_i \notin RD.N_{LMST}$ *ET* $RD.Sender \in N_i^{LMST}$ **alors**
        Supprimer RD.Sender de l'ensemble N_i^{LMST};
    fin
    si $HopToActor_i > RD.HopToActor + 1$ **alors**
        // Mettre à jour les informations locales de i
        $Parent_i \leftarrow RD.Sender$;
        $HopToActor_i \leftarrow RD.HopToActor + 1$;
        $ActorId_i \leftarrow RD.ActorId$;
        $DL \leftarrow RD.DL$;
        // Mettre à jour les champs du message RD
        $RD.Cost \leftarrow RD.Cost + power_{i,RD.Sender}$;
        $RD.Sender \leftarrow Id_i$;
        $RD.HopToActor \leftarrow HopToActor_i$;
        // Le noeud i rediffuse le message RD en utilisant sa puissance de transmission maximale
        Le noeud i rediffuse le message RD;
    fin
fin

plus d'énergie par rapport au premier qui a été établi dans la phase précédente. Donc, chaque noeud qui veut sélectionner le noeud parent du message PDM vérifie si son noeud parent, qui a été défini dans l'étape précédente, est la meilleure solution. C.à.d. ce noeud parent assure l'arrivée des données avant l'échéance imposée et aucun autre noeud voisin de N_i^{MAX} ne présente un chemin plus économe en énergie. Si cela n'est le cas, alors, il faut chercher des raccourcis dans la topologie initiale. Tout ce processus est décrit dans ce qui suit.

Algorithme 3 : Processus d'acheminement d'un message PDM.

```
// S est le noeud source, celui qui détecte l'évènement
// Courant correpond au noeud courant du message PDM
// n correpond au noeud parent sélectionné
courant ← S;
n ← −1;
tant que n ≠ ActorId faire
    Le noeud Courant calcule la valeur DEP selon la formule (3.2);
    pour chaque noeud j ∈ N_i^MAX faire
        Le noeud Courant calcule la valeur Progress_j selon la formule (3.3);
    fin
    Le noeud courant met à jour n selon la formule (3.4);
    Le noeud courant diffuse le message PDM;
fin
```

Chaque noeud i qui veut déterminer son noeud parent après la réception du message PDM, calcule dans un premier temps, la valeur DEP (DEsired Progress). Cette dernière est définie comme étant l'avancement désiré, en nombre de sauts, du noeud i dans la topologie éparse LMST afin de ne pas dépasser la borne temporelle donnée. Cette valeur DEP est calculée comme suit [1] :

$$DEP = \lceil \frac{HopToActor_i}{DL - ERD} \rceil \tag{3.2}$$

Dans cette formule, DL et ERD correspondent, respectivement, aux valeurs des deux champs DL et ERD du message PDM reçu. ERD est initialisé par 0 lorsque le message PDM est créé pour la première fois par le noeud source.

Une fois cette valeur calculée, le noeud i calcule pour chacun voisin j dans son ensemble N_i^{MAX} l'avancement que j propose. Cette avancement est calculé comme suit :

$$Progress_j = HopToAcor_i - HopToActor_j \tag{3.3}$$

Dans cette formule, $HopToActor_j$ est récupérée à partir de l'élément qui correspond à j dans l'ensemble N_i^{MAX}.

Après le calcul de ces valeurs, le noeud i sélectionne son parent n comme suit :

$$n = \begin{cases} \arg\min_{j \in N_i^{MAX} \text{ et } Progress_j > DEP} Cost_j & si \; \exists j \in N_i^{MAX} \ni Progress_j \geq DEP \\ \arg\max_{j \in N_i^{MAX}} Progress_j & sinon \end{cases} \tag{3.4}$$

Le noeud i choisit, comme noeud parent n, parmi tous ses nœuds voisins assurant un avancement qui dépasse DEP, celui qui présente un chemin vers le noeud actionneur avec le plus petit coût énergétique. S'il ne trouve aucun noeud voisin qui assure un tel avancement qui est supérieur à DEP alors, il sélectionne celui qui assure le plus grand avancement.

Une fois ce noeud n sélectionné, le noeud i met à jour les champs du message (PDM) et l'envoie vers n. Ce même processus se répète jusqu'à ce que le message PDM arrive au noeud actionneur. À partir du moment, les données ultérieures vont emprunter le même chemin emprunté par le message PDM.

3.7 Conclusion

Dans ce chapitre, nous avons présenté les détails de notre protocole proposé LDER, particulièrement, tous les détails nécessaires à sa bonne compréhension, à savoir : les structures de données utilisées, les messages échangés et les étapes de son fonctionnement. Ce chapitre a présenté également une description brève de la problématique de notre travail et aussi quelques remarques qui concernent le protocole DEDA, celui sur lequel nous nous sommes basés pour proposé LDER.

Après cette description, le chapitre suivant présente le processus de validation de notre proposition ainsi que les résultats obtenus.

Chapitre 4

Validation

4.1 Introduction

Dans ce chapitre, nous présentons les détails propres à la validation de notre protocole LDER. Une validation par simulation a été réalisée à l'aide du simulateur J-sim [18].Notre propre protocole avec le protocole DEDA ont été les deux intégrés dans ce simulateur. ce dernier a été intégré pour des fins de comparaison. Ce chapitre présente la méthodologie de simulation suivi et les paramètres adoptés. Il présente également les résultats qui ont été obtenus sous forme d'instantanés et de courbes ainsi qu'une discussion sur ces résultats.

4.2 Simulateur J-sim

J-sim [18] est un simulateur réseau qui est souvent utilisé dans la littérature afin de valider les solutions proposées. Il présente une plateforme qui offre un cadre de développement de nouveaux protocoles et de l'évaluation de leurs performances. La construction de J-sim a été faite sur la base d'une architecture logicielle à base de composants appelée ACA (Autonomous Component Architecture). Le fait de se baser sur une telle architecture à base de composants présente l'avantage de pouvoir développer des composants individuels, par exemple des protocoles, qui peuvent être réutilisés dans d'autres applications.

J-sim propose via son cadre INET, toutes les facilités qui permettent l'implémentation de nouveaux protocoles. J-sim est doté d'un module qui est dédié spécialement pour les WSNs. Il définit trois types de nœuds : noeud capteurs, nœuds puits et nœuds cibles. Ces derniers sont des nœuds qui simulent des objets de notre vie qui génèrent des évènements (e.g. un véhicule mobile qui génère des vibrations au sol).

L'intégration de nouveaux protocoles dans J-sim et leur simulation se fait selon le processus suivant :

0. Comme première étape, il faut implémenter la solution visée à l'aide du langage Java. Cette implémentation doit se faire selon la philosophie de l'architecture ACA et aussi sur la base du cadre INET.
0. Une fois cette solution implémentée, il faut décrire le scénario de simulation souhaité à l'aide du langage Tcl\Java. Ce dernier est une extension du langage TCL qui présente l'avantage de pouvoir instancier et appeler à partir du TCL, des objets Java. En fait, c'est à ce niveau de description que les différents composants qui correspondent aux différents protocoles sont composés selon les simulations désirés.

0. Dans la dernière étape, il faut simplement lancer les différentes simulations définies et récupérer les résultats obtenus.

L'intégration de notre protocole LDER et aussi du protocole DEDA ainsi que leur simulation a été faite en suivant ce même processus.

4.3 Méthodologie et paramètres de simulation

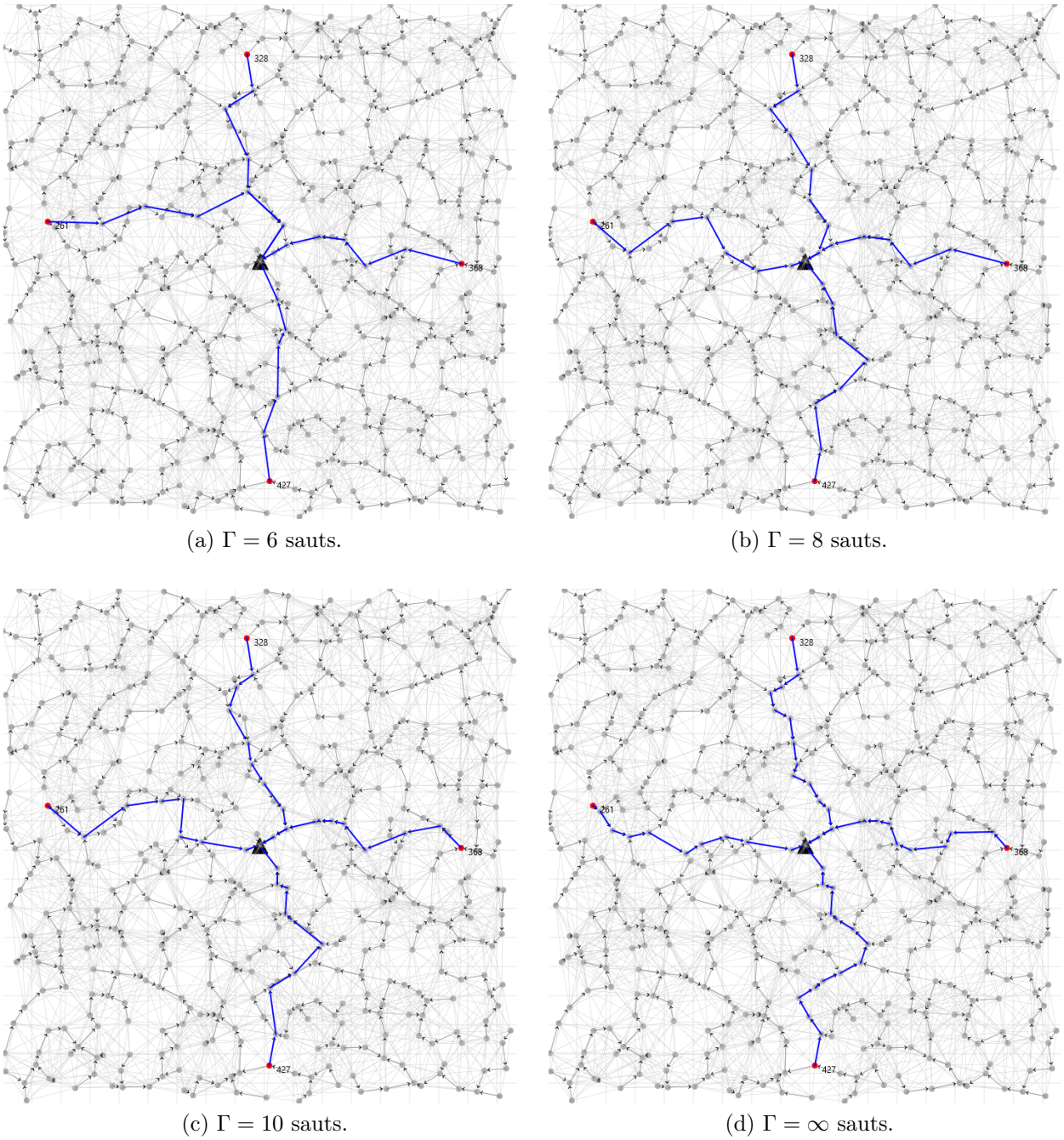
Table 4.1: Paramètres de simulation.

Paramètre	Valeur
Nombre de nœuds capteurs	500
Nombre de nœuds actionnaire	1
Nombre de nœuds sources	4
Densité	20
Portée maximale (m)	80
path loss (α)	2

Comme première étape de notre processus de validation, nous avons intégré notre protocole LDER et aussi le protocole DEDA dans le simulateur J-sim. Ce dernier protocole a été intégré pour des fins de comparaison. Diverses expérimentations ont été ensuite menées pour pouvoir obtenir des résultats. Le tableau (4.1) illustre les principaux paramètres de simulation adoptés. Comme le montre ce tableau, un réseau de 500 nœuds capteurs et un seul noeud actionneur a été supposé. 4 nœuds capteurs sources répartis à différents endroits du réseau ont été également considérés. L'aire du terrain de déploiement a été calculée suivant cette formule : $\sqrt{(n\pi r^2)/d}$. Dans cette dernière formule, n correspond au nombre de nœuds réseau, r à la portée de communication maximale des nœuds et d à la densité réseau, c.à.d. le nombre de noeud moyen dans une zone donnée. La valeur de r qui a été considérée est 80 m et celle de d est 20. La valeur de α a été mise à 2, valeur qui est communément utilisée dans les travaux connexes.

L'objectif principal que nous avons visé par les différentes d'expérimentations menées est de pouvoir montrer, d'un côté, la capacité de LDER à assurer le compromis entre la conservation de l'énergie et l'arrivée des données prélevées avant l'échéance imposée, et d'un autre côté, sa capacité à trouver des chemins qui ont des coûts énergétiques moins importants par rapport à ceux qui sont calculés par DEDA. Afin de montrer cela, différentes expérimentations ont été menées en supposant différentes bornes temporelles qu'il ne faut pas dépasser (valeurs de Γ). Les résultats obtenus sont présentés sous la forme :

- D'instantanés qui montrent visuellement les routes établies par chaque protocole selon les différentes valeur de Γ considérées. Ils montrent clairement les différents raccourcis ajoutés dans la topologie initiale.
- De courbes pour montrer les différents coûts (i.e. puissances de transmission totales) des routes construites par chaque protocole en fonction de la valeur de Γ.



(a) $\Gamma = 6$ sauts.

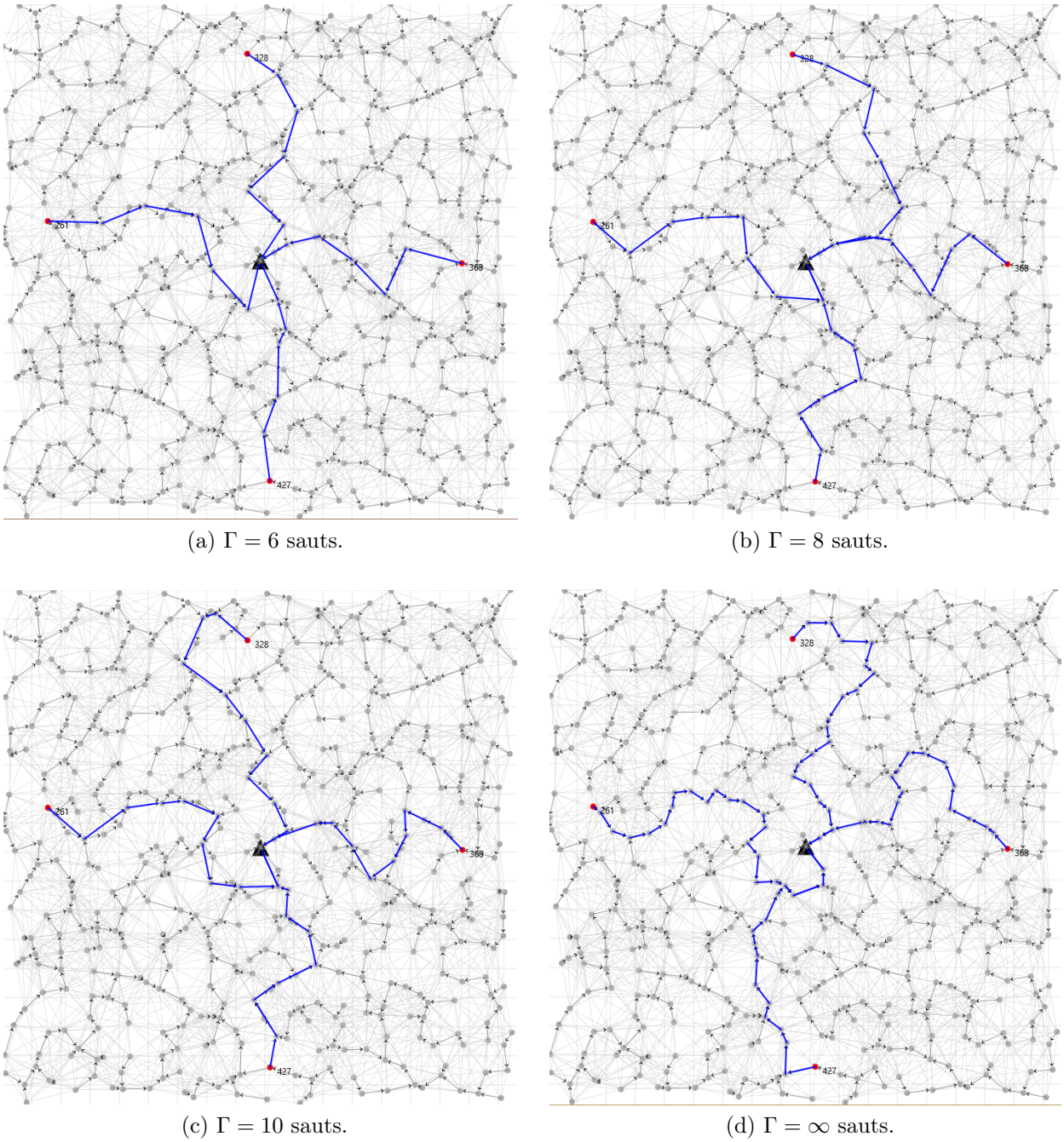
(b) $\Gamma = 8$ sauts.

(c) $\Gamma = 10$ sauts.

(d) $\Gamma = \infty$ sauts.

FIGURE 4.1: Instantanés montrant les différentes routes finales construites par notre protocole LDER avec différentes valeurs de Γ.

4.4 Résultats et discussion



(a) $\Gamma = 6$ sauts.

(b) $\Gamma = 8$ sauts.

(c) $\Gamma = 10$ sauts.

(d) $\Gamma = \infty$ sauts.

FIGURE 4.2: Instantanés montrant les différentes routes finales construites par le protocole DEDA avec différentes valeurs de Γ.

Différentes expérimentations ont été menées avec les deux protocoles LDER et DEDA avec quatre valeurs de Γ différentes : 6, 8, 10 et ∞. La figure (4.1) montre les instantanés obtenus par notre protocole LDER et la figure (4.2) montre ceux qui ont été obtenus par le protocole DEDA.

Particulièrement, Les quatre figures (4.1a), (4.1b), (4.1c) et (4.1d) visualisent les différentes routes obtenues après l'exécution de notre protocole LDER, respectivement, avec

les valeurs 6, 8, 10 et ∞ de Γ. À partir de ces figures, deux observations peuvent être citées. Comme première observation, nous remarquons que le nombre de sauts de chaque route construite ne dépasse jamais la valeur de Γ. Comme deuxième, nous constatons que le nombre de sauts moyen des routes augmente à chaque augmentation dans la valeur de Γ. Cela est tout à fait raisonnable car, si nous constatons bien dans les quatre figures, nous remarquons plus de raccourcis à chaque diminution dans la valeur de Γ. Cela est du au besoin de trouver, à chaque fois, des nœuds pouvant donner des avancements dans la topologie initiale qui permettent le non dépassement de l'échéance imposée. En fait, cette augmentation dans le nombre moyen de sauts, et comme le montre la figure (4.3), s'accompagne avec une conservation plus importante d'énergie car, l'ajout de raccourcis se traduit par l'ajout de liens non économes en énergie, un ajout qui se multiplie à chaque minimisation dans la valeur de Γ.

Les quatre figures (4.2a), (4.2b), (4.2c) et (4.2d) visualisent les différentes routes obtenues après l'exécution du DEDA, respectivement, avec les valeurs 6, 8, 10 et ∞ de Γ. Les deux observations remarquées dans les instantanés de notre protocole LDER sont également remarquées dans ceux de DEDA. Cependant, si nous comparons la figure (4.1d) avec la figure (4.2d), nous remarquons qu'avec DEDA, aucun raccourci a été ajouté et les routes construites sont complètement créées sur la topologie LMST car, la valeur de Γ est importante et aucun besoin d'emprunter des raccourcis ne surgisse. Cela n'est pas le cas avec notre protocole LDER où certain raccourcis sont ajoutés, non pas pour assurer l'arrivée à temps des données, mais afin d'emprunter d'autres chemins plus économes en énergie par rapport à ce que la topologie LMST propose. Cela est clairement illustré par la figure (4.3) qui montre la capacité de notre protocole LDER à trouver, avec chaque valeur de Γ, des chemins avec des puissances de transmission totales moins importantes par rapport à celles des chemins construits par DEDA.

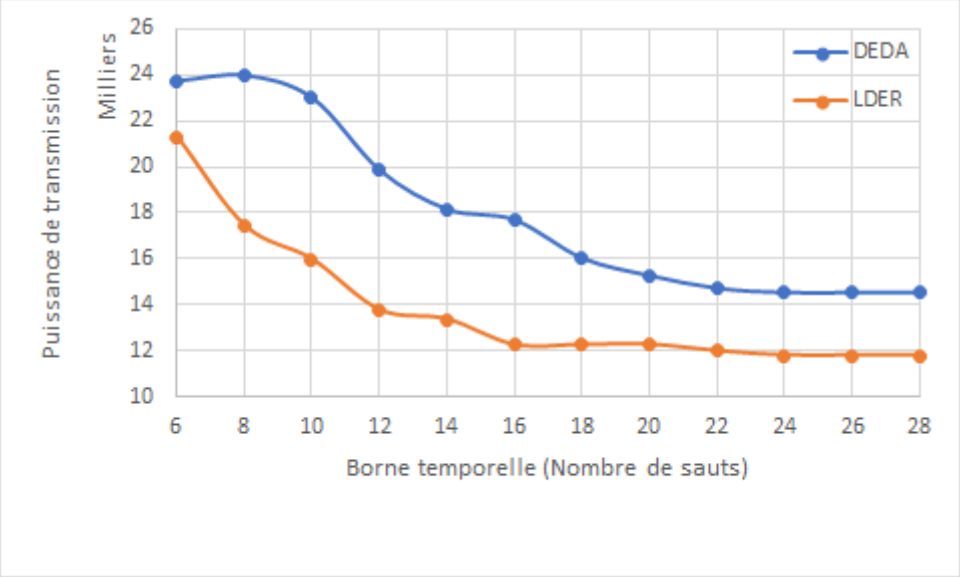


FIGURE 4.3: Puissance de transmission moyenne des routes finales construites par notre protocole LDER et aussi le protocole DEDA en fonction de Γ.

4.5 Conclusion

Dans ce chapitre, nous avons présenté les résultats de validation de notre protocole LDER avec le processus qui a été suivi pour arriver à ces résultats. Cette validation a été faite par simulation à l'aide du simulateur J-sim avec lequel, nous avons comparé LDER avec le protocole DEDA. Divers résultats sous la forme de courbes et d'instantanés ont été présentés. Ces résultats ont montré la supériorité de notre proposition.

Conclusion générale

Dans ce travail, nous nous sommes intéressés aux réseaux de capteurs et d'actionneurs sans fil. Avec ces réseaux, deux types de nœuds sont distingués. Les premiers sont des nœuds capteurs, caractérisés par des capacités restreintes de traitement, de stockage et de communication. Ces nœuds sont alimentés par des batteries qui limitent leur durée de vie. Le deuxième type de nœuds sont des nœuds actionneurs ayant des ressources plus importantes.

Le problème principal auquel nous nous sommes intéressés dans ce travail, été celui du routage des données captées par les nœuds capteurs vers les nœuds actionneurs appropriés. En fait, ce routage doit d'un côté, minimiser l'énergie qui est consommée lors de l'acheminement de ces données, et d'un autre côté, il doit assurer l'arrivée de ces données avant une certaine échéance donnée. Pouvoir assurer, en même temps, ces deux objectifs revient à pouvoir assurer un compromis entre la minimisation de la consommation énergétique des nœuds et la délivrance à temps des données prélevées.

Pour pouvoir répondre à cette problématique, nous avons proposé notre propre protocole, que nous avons baptisé LDER, comme acronyme de *Localized Delay-bounded and Energy-efficient Routing.* Notre proposition a été basée sur un autre protocole appelé DEDA qui ne procède pas correctement dans le cas des WSANs évènementiels. Notre protocole procède, dans un premier temps, à l'établissement d'une topologie éparse de la topologie initiale du réseau (celle obtenue avec les puissances de transmission maximales des nœuds) en appliquant l'algorithme LMST (Local Minimum Spanning Tree). Au-dessus de cette nouvelle topologie éparse, un arbre de plus court chemin qui relie chaque noeud actionneur avec ses nœuds capteurs correspondants est ensuite établi. À l'occurrence d'évènements, la route initiale dans l'arbre créé précédemment est modifiée, en ajoutant des raccourcis dans la topologie originale du réseau si, la non délivrance à temps des données est constatée, ou bien si, un autre chemin plus énergétiquement efficace peut être emprunté. En fait, c'est ce deuxième point qui n'a pas été considéré dans DEDA, ce dernier qui ne propose aucun moyen pour la détection de chemin plus économes en énergie à part ceux de la topologie LMST.

Afin de valider LDER, nous nous sommes procédés par simulation à l'aide du simulateur J-sim. LDER et DEDA ont les deux été complètement intégrés dans ce simulateur. Plusieurs expérimentations ont été menées afin d'un côté, montrer la capacité de LDER à assurer la balance entre les deux besoins cités précédemment, et d'un autre côté, montrer la supériorité de notre protocole en comparaison à DEDA. Les résultats obtenus qui ont été présentés sous formes de courbes et d'instantanés ont été positifs.

Comme perspectives, nous comptons améliorer LDER pour considérer, pas uniquement les puissances de transmission des nœuds capteurs, mais aussi, leurs énergies résiduelles. Cela permet d'augmenter la durée de vie du réseau et donc celle de l'application en main. Outres cette première perspective, nous visons à essayer d'autres topologies éparses, à part LMST, comme RNG (relative neighborhood graph) ou autres.

Bibliographie

[1] Xu Li, Shuo Yan, Chendong Xu, and Ivan Stojmenović. Localized delay-bounded and energy-efficient data aggregation in wireless sensor and actor networks. *Wireless Communications and Mobile Computing*, 11 :1603–1617, 12 2011.

[2] Alexandre Mouradian. *Proposition et vérification formelle de protocoles de communications temps-réel pour les réseaux de capteurs sans fil.* PhD thesis, INSA de Lyon, 2013.

[3] Bilel Romdhani. *Exploitation de l'hétérogénéité des réseaux de capteurs et d'actionneurs dans la conception des protocoles d'auto-organisation et de routage.* PhD thesis, INSA de Lyon, 2012.

[4] Djamila Bendouda. *Contrôle des réseaux de capteurs et actionneurs sans fil pour la supervision.* PhD thesis, 2018.

[5] Tommaso Melodia, Dario Pompili, Vehbi C. Gungor, and Ian F. Akyildiz. Communication and coordination in wireless sensor and actor networks. *IEEE Transactions on Mobile Computing*, 2007.

[6] Gary Hardy, Corinne Lucet, and Nikolaos Limnios. *Computing all-terminal reliability of stochastic networks with binary decision diagrams.* PhD thesis, 2005.

[7] Rahim Kacimi. *Techniques de conservation d'énergie pour les réseaux de capteurs sans fil.* PhD thesis, 2009.

[8] Nouha Sghaier. *Techniques de conservation de l'énergie dans les réseaux de capteurs mobiles : découverte de voisinage et routage.* PhD thesis, Université Paris-Est, 2013.

[9] Karel Heurtefeux. *Protocoles localisés pour réseaux de capteurs.* PhD thesis, INSA de Lyon, 2009.

[10] Nicolas Gouvy. *Routage g éographique dans les r éseaux de capteurs et actionneurs.* PhD thesis, Université des Sciences et Technologie de Lille-Lille I, 2013.

[11] Tommaso Melodia, Dario Pompili, Vehbi C. Gungor, and Ian F. Akyildiz. Communication and coordination in wireless sensor and actor networks. *IEEE Transactions on Mobile Computing*, 6(10) :1116–1129, 2007.

[12] Muhammad Farukh Munir and Fethi Filali. A novel self organizing framework for sanets. In *12th European Wireless Conference 2006 - Enabling Technologies for Wireless Multimedia Communications*, pages 1–7, 2006.

[13] Muhammad Munir and Fethi Filali. Analyzing the performance of a self organizing framework for wireless sensor-actuator networks. pages 66–73, 03 2007.

[14] Ghalib Shah, Muslim Bozyigit, and Faisal Hussain. Cluster-based coordination and routing framework for wireless sensor and actor networks. *Wireless Communications and Mobile Computing*, 11 :1140–1154, 08 2011.

[15] Hakki Bagci, Ibrahim Korpeoglu, and Adnan Yazıcı. A distributed fault-tolerant topology control algorithm for heterogeneous wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 26(4) :914–923, 2014.

[16] Ke Li and Chien-Chung Shen. Balancing transmission power and hop count in ad hoc unicast routing with swarm intelligence. In *2008 IEEE Swarm Intelligence Symposium*, pages 1–8. IEEE, 2008.

[17] Ning 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.

[18] Ahmed Sobeih, Jennifer C Hou, Lu-Chuan Kung, Ning Li, Honghai Zhang, Wei-Peng Chen, Hung-Ying Tyan, and Hyuk Lim. J-sim : a simulation and emulation environment for wireless sensor networks. *IEEE Wireless Communications*, 13(4) :104–119, 2006.