N° d'ordre: 18/2010-M/IN

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE HOUARI BOUMEDIENE
USTHB / ALGER

FACULTE D'ELECTRONIQUE ET D'INFORMATIQUE

MEMOIRE

Présenté pour l'obtention du diplôme de MAGISTER

En : INFORMATIQUE

Spécialité : Informatique Mobile

Par : Melle. IABBASSEN Dalila

SUJET

Dissémination des données dans les réseaux de capteurs sans fil

Soutenu publiquement le 14/07/2010, devant le jury composé de :

Mr- N. BADACHE, Professeur, USTHB.	**Président**
Mme- S. MOUSSAOUI, Maître de Conférences/A, USTHB.	**Directrice de thèse**
Mr- A. BELKHIR, Professeur, USTHB.	**Examinateur**
Mr- M. BENCHAIBA, Maître de Conférences/A, USTHB.	**Examinateur**

Remerciements

Je remercie énormément Dr. Samira MOUSSAOUI, ma directrice de thèse, pour m'avoir donnée la possibilité de travailler sur ce sujet de recherche très intéressant. Je la remercie pour son aide précieuse et ses conseils, qui m'ont permis de bien comprendre les différents aspects de ce sujet et de bien analyser les problèmes posés.

Je tiens aussi à remercier vivement les membres du Jury : Pr. Nadjib BADACHE, Pr. Abdelkader BELKHIR et Dr. BENCHAIBA, pour le temps et l'effort investis pour juger mon travail.

Je remercie énormément mes très chers parents et mes sœurs et frères, qui m'ont soutenu vivement durant toutes mes études. Leur amour et leurs conseils m'ont été extrêmement précieux durant toutes ces années.

Je voudrais aussi remercier mes amis de post-graduation avec qui j'ai partagé d'excellents moments. Toute ma gratitude à Leila MADI, Souhila AOUFI et Sarah BENKOUIDER, qui m'ont beaucoup aidée avec leurs analyses pertinentes et leur précieux soutien.

Mes sincères remerciements vont aussi à mes amies Amina GUERMOUCHE, Lamia BENAMARA et Soumia BELKENNICHE, qui m'ont aidée à acquérir une grande partie de ma documentation.

Enfin, je remercie mes amies Yasmine et Wassila ainsi que tous mes proches pour les moments heureux que nous avons partagés et qui m'ont rajouté un souffle supplémentaire afin d'accomplir ce travail.

Résumé

Les réseaux de capteurs sans fil - *Wireless Sensor Networks* (WSN) - sont considérés comme un type spécial de réseaux ad hoc dédié à une application bien précise, dans le but d'acquérir des données et les transmettre à une station de traitement.

Malgré que les réseaux de capteurs présentent des ressources matérielles limitées en termes de puissance de calcul, de mémoire stockage et d'énergie, ils soulèvent un intérêt grandissant de la part des industriels ou d'organisations civiles où la surveillance et la reconnaissance de phénomène physique sont une priorité. Par exemple, ces capteurs mis en réseau peuvent surveiller une zone délimitée pour détecter soit l'apparition d'un phénomène donné (apparition de vibrations, déplacement linéaire...) soit mesurer un état physique comme la température (détection des incendies en forêts) ou la pression.

Le paradigme de communication dans ce type de réseau est généralement du type N-vers-1, où les capteurs collectent des données de l'environnement qui les entoure (i.e., température, pression, ...), et les disséminent vers un point central, appelé puits. Ce puits a pour principal objectif la collecte des différentes données générées par les capteurs et éventuellement leur traitement et/ou agrégation.

Récemment, les agents mobiles ont été proposés pour une dissémination efficace des données dans les réseaux de capteurs sans fil. Dans les architectures client/serveur traditionnelles, les données de différentes sources sont transférées à une destination ; alors que dans les paradigmes avec agents mobiles, un code exécutable spécifique à une tache traverse les sources pertinentes pour assembler les données. Les agents mobiles peuvent être utilisés pour réduire considérablement le coût de communication, spécialement à travers de faibles bandes passantes, en déplaçant la fonction de traitement vers la donnée plutôt qu'apporter la donnée vers un processeur central.

Ce projet propose d'utiliser une approche client/serveur combinée avec une autre approche agents mobiles pour réduire et agréger les données dans une architecture planaire de réseau de capteurs sans fil. L'architecture proposée est dite Mobile Line Based Data Dissemination (MLBDD). Des simulations étendues montrent que MLBDD fait preuve d'une meilleure performance en terme de consommation d'énergie.

Summary

The wireless sensor networks (WSN) are considered as a special type of ad hoc networks dedicated to a specific application in order to acquire data and transmit them to a treatment node.

Although sensor networks have limited material resources in terms of computing power, memory storage and energy, they raise a growing interest from the industry or civic organizations where monitoring and recognition of physical phenomenon are a priority. For example, these sensors can be networked to monitor a defined area or to detect the occurrence of a given phenomenon (appearance of vibration, linear motion ...) is to measure a physical condition such as temperature (fire detection in forests) or pressure.

The paradigm of communication in this type of network is generally of the type N-to-1, the sensors collect data from the surrounding environment (ie, temperature, pressure, ...), and disseminate them to a central node (sink). The sinks' primary objective is the collection of data generated by different sensors and possibly their treatment and / or aggregation.

Recently, mobile agents have been proposed for efficient data dissemination in wireless sensor networks. In the client / server architectures traditional data sources are transferred to a destination, while in the mobile agent paradigms, executable code specific to a task through appropriate sources to gather information. Mobile agents can be used to significantly reduce the cost of communication, especially through low bandwidths, by moving the processing function to the data rather only bring the data to a central processor.

This project proposes to use a client/server approach combined with another mobile agent approach to reduce and aggregate the data in a planar architecture of wireless sensor network. The proposed architecture is called Mobile Line-Based Data Dissemination (MLBDD). Extensive simulations show that MLBDD demonstrated better performance in terms of energy consumption.

Sommaire

SOMMAIRE

Chapitre III. Dissémination des données basée Agents Mobiles

Chapitre IV. MLBDD (Mobile Line Based Data Dissemination)

Chapitre V. Evaluation des performances

LISTE DES FIGURES

Introduction générale

Introduction générale

Les réseaux de capteurs sans fil - *Wireless Sensor Networks* (WSN) - sont considérés comme un type spécial de réseaux ad hoc dédié à une application bien précise, dans le but d'acquérir des données et les transmettre à une station de traitement.

Malgré que les réseaux de capteurs présentent des ressources matérielles limitées en termes de puissance de calcul, de mémoire stockage et d'énergie, ils soulèvent un intérêt grandissant de la part des industriels ou d'organisations civiles où la surveillance et la reconnaissance de phénomène physique sont une priorité. Par exemple, ces capteurs mis en réseau peuvent surveiller une zone délimitée pour détecter soit l'apparition d'un phénomène donné (apparition de vibrations, déplacement linéaire...) soit mesurer un état physique comme la température (détection des incendies en forêts) ou la pression. Dans beaucoup de scénarios de gestion de crise (séismes, inondations,...) ces réseaux de capteurs peuvent permettre une meilleure connaissance du terrain afin d'optimiser l'organisation des secours, ou bien même renseigner précisément les scientifiques sur les causes d'un phénomène physique particulier.

Les scénarios usuels d'utilisation envisagent des milliers de capteurs que l'on pourra disperser pour surveiller des zones sensibles. Le facteur de résistance à l'échelle sera donc crucial. S'ajoute à ces difficultés le fait que les capteurs possèdent des ressources très limitées en terme de bande passante, mais aussi en terme d'autonomie de fonctionnement puisque dans la plupart des scénarios de déploiement, les capteurs fonctionneront avec de petites batteries. Cette limitation des ressources rend nécessaire une certaine forme de coopération à grande échelle où les interactions entre capteurs peuvent être extrêmement complexes. Ceci nécessite la mise en place d'un protocole au niveau de la couche middleware pour la dissémination et la récupération des données d'une manière efficace.

Afin de bien étudier ce sujet, nous avons organisé ce travail en cinq chapitres comme suit :

Dans le chapitre 1, nous décrivons les réseaux de capteurs sans fil, leurs applications et contraintes et nous définissons la dissémination des données que nous divisons en deux types : - Dissémination des données basée Client/Serveur : qui fera l'objet du chapitre 2, où nous détaillons ce modèle et nous l'illustrons avec trois protocoles : DD [Int-00], TTDD [FYe-02] et LBDD [Ben-07].

- Dissémination des données basée Agents Mobiles : que nous étudions dans le chapitre 3, où nous exposons ce paradigme et nous l'illustrons avec trois protocoles : MAWSN [Che-06], MADD [Che-07] et AbDD [Sha-08].

Dans le chapitre 4, nous proposons une nouvelle solution MLBDD (Mobile Line Based Data Dissemination). Celle-ci est une fusion des deux approches LBDD et MAWSN.

L'évaluation de notre solution est présentée dans le chapitre 5 que nous avons comparé à LBDD.

Les conclusions tirées de ce projet et les perspectives sont enfin présentées en fin du présent mémoire.

Chapitre I

La dissémination dans les réseaux de capteurs sans fil

Chapitre I. La dissémination dans les réseaux de capteurs sans fil

I.1. Introduction

Un capteur est un petit appareil autonome capable d'effectuer des mesures simples sur son environnement immédiat. L'utilisation de ces capteurs n'a rien d'une nouveauté, ceux-ci sont utilisés depuis longtemps dans des domaines comme l'aéronautique ou l'automobile.

Ce qui est novateur, c'est la possibilité pour ces capteurs de communiquer de manière radio (sans fil) avec d'autres capteurs proches (quelques mètres) et pour certains d'embarquer de la capacité de traitement (processeur) et de la mémoire. On peut ainsi constituer des réseaux de capteurs sans fil (RCSFs) qui collaborent sur une étendue assez vaste. Cette nouvelle technologie est considérée comme la fusion de deux grands pôles de l'informatique : les systèmes embarqués et les réseaux de communications sans fil.

I.2. Généralités sur les RCSFs

I.2.1. Définition d'un réseau de capteurs sans fil

Les réseaux de capteurs sans fil - *Wireless Sensor Networks* (WSN) - sont considérés comme un type spécial de réseaux ad hoc dédié à une application bien précise, dans le but d'acquérir des données et les transmettre à une station de traitement.

Les nœuds de ce type de réseaux consistent en un grand nombre de micro-capteurs (jusqu'à des centaines de milliers) capables de récolter et de transmettre des données environnementales (température, pression, humidité…) d'une manière autonome. La position de ces nœuds n'est pas obligatoirement prédéterminée. Ils sont dispersés aléatoirement à travers une zone géographique (champ de captage), qui définit le terrain d'intérêt pour le phénomène capté. Les données captées sont acheminées grâce à un routage multi-sauts à un nœud considéré comme un "point de collecte", appelé nœud puits (ou *sink*). Ce dernier peut être connecté à l'utilisateur du réseau via Internet ou un satellite. Ainsi, l'usager peut adresser des requêtes aux autres nœuds du réseau, précisant le type de données requises et récolter les données environnementales captées par le biais du nœud puits.

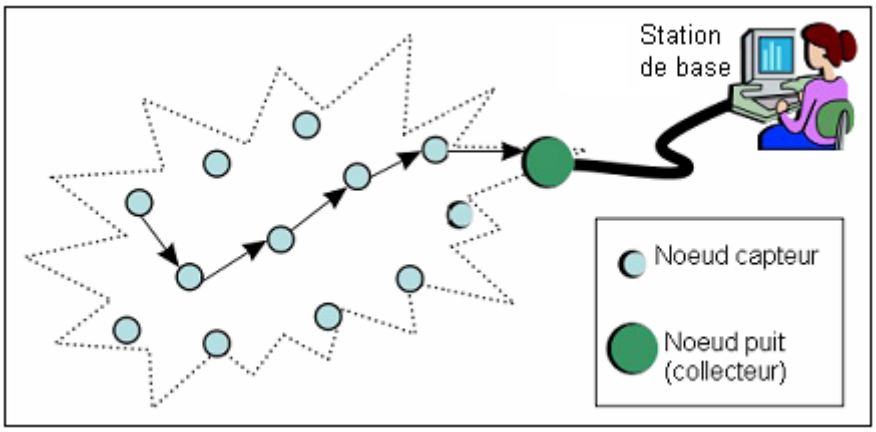

Figure I.1: Exemple d'un réseau de capteurs sans fil.

I.2.2. Composition du réseau de capteurs sans fil

Le déploiement d'un réseau de capteur nécessite 3 types d'éléments (voir figure I.1):

Le capteur : C'est l'élément de base pour l'acquisition des données. Il est caractérisé par sa petite taille et son faible coût mais avec des ressources très limitées (énergie, bande passante et puissance de calcul).

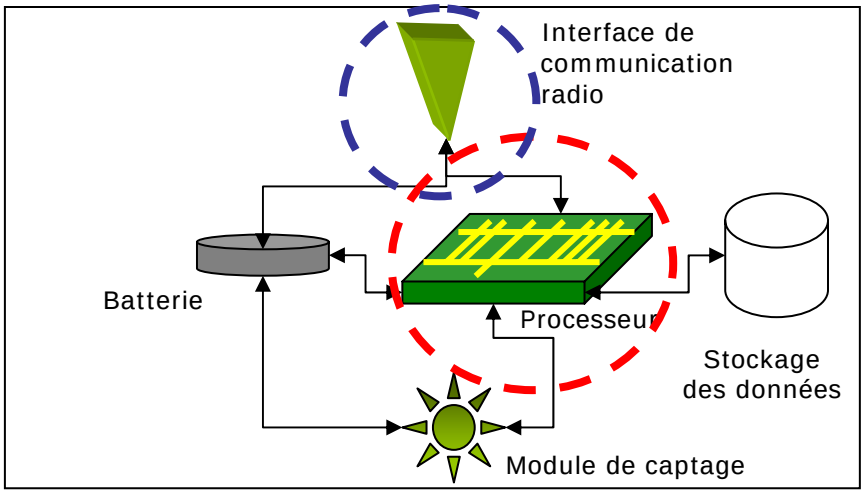

Figure I.2: Anatomie d'un nœud capteur.

Le rôle du nœud capteur est :

- Le captage des données : L'unité de captage englobe généralement deux sous-unités : le capteur lui-même, tels que les capteurs séismiques, thermiques, visuels, infrarouges,

acoustiques et radar qui sont capables de surveiller une grande variété de phénomènes ambiants (tels que la distance, lumière, sons, température, pression, débit, courant, déplacements…etc.), en plus d'un convertisseur analogique-numérique qui transforme les signaux analogiques produits par les capteurs, et basés sur le phénomène observé en signal digitale, ce dernier est transmis par la suite à l'unité de traitement.

- Traitement des données : Le micro-capteur contient une unité de traitement composée d'un processeur associé généralement à une petite unité de stockage et fonctionne à l'aide d'un système d'exploitation spécialement conçu pour les micro-capteurs (*TinyOS* par exemple). Cette unité est chargée d'exécuter les protocoles de communications qui permettent de faire collaborer le nœud avec les autres entités du réseau. Elle peut aussi analyser les données captées pour alléger la tâche du nœud puits.

- La communication : Le nœud capteur doit envoyer les données collectées au nœud puits, et ceci va se faire via l'interface de communication sans fil à travers plusieurs sauts. Le nœud capteur doit donc jouer le rôle d'un routeur afin de faire parvenir les données au sink.

L'unité de communication sans fil peut fonctionner en mode émission, réception, idle et sleep. Il est important d'éteindre complètement l'interface de communication plutôt que de la mettre en mode idle lorsque celle-ci ne fait ni réception ni émission, car ce mode consomme beaucoup d'énergie (voir figure I.3).

Un nœud capteur peut contenir d'autres unités dépendantes de l'application du réseau. En effet, la plupart des opérations de captage et des algorithmes de routage dans les réseaux de capteurs sans fil requièrent la connaissance de la localisation des nœuds avec une grande précision, car ces nœuds sont déployés d'une manière aléatoire et fonctionne d'une façon autonome, ceci rend l'intégration d'une unité, consacrée au système de localisation, très commune dans un nœud capteur.

D'où, il est souvent supposé que ces nœuds possèdent un système de localisation GPS avec une précision au moins égale à 5m [LLi-01]. La conception des nœuds capteurs peut aller jusqu'à prévoir un système de mobilisation du capteur pour le déplacer en cas de nécessité.

Toutes ces unités peuvent exiger leur intégration dans un boîtier de taille minimale inférieure à un centimètre cube, et avec un poids très léger qui permet aux nœuds de rester suspendus dans l'air, si l'application l'exige.

Le collecteur (puits ou sink): Les collecteurs sont un autre type de nœuds du réseau qui permettent de rassembler les données provenant de plusieurs capteurs et qui possèdent également des fonctions de passerelle, reliant ainsi les capteurs avec un réseau externe (Internet par exemple). Ces derniers possèdent beaucoup plus de capacités que les nœuds capteurs tant au niveau de la mémoire que de la vitesse de traitement ou des réserves en énergie [Sen-06].

Ce nœud est responsable, en plus de la collecte des rapports, de la diffusion des demandes sur les types de données requise aux capteurs via des messages de requêtes. Un réseau de capteurs peut contenir plusieurs nœuds puits diffusant des intérêts différents. Par exemple, un nœud puits peut demander à tous les capteurs se trouvant dans la région nord du champ de captage d'envoyer un rapport de température chaque minute, pendant qu'un autre peut être intéressé seulement par les hautes températures (> 50° C) dans la région sud. Par conséquent, un capteur doit pouvoir stocker toutes les requêtes reçues, et les traiter séparément [Khe-04].

La station de base : C'est l'élément recueillant les données et/ou les analyses du réseau et servant également à son administration. Ça peut être un PC classique, portable, Palm, etc.

Ainsi les données détectées vont parcourir le réseau de capteur en capteur jusqu'à la station de base. En retour, celle-ci peut demander l'exécution d'une tâche particulière à un capteur donné [Sen-06].

I.2.3. Domaines d'applications des réseaux de capteurs sans fil

Comme beaucoup de technologie, le développement des WSNs a été suscité par des besoins militaires. En effet, les armées souhaitent être en mesure d'espionner discrètement leurs ennemis. L'absence de câbles entre les nœuds, leur faible taille, le nombre élevé de nœuds déployables – pour couvrir une zone étendue – répondent à ces critères. Ainsi plusieurs applications ont été réalisées dont le *Sound Surveillance System* ou le *Distributed Sensor Network* [Khe-05].

Puis, la diminution des coûts de fabrication des capteurs ainsi que la réduction de leur taille a entraîné une utilisation dans des applications civiles. Aujourd'hui, les réseaux de capteurs ont envahit plusieurs domaines d'applications. Ils peuvent par exemple être utilisés à des fins de surveillance environnementale. En effet, des capteurs peuvent être placés dans des régions glaciaires ou tropicales afin de suivre de manière précise les effets du réchauffement

de la planète, les changements climatiques ou l'augmentation de la pollution. Dans les champs agricoles, les capteurs peuvent être semés avec les graines. Ainsi, les zones sèches seront facilement identifiées et l'irrigation sera donc plus efficace. Sur les sites industriels, les centrales nucléaires ou dans les pétroliers, des capteurs peuvent être déployés pour détecter des fuites de produits toxiques (gaz, produits chimiques, éléments radioactifs, pétrole, etc.) et alerter les utilisateurs dans un délai suffisamment court pour permettre une intervention efficace. Ils peuvent également être employés pour une surveillance de l'habitat car leur déploiement, par exemple en montagne, pourrait permettre de recenser les animaux fréquentant un territoire donné. Nous citons comme exemple, un biologiste voulant savoir l'existence d'un certain oiseau, et lorsque l'oiseau est détecté, il doit être suivi d'aussi près que possible. Dans ce cas, le réseau de capteurs est utilisé pour la reconnaissance automatique de cible et son suivi.

Dans certaines applications potentielles, les gens peuvent interroger le réseau de capteurs pour leur intérêt, tel que "Est-ce que Johannes est dans son bureau?", ou "Est-ce qu'il y a un siège vide dans la salle de réunion?".

Exemple de traitement d'une tâche dans un RCSF [Int-00]

Un utilisateur d'un réseau de surveillance à distance sera en mesure de contacter (en utilisant, peut-être une liaison radio à longue distance), l'un des capteurs dans le champ, et poser la tâche suivante: « Chaque I ms pour les prochaines T secondes, envoyez-moi une estimation de localisation de n'importe quel animal à quatre pattes dans la sous-région R du champ de capture ». En général, le réseau devrait supporter une variété de types de tâche. Toutefois, les réseaux de capteurs sont orientés tâche contrairement aux fins générales des communications réseaux, les types de tâche à accomplir sont connus au moment du déploiement du réseau de capteurs.

En utilisant les communications sans fil saut par saut, cette tâche est transmise aux nœuds dans la sous-région R du champ de capture. Chaque nœud recueille des échantillons via ses capteurs, et les rassemble pour les comparer à la forme d'onde stockée localement. Si le nœud détecte une forme d'onde semblable à un animal à quatre pattes, il génère des descriptions d'événement chaque milliseconde, qui contient les éléments suivants: son propre emplacement, une valeur de codes correspondant à l'animal, l'intensité du signal, et un certain degré de confidence dans son estimation. Les capteurs au sein de la région R doivent

coordonner pour sélectionner la meilleure estimation possible. Cette estimation est alors dirigée vers le demandeur de la tâche.

Aussi, l'utilisation des réseaux de capteurs dans le domaine de la médecine pourrait apporter une surveillance permanente des patients et une possibilité de collecter des informations physiologiques de meilleure qualité, facilitant ainsi le diagnostic de quelques maladies. En effet, cela peut servir à la surveillance du niveau de glucose, le monitoring des organes vitaux ou la détection de cancers. Mais ceci n'est pour l'instant qu'une utilisation future qui dépend encore grandement des progrès à venir de cette technologie.

Aujourd'hui, plus d'applications spécifiques dans différents domaines se posent, et au lieu de déployer des réseaux de capteurs conçu uniquement pour des applications spécifiques, les futurs réseaux vont avoir des nœuds avec différents capteurs physiques pour une grande variété de scénarios d'application et différents groupes d'utilisateurs (Les MICA motes contiennent déjà des capteurs de température, un magnétomètre, un accéléromètre, un microphone, et aussi plusieurs actionneurs).

I.2.4. Caractéristiques et contraintes des réseaux de capteurs sans fil

Alors même qu'un grand nombre d'applications mettent en jeu des WSNs, ceux-ci ont plusieurs restrictions que ces applications doivent contourner. Par exemple, ils ont une faible puissance de calcul, une réserve d'énergie limitée et une bande passante réduite. Nous détaillerons quelques facteurs des WSNs dans ce qui suit :

I.2.4.1. L'environnement : Les capteurs sont déployés soit à un endroit très proche du phénomène observé ou alors directement dans le phénomène lui-même. En effet, les capteurs sont souvent déployés en masse dans des endroits hostiles tels que des champs de bataille au delà des lignes ennemies, à l'intérieur de grandes machines, au fond d'un océan, dans des champs biologiquement ou chimiquement souillés, etc. Par conséquent, ils doivent pouvoir fonctionner sans surveillance dans des régions géographiquement éloignés ou inaccessibles.

I.2.4.2. Passage à l'échelle : La surveillance d'un phénomène peut nécessiter le déploiement d'un nombre de nœuds qui est de l'ordre de plusieurs milliers de capteurs. Suivant l'application, ce nombre peut encore augmenter jusqu'à des millions de capteurs, ce qui entraîne des congestions et des erreurs de communication. Un tel déploiement,

nécessite que le protocole utilisé pour la communication soit capable de détecter les erreurs et de contrôler le flux il doit aussi exploiter la nature fortement dense des réseaux de capteurs. Cette densité peut varier entre quelques capteurs jusqu'à plusieurs centaines de capteurs dans une région de taille inférieure à 10 mètres de diamètre [Aky-02].

La densité des nœuds dépend également de l'application pour laquelle le réseau de capteurs est employé. Une application de diagnostic de machines nécessite par exemple une densité proche de 300 nœuds par région de 25 m^2, tandis que la densité nécessaire pour le contrôle des véhicules ne peut pas dépasser 10 capteurs par une région de même taille [Aky-02].

I.2.4.3. Topologie du réseau : Les caractéristiques de déploiement aléatoire, fonctionnement autonome, et fréquence élevée de pannes rendent la maintenance de la topologie d'un réseau de capteurs une tâche complexe. En effet plusieurs centaines de capteurs sont déployés avec une densité pouvant être supérieur à 20 nœuds par m^3, ceci exige une bonne gestion de la maintenance de la topologie du réseau déployé. Cette maintenance de topologie se fait en trois phases :

- **Déploiement :** Les nœuds capteurs peuvent être éparpillés sur le champ de captage en masse ou placés d'une manière individuelle et ceci par le biais de plusieurs moyen tels que : les jeter d'un avion, utiliser une artillerie, roquette ou missile, ou les placer nœud par nœud d'une façon manuelle ou en utilisant des robots.

- **Post-déploiement :** Après la phase de déploiement, la topologie du réseau peut subir des changements dus aux : changement de position des nœuds, non accessibilité à cause du brouillage ou des obstacles en mouvements, épuisement d'énergie, mal fonctionnement des nœuds, ou pour des besoins de l'application.

  En effet, Bien que les nœuds d'un réseau de capteurs peuvent être déployés d'une manière statique, la panne matérielle constitue un évènement très commun à cause de l'épuisement d'énergie ou la destruction. Il est possible également d'avoir un réseau de capteur avec des nœuds mobiles qui ont une mobilité très élevée. Par conséquent, la topologie du réseau de capteur est exposée fréquemment aux changements après la phase de déploiement.

- **Redéploiement de nœuds additionnels :** Des nœuds capteurs additionnels peuvent être installés pour remplacer ceux qui sont en panne ou bien pour répondre aux besoins des tâches assignées au réseau. Cette addition entraîne la réorganisation du réseau et le changement de sa topologie. Ainsi, Une bonne gestion du réseau, faisant face au facteur de changement fréquent de la topologie d'un réseau ad hoc caractérisé par une contrainte exigeante de consommation d'énergie doit passer obligatoirement par la conception de protocoles de dissémination et de routage spéciaux pour ce type de réseau.

I.2.4.4. Tolérance aux pannes : Certains nœuds capteurs peuvent générer des erreurs ou ne plus fonctionner à cause d'un manque d'énergie, d'une défaillance matérielle ou d'une interférence. Ceci ne doit pas affecter la globalité de la tâche du réseau de capteurs. En cas de défaillance, de nouveaux liens et routes doivent être établis pour assurer la collecte des données. La redondance peut également être utilisée, tout en veillant à conserver une faible consommation d'énergie.

Le critère de tolérance aux fautes dépend essentiellement de l'environnement de déploiement du réseau. En effet, si le réseau de capteurs est destiné aux environnements avec un faible degré d'interférences, tel que ceux utilisés dans les bâtiments pour surveiller le taux d'humidité et le degré de température, les protocoles utilisée ne doivent pas cibler une grande tolérances aux pannes, car dans ce type de réseau, il n'existe pas une grande interférence avec l'environnement, et ses nœuds ne sont pas exposés au risque d'endommagement.

Par contre, si le réseau est destiné aux applications militaires telles que la surveillance et le contrôle d'un champs de bataille, le niveau de tolérance aux pannes visé par les protocoles employés doit être très élevé, car les nœuds sont exposés à un grand risque d'endommagement par des actions hostiles, et les informations captées sont très critiques.

Par conséquent, le niveau de tolérance aux pannes requis dépend de l'application du réseau de capteurs conçu, et les schémas de conception doivent prendre en charge ce paramètre.

I.2.4.5. Les contraintes matérielles : Malgré leur petite taille et leur faible coût, les capteurs subissent de fortes contraintes notamment d'énergie, de calcul et de stockage.

- **Dimension :** La taille réduite des capteurs peut présenter de nombreux avantages en fonction de l'utilisation prévue du système, et elle permet un déploiement flexible et simple du réseau. Cependant, la puissance des batteries utilisées pour alimenter les nœuds capteurs est limitée, par la petite taille de ces derniers.

- **Énergie :** Comme les nœuds capteurs sont des composant micro-électroniques, ils ne peuvent être équipés que par des sources limitées d'énergie (<0.5 Ah, 1.2 V). De plus, dans certaines applications, ces nœuds ne peuvent pas être dotés de mécanismes de rechargement d'énergie, par conséquent, la durée de vie d'un nœud capteur dépend fortement de la durée de vie de la batterie associée.

  Sachant que les réseaux de capteurs sont basés sur la communication multi-sauts, chaque nœud joue à la fois un rôle d'initiateur de données et de routeur également. La baisse d'énergie d'un certain nombre de nœuds entraîne un changement significatif sur la topologie globale du réseau, et peut nécessiter un routage de paquets différent et une réorganisation totale du réseau. C'est pour cela que le facteur de consommation d'énergie est d'une importance primordiale dans les réseaux de capteurs. La majorité des travaux de recherche menés actuellement se concentrent sur ce problème afin de concevoir des algorithmes et protocoles spécifiques à ce genre de réseau qui consomment le minimum d'énergie.

  En effet, dans les réseaux ad hoc classiques, la consommation d'énergie est un facteur important mais ne constitue pas la première considération pour les concepteurs, car les batteries sont supposées toujours remplaçables par l'utilisateur, les chercheurs ont cependant concentré leurs efforts sur les facteurs de qualité de service dans ce type de réseau, tel que le débit de transmission et la tolérance aux panne.

  Par contre, Dans les réseaux de capteurs, l'efficacité en consommation d'énergie représente une métrique de performance significative, qui influence directement sur la durée de vie du réseau en entier. Pour cela, les concepteurs peuvent au moment du développement de protocoles négliger les autres

métriques de performance telle que la durée de transmission et le débit, au détriment du facteur de consommation d'énergie.

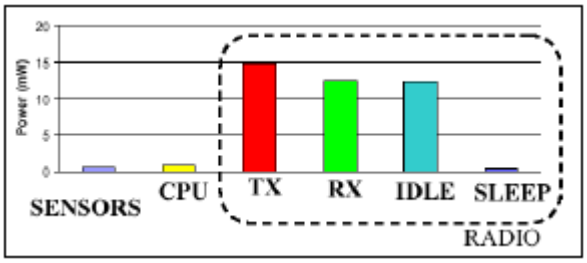

Figure I.3 : Consommation d'énergie dans un nœud capteur [Est-02].

L'énergie du nœud capteur est consommée par trois éléments : le captage, le traitement des données et la communication. Le taux d'énergie consommée dans chacun de ces trois éléments dépend de l'application du réseau de capteurs. Mais généralement, c'est la communication qui consomme beaucoup plus d'énergie que les autres tâche du nœud capteur (voir figure I.3).

- **Puissance de traitement :** Malgré la construction des processeurs de plus en plus petits avec une puissance de calcule plus élevée, les unités de calcul et de stockage mémoire restent toujours une ressource rare pour les nœuds capteurs. Par exemple les nœuds Smart-Dust possèdent une unité de traitement avec une fréquence de 4MHZ et 512 octets de mémoire en plus d'une EEPROM de 512 octets également. Le système d'exploitation exécuté sur ces nœuds est le TinyOS, ce dernier possède 3500 octets pour le code du système, en plus d'un espace de 4500 octets disponibles pour d'autres programmes.

I.2.5. Différences entre les réseaux de capteurs et les réseaux AdHoc classiques

La réalisation des différentes applications liées aux réseaux de capteurs requièrent l'utilisation des techniques employées dans les réseaux ad hoc sans fil, cependant, la multitude des protocoles proposée pour les réseaux ad hoc traditionnels, ne peut pas être directement appliquée aux réseaux de capteurs, a causes des caractéristiques uniques de ces derniers, et les exigences imposées par leurs applications.

Nous distinguons plusieurs critères faisant la différence entre les réseaux de capteurs et les réseaux ad hoc conventionnels, entre autres :

- Le nombre de nœuds est nettement plus grand dans les réseaux de capteurs que dans les réseaux ad hoc.
- Les nœuds capteurs sont déployés d'une manière dense.
- Les nœuds capteurs sont plus exposés aux pannes.
- Les réseaux de capteurs utilisent principalement les communications broadcast alors que la plupart des réseaux ad hoc sont basés sur les communications point à point.
- Les nœuds capteurs sont caractérisés par des ressources plus limités (ressource d'énergie, puissance de calcul et mémoire).
- Les nœuds capteurs ne possèdent aucune identification globale telle que les adresses IP dans les réseaux ad hoc.

Aussi, dans les réseaux Ad hoc traditionnels, les tâches qui traitent l'organisation, le routage, et la gestion de mobilité visent l'optimisation des différents paramètres de qualité de service (QoS) tel que l'efficacité dans le débit et les délais de transmission sous la contrainte de mobilité. La consommation d'énergie est d'une importance secondaire, puisque les batteries des unités mobiles utilisées peuvent être facilement remplacées. Cependant, les réseaux de capteurs englobent un grand nombre de nœuds possédant des sources d'énergie irremplaçables à cause de leur utilisation distante non assistée dans les environnements hostiles. Ces capteurs communiquent entre eux avec un taux de transmission très faible de l'ordre de 1 à 100 kbps. Pour cela, et contrairement aux réseaux ad hoc classiques, le but principal des techniques utilisées est de prolonger la durée de vie des batteries afin de prévenir les dégradations de connectivité dans le réseau.

I.3. Classification des réseaux de capteurs sans fil

I.3.1. Topologie du réseau

La topologie détermine l'organisation des capteurs dans le réseau. Il existe deux principales topologies dans les protocoles de routage pour les WSNs [Chl-05] :

Topologie plate : Un réseau de capteurs sans fil plat est un réseau homogène, où tous les nœuds sont identiques en termes de batterie et de complexité du matériel et ont le même rôle, excepté le nœud puits qui joue le rôle d'une passerelle et qui est responsable de la transmission de l'information collectée à l'utilisateur final. Selon le service et le type de

capteurs, une densité de capteurs élevée (plusieurs nœuds capteurs/m²) ainsi qu'une communication multi-sauts peut être nécessaire pour l'architecture plate.

Ce type de solution permet une grande tolérance aux pannes, cependant elle souffre d'une faible scalabilité. En effet, si tous les nœuds opèrent de la même façon et d'une manière distribuée, on aura un grand nombre de messages de contrôle nécessaires pour le bon fonctionnement du réseau.

Topologie hiérarchique : Afin d'augmenter la scalabilité du système, les topologies hiérarchiques ont été introduites en divisant les nœuds en plusieurs niveaux de responsabilité. L'une des méthodes les plus employées est le clustering, avec laquelle le réseau est partitionné en groupes appelés clusters. Un cluster est constitué d'un chef (*clusterhead*) et de ses membres. Suivant l'application, les membres peuvent être des voisins directs du chef ou pas.

Avec une approche hiérarchique, il est plus facile d'intégrer un mécanisme d'agrégation au système : les nœuds membres transmettent leurs données vers le clusterhead, qui va par la suite agréger ces lectures afin de transmettre le résumé à la station de base. Pour augmenter la tolérance aux pannes, la sélection des chefs doit être dynamique afin d'éviter d'avoir des membres sans clusterhead.

Cette architecture sans fil est influencée par un certain nombre de facteurs et contraintes tels que la tolérance aux fautes, le redimensionnement, les coûts de production, l'environnement, les contraintes matérielles, les médias de transmission et la consommation d'énergie.

L'inconvénient de la hiérarchisation est la surcharge des clusterheads, induisant à un déséquilibre de la consommation d'énergie.

I.3.2. Schéma de communication

L'un des points clés de différence entre les réseaux ad hoc et les réseaux de capteurs est le modèle de communication. Dans un réseau ad hoc, les communications sont point à point. Dans un WSN, la plus part des communications sont du type many-to-one, ce qui rend le routage ad hoc et les différents mécanismes de communications (sécurité, optimisation, ... etc) incompatibles. En réalité, un WSN doit assurer trois schémas de communication [Val-05]:

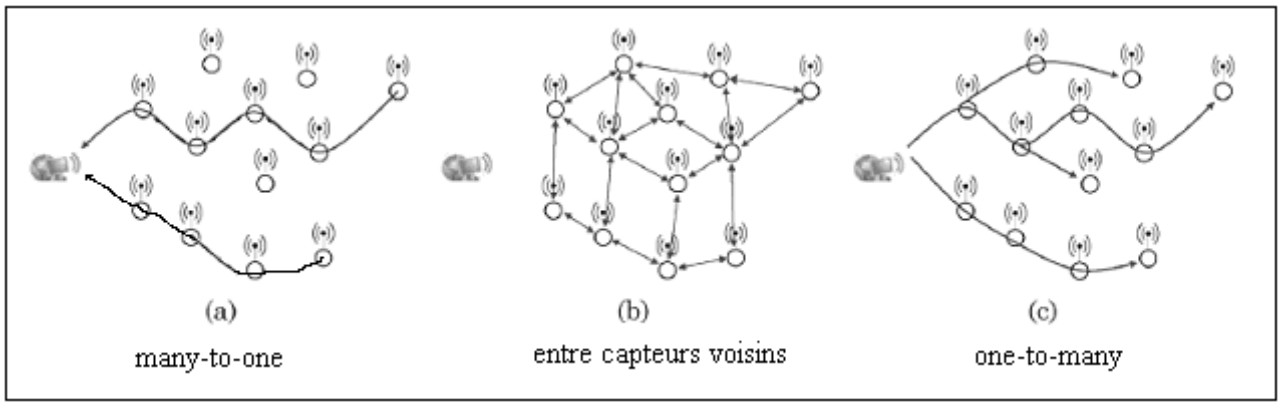

Figure I.4 : Schéma de communication dans les RCSFs.

- **Many-to-one :** il est généralement utilisé pour acheminer les rapports de données vers le puits. En raison des redondances, les lectures doivent être agrégées dans des zones particulières du réseau.

- **Communications locales** : sont les communications à un saut entre des capteurs voisins.

- **One-to-many :** ce schéma est utilisé par le puits pour communiquer avec ses capteurs, afin de propager des requêtes ou contrôler le fonctionnement des nœuds. Ces communications peuvent être générales ou ciblées : i.e. destinées vers une certaine région du réseau.

I.3.3. Paradigme de communication

Pour les WSNs, il existe trois principaux paradigmes de communication [Nic-05] :

Node-centric : Ce paradigme est celui employé dans les réseaux conventionnels, où les communications se basent sur l'identification des nœuds participants, qui se fait à l'aide d'adresses IP. Les différents mécanismes de propagation se focalisent sur un découpage hiérarchique logique des nœuds en sous-réseaux. Les réseaux ad hoc utilisent ce genre de paradigme, qui s'intègre bien avec l'utilisation de ce type d'environnement. Cependant, pour les réseaux de capteurs, un routage basé sur une identification individuelle des nœuds ne reflète pas l'usage réel du réseau. Pour cela, un autre paradigme a été introduit : data-centric. Néanmoins, le paradigme node-centric n'est pas à écarter totalement, car certaines applications nécessitent une interrogation individuelle des capteurs. Dans ce cas, le déploiement d'une propagation data-centric n'aura aucune valeur ajoutée, et consommera éventuellement plus de ressources.

Data-centric : Dans un WSN, la donnée est plus importante que le nœud lui-même, ce qui rend son identification inutile. Dans le paradigme data-centric, les communicants sont identifiées par leurs données, et donc tout le système (routage, interrogation, . . . etc.) doit être régit par cette propriété. Ainsi, le système peut être vu comme une base de données distribuée, où les nœuds forment des tables virtuelles, alimentées par les données captées. En abandonnant l'identification des nœuds, un capteur perd sa valeur individuelle, qui sera attribuée aux données de la zone, formée d'un ensemble de capteurs.

Position-centric : Dans cette approche, les positions des nœuds représentent le moyen principal d'adressage et de routage. Dans certaines applications, il est plus intéressant d'interroger le système en utilisant les positions des nœuds, que leurs adresses IP. Dans ce cas, le routage s'effectue grâce à des techniques géométriques afin d'acheminer l'information d'une zone géographique vers une autre. Cependant, ce type de mécanismes nécessite un déploiement d'une solution de positionnement, dont le degré de précision requis dépend de l'application ciblée. L'utilisation du GPS reste trop coûteuse pour un WSN. Pour cela, plusieurs méthodes ont été développées qui permettent de positionner un capteur en connaissant les coordonnées d'un ensemble réduit d'entités appelées seed.

I.3.4. Type d'application

La méthode de capture des données dans un WSN dépend de l'application et de l'importance de la donnée. De ce fait, les WSNs peuvent être catégorisés comme *time-driven*, *event-driven ou query-driven.*

Application time-driven : Un réseau time-driven est approprié pour des applications qui nécessitent un prélèvement périodique des données. Par exemple, cela est utile dans des applications de monitoring (feu, météo) afin d'établir des rapports périodiques. Dans ce cas, le routage est proactif et nécessite une maintenance continue des chemins. En raison de cet aspect périodique, les capteurs peuvent se mettre en veille pendant les périodes d'inactivité, et n'enclencher leur dispositif de capture qu'à des instants particuliers. Cet apport permet d'allonger la durée de vie du réseau.

Application event-driven : Dans des applications temps réel, les capteurs doivent réagir immédiatement à des changements soudains des valeurs captées. Un prélèvement périodique des données est inadapté pour ce type de scénarios. Pour cela, le protocole doit être réactif et doit donner des réponses rapides à l'occurrence d'un certain nombre d'évènements.

Application query-driven : Le modèle query-driven est semblable au modèle event-driven sauf que la collecte des informations sur l'état de l'environnement est initiée par des interrogations envoyées par le puits. La plupart des applications query-driven sont des applications interactives, critiques et leur tolérance aux délais dépend de l'urgence de l'interrogation.

Notons que le modèle query-driven peut être utilisé pour contrôler et reconfigurer les nœuds. Par exemple, le puits peut envoyer des commandes au lieu d'interrogations pour modifier le programme d'un nœud capteur, modifier son taux de trafic ou son rôle.

I.4. Dissémination des données dans les RCSFs

I.4.1. Dissémination des données

Bien qu'ils partagent de nombreux problèmes communs avec les réseaux mobiles ad hoc (MANET), le principal problème des réseaux de capteurs sans fil est de savoir comment fournir les économies d'énergie pour prolonger la durée de vie du réseau. Parmi tous les aspects de la consommation d'énergie, la communication des données est considérée comme le principal consommateur d'énergie.

L'objectif de la dissémination des données est d'envoyer n'importe quels types d'informations (données ou requêtes) à tous les nœuds concernés par ces informations, tout en minimisant le nombre de nœuds de transmission et le coût d'énergie [Car-05][Mou-07]. La dissémination des données est considérée comme une des phases principales de consommation d'énergie dans la communication des WSNs [JLu-07]. D'où la façon d'éliminer le trafic des données redondantes et de réduire les coûts de communication sont les principaux défis dans la dissémination des données. Concernant les applications des WSNs telles que la surveillance de l'environnement, la dissémination des données est très importante. En effet, Il est connu que l'inefficacité de la dissémination des données (comme les diffusions aveugles) causent des tempêtes de diffusion et bloquent toutes les communications des données dans le réseau [SNi-99].

I.4.1.1. Compromis entre énergie, latence et précision des informations

Généralement, Une application qui nous donne des informations précises en un temps court avec une faible consommation d'énergie est souhaitable dans tous les domaines.

Cependant, la précision de l'information exige la récolte et l'analyse de plusieurs données, ce qui engendre une grande latence et une consommation d'énergie supplémentaire. De ce fait, une approche ne peut généralement pas optimiser ses performances dans tous les aspects. Au lieu de cela, et en se basant sur l'importance relative de ses besoins, une application généralement donne moins d'importance pour l'optimisation des performances en respectant l'attribut le plus important. Par exemple, pour des applications de missions critiques, la latence de bout en bout est sûrement l'attribut le plus important et devrait donc toujours être gardé en dessous d'un certain seuil, même si cela demande une consommation d'énergie supplémentaire [Cas-08].

I.4.2. Agrégation (fusion) des données

Il a été montré dans plusieurs publications scientifiques que la transmission d'un bit est équivalente, en terme d'énergie, à l'exécution d'environ 1000 instructions. Cette valeur augmente avec la portée de la radio. Plus le capteur devra transmettre loin, et par conséquent augmenter sa puissance d'émission, plus il va consommer de l'énergie, et par conséquent réduire sa durée de vie. Il convient donc de réduire la quantité des données transmises en les compressant ou en les agrégeant avant de les expédier vers le puits.

En effet, dans la majorité des WSNs, les capteurs sont déployés dans une région afin d'extraire des données environnementales. Une fois les données collectée par multiples sources (plusieurs capteurs à proximité de l'événement capté), elles seront peut être transmises via multiples sauts vers une seule destination (sink). Cela, couplé avec le fait que l'information collectée par des capteurs voisins est souvent redondante et corrélée, et que l'énergie est la ressource la plus précieuse, nécessite l'emploi de la fusion des données. Au lieu de transmettre toutes les données vers un nœud central pour leur traitement, la donnée est traitée localement et seule l'information agrégée est renvoyée vers le puits. La fusion des données réduit le nombre de paquets à transmettre via les capteurs, et donc la consommation d'énergie et de bande passante. Ses avantages deviennent évidents, spécialement dans un réseau à grande échelle.

La fusion des données peut être divisée en deux catégories comme suit :

I.4.2.1. Les opérations d'agrégation : Dans la plupart des scénarios d'applications, la façon typique d'extraire les informations d'un WSN est de disséminer une requête d'agrégation déclarative dans le réseau, en demandant aux nœuds de surveiller périodiquement l'environnement, et de retourner le résultat agrégé vers le puits d'une façon périodique [Tri-06]. Cela se fait en utilisant les 5 opérations SQL (Structured Query Language) de base pour la fusion des données : Count, Min, Max, Sum et Average [Mad-02]. De plus, la majorité des systèmes de bases de données permettent de définir des fonctions propres à l'utilisateur (UDF) qui peuvent spécifier des agrégations plus complexes.

Exemple : calculer la température moyenne (average) dans une certaine région du champ de capture chaque 10 min. Un exemple d'une telle requête est : « *select avg (temperature) from Sensors where location in Region every 10 min* » [Tri-06].

I.4.2.2. Élimination de la redondance : Les lectures collectées par un même capteur à différents instants ou à travers plusieurs capteurs voisins peuvent avoir de grande corrélation entre elles, et contiennent des informations redondantes. Au lieu de transmettre toute l'information corrélée vers le demandeur, il sera plus efficace pour certains nœuds intermédiaires d'abréger l'information reçue pour en résulter des informations agrégées, dans le but de réduire la quantité des données à transmettre (et donc l'énergie consommée et la bande passante utilisée pendant la transmission).

Comme exemple de capteurs corrélés, on a les capteurs de température et d'humidité dans une même zone géographique, ou des capteurs de surveillance de mouvement des véhicules. Un autre exemple intéressant de données sensorielles corrélées sont les capteurs audio (microphones) qui captent un événement commun comme un concert ou les cris de baleines [Cho-03]. La corrélation existe sur la donnée sensorielle sous deux formes :

- **Corrélation temporelle :** Le trafic des données fait preuve de fortes corrélations temporelles. En effet, un nœud capteur tant à trouver dans sa mémoire un ancien jour avec un trafic très similaire et il est capable de rapprocher les lectures de ce jour comme une fonction linéaire aux lectures de l'ancien jour. Cela lui permet de propager seulement peu de paramètres de régression vers le puits, plutôt que les séries en temps complètes. Dans la figure I.5, le nœud capteur C détecte que la série en temps du jour courant (Mardi) est fortement corrélée avec la série en temps du Lundi dont le puits

connaît déjà (figure I.5). Il expédie seulement un ensemble de paramètres de régression de façon à ce que le puits puisse dériver la série en temps courante en se basant sur les données du Lundi, avec des bornes maximales d'erreurs définies par l'utilisateur [bac-08].

Cela démontre la force de la corrélation temporelle dans le trafic des données, et révèle une opportunité unique pour l'exploitation de cette corrélation afin d'achever des économies en communications [Sko-06].

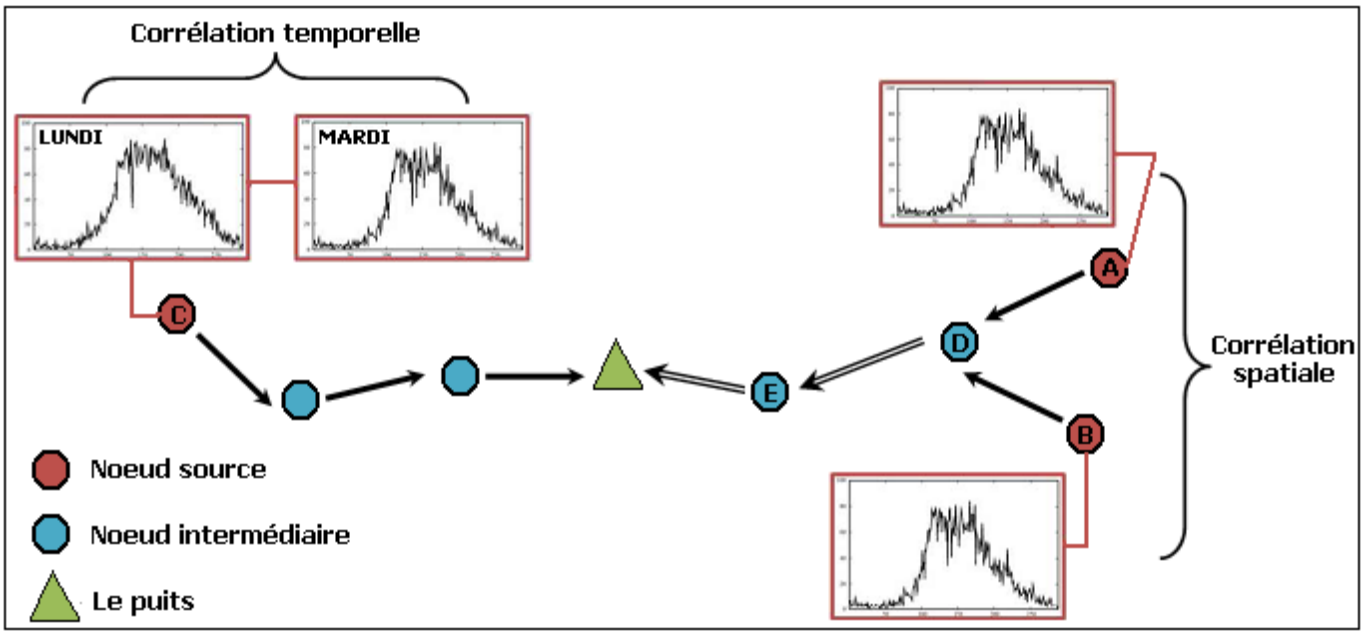

Figure I.5 : Corrélation temporelle et spatiale dans un RCSF [Bac-08].

- **Corrélation spatiale :** La force de la corrélation spatiale dépend directement de la distance séparant les différents capteurs [Sko-06]. De ce fait, des économies de communication similaires peuvent être achevées en exploitant les corrélations spatiales ; les nœuds capteurs A et B dans la figure I.5 envoient leur données en séries en temps vers le nœud D, le prochain saut dans le chemin vers le puits. Si une forte corrélation entre les deux signaux existe, le nœud D va exprimer la série en temps du nœud A comme une fonction linéaire de la série en temps du nœud B ; il expédie ensuite seulement les séries en temps des données du nœud B et les paramètres de régression pour le nœud A vers le puits via le prochain saut E. [Bac-08].

I.4.3. Approches de dissémination des données

Selon la façon de récolter les données et de les traiter, nous pouvons distinguer deux approches de dissémination des données dans les RCSFs :

- **Dissémination basée client/serveur :** où le puits envoie des requêtes aux nœuds capteurs, ensuite chaque nœud capteur va traiter la requête du puits pour lui renvoyer individuellement les données désirées qui seront ensuite traitées et agrégées au niveau du puits.

- **Dissémination basée agent mobile :** où le puits expédie un ou plusieurs agents mobiles vers les nœuds capteurs. Cet agent va transporter le code de traitement des données. De cette façon, les données seront traitées et agrégées localement au niveau des nœuds capteurs, ensuite l'agent va récolter ces données déjà traitées pour les faire parvenir au puits.

I.5. Conclusion

En raison des diverses propriétés des RCSFs (contraintes physiques, topologie de déploiement, paradigmes de communication…), ce n'est pas évident de concevoir une solution qui répond à toutes les exigences de ce type de réseaux.

Pour cela, plusieurs protocoles ont été proposés pour une dissémination efficace des données dans les WSNs. Nous avons classifié ces protocoles en deux grandes catégories : ceux basés sur le paradigme classique client/serveur et ceux utilisant des agents mobiles.

Nous aborderons en détails ces deux paradigmes dans les deux prochains chapitres, en les illustrant avec quelques solutions proposées dans la littérature.

Chapitre II

Dissémination des données basée Client / Serveur

Chapitre II. Dissémination des données basée Client/Serveur

II.1. Introduction

Les WSNs forment un environnement distribué typique et le paradigme client/serveur est un des modèles les plus adoptés dans les environnements distribués [Fug-98]. Dans ce paradigme, un serveur offre un ensemble de services, ressources, et des programmes pour l'exécution des services. Le client demande l'exécution d'un certain service. Comme réponse, le serveur traite la requête du service en exécutant le programme correspondant et en accédant aux ressources impliquées dans le serveur.

II.2. Définition du paradigme client/serveur

Dans le paradigme client/serveur, une application est divisée en deux processus, un processus client s'exécutant localement qui demande des services et un processus serveur sur un site distant qui offre des services au client. Les processus client et serveur doivent communiquer entre eux dans le but d'exécuter leur tache avec succès. La communication est faite par le biais d'échange de messages. Lorsqu'un client souhaite la réalisation d'un service précis, il envoie une requête au serveur qui exécute le service et lui renvoie le résultat.

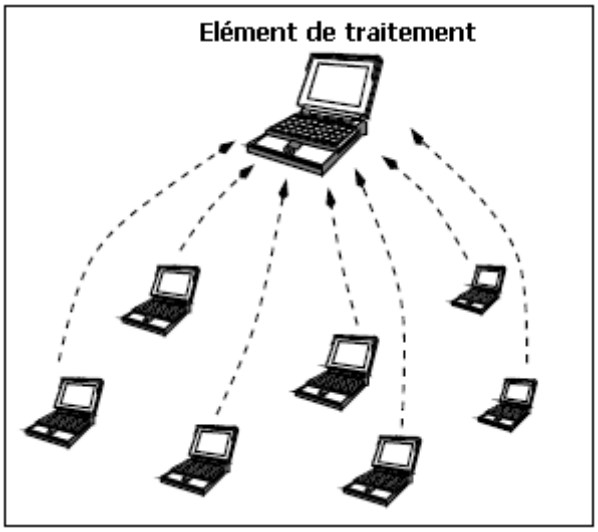

Figure II.1 : Le paradigme Client/Serveur.

II.3. Avantages et inconvénients du paradigme client/serveur

Le modèle client/serveur est le plus populaire dans les systèmes distribués, pour les raisons suivantes :

- Toutes les données sont centralisées sur un seul serveur, ce qui simplifie les contrôles de sécurité et la mise à jour des données et des logiciels ;
- Compromis entre la facilité de déploiement et la maîtrise du savoir faire lié à la fonctionnalité propre du service ;
- Possibilité pour le client de sous-traiter des tâches sur des machines plus performantes ;
- Étant donné que le serveur est au centre du réseau, il peut gérer des ressources communes à tous les utilisateurs, comme par exemple une base de données centralisée, afin d'éviter les problèmes de redondance et de contradiction ;
- Une meilleure sécurité, car le nombre de points d'entrée permettant l'accès aux données est moins important.

Quoiqu'il soit très largement utilisé, le modèle client/serveur a plusieurs inconvénients qui peuvent dégrader les performances du système :

- Si trop de clients veulent communiquer avec le serveur au même moment, ce dernier risque de ne pas supporter la charge ;
- Si le serveur n'est plus disponible, plus aucun des clients ne marche, étant donné que tout le réseau est architecturé autour de lui ;
- Les coûts sont élevés dû à la technicité du serveur ;
- Il requiert une grande bande passante dû au grand nombre de messages à échanger ;
- Le client doit garder la connexion active lorsque la requête du client est exécutée sur le serveur. Si la connexion échoue (ce qui arrive fréquemment avec les

connexions sans fil non fiable), le client doit envoyer une nouvelle fois sa requête au serveur, qui va la traiter du début.

II.4. Le paradigme client/serveur dans les WSNs

Dans le traitement collaboratif, le modèle informatique le plus utilisé est le client/serveur, où les capteurs individuels (clients) envoient la donnée brute ou prétraitée vers un centre de traitement (serveur) et l'intégration de données est exécutée au centre (généralement au niveau du puits ou des super nœuds). Comme cité ci-dessus, il existe quelques inconvénients avec ce modèle qui devraient être pris en considération notamment pour les WSNs. En effet, malgré la non fiabilité et la bande passante limitée des liaisons sans fil utilisées dans les réseaux de capteurs, en plus de la contrainte d'énergie qui est cruciale dans les WSNs, les protocoles utilisant le paradigme client/serveur dans ce type de réseau sont nombreux. Par exemple, le protocole DD [Int-00] est l'un des modèles les plus utilisés dans beaucoup de travaux sur les réseaux de capteurs.

II.5. DD [Int-00]

DD (*Directed Diffusion*) [Int-00] est un protocole de dissémination de données, permettant d'utiliser plusieurs chemins pour le routage d'information. Le puits diffuse un intérêt sous forme de requête, afin d'interroger le réseau sur une donnée particulière, puis décide du meilleur chemin à utiliser pour la réception des données.

DD repose sur quatre éléments : désignation des demandes, propagation des intérêts et établissement des gradients, propagation des données et enfin renforcement des chemins.

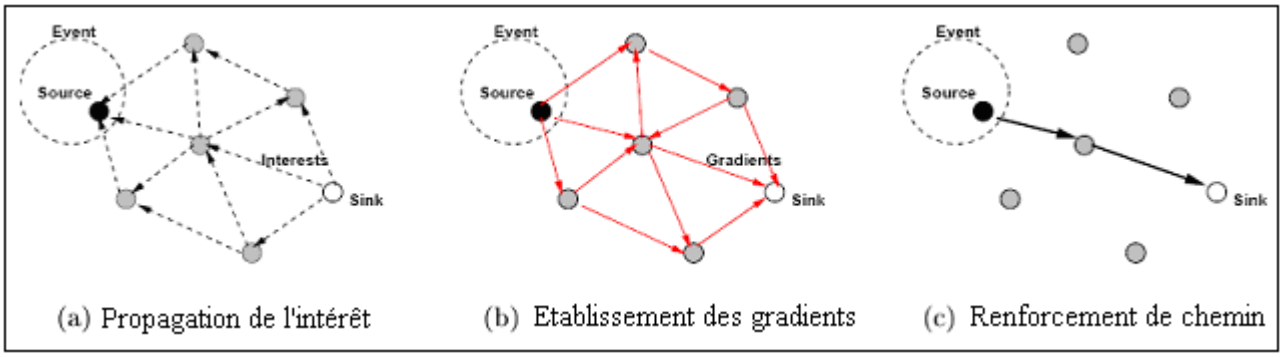

Figure II.2 : Principe de fonctionnement de DD.

II.5.1. Désignation des demandes

Dans DD, la description d'une demande est désignée par une liste de paires *attribut-valeur* qui décrivent un intérêt. Par exemple, une tâche de détection d'un animal à 4 pattes pourrait être décrite comme suit :

```
type = animal à quatre pattes      // détecter la présence d'un animal
intervalle = 20 ms                 // envoyer les événements chaque 20 ms
durée = 10 secondes                // pour les 10 prochaines secondes
rect = [-100, 100, 200, 400]       // pour les capteurs dans la région spécifiée
```

Les données envoyées en réponse aux intérêts sont également structurées en utilisant le même schéma. Ainsi, par exemple, un capteur qui détecte un animal pourrait produire les données illustrées dans la figure suivante :

```
type = animal à quarte pattes      // type de l'animal capté
emplacement = [125, 220]           // emplacement du nœud
intensité = 0.6                    // mesure de l'amplitude du signal
confidence = 0.85                  // taux de sûreté de l'événement
timestamp = 01:20:40               // l'instant de génération de l'événement
```

II.5.2. Propagation de l'intérêt et établissement des gradients

Lorsqu'un puits requiert une donnée du réseau, il diffuse un intérêt, contenant sa description ainsi que le débit d'information désiré, i.e. : l'intervalle entre deux émissions d'événements. Initialement, le puits spécifie un grand intervalle, dans un but d'exploration. Cela permet d'établir les gradients et de découvrir d'éventuelles sources, sans pour autant encombrer le réseau. Par la suite, le puits augmentera le débit en présence des données voulues (renforcement de chemins).

Afin de propager l'intérêt, DD emploie l'inondation globale dans le réseau. Chaque nœud maintient localement un cache d'intérêts contenant les informations suivantes :

– La description de l'intérêt, en utilisant le schéma de désignation.

– Un ensemble de gradients.

Un gradient est un vecteur représentant l'intérêt. Il est caractérisé par deux éléments :

– Une direction, modélisée par le voisin émetteur de l'intérêt.

– Une amplitude, représentée par le débit de données désiré par le voisin concerné.

Lorsqu'un nœud reçoit un intérêt, il parcourt son cache :

– Si le cache ne contient aucune entrée relative à l'intérêt reçu, une nouvelle entrée est créée avec un gradient vers le voisin émetteur.

– Dans le cas contraire, le nœud recherche un gradient vers le voisin émetteur, et met à jour en conséquence l'entrée en question.

Après le traitement du cache, le nœud relaie l'intérêt vers ses voisins. La méthode la plus utilisée est l'inondation, i.e. : envoyer le message à tous les voisins directs.

Nous remarquons que la propagation d'intérêt élimine la notion de puits. Le cache d'intérêt ne conserve aucune information sur le puits initiateur. Cela permet de déployer un réseau avec plusieurs puits ayant les mêmes intérêts. En plus, les données peuvent être routées vers ces puits en utilisant des portions de routes communes, économisant ainsi l'énergie des nœuds.

II.5.3. Propagation des données

Lorsqu'un nœud détecte un événement, il le compare aux différents intérêts stockés dans son cache. Lorsqu'il trouve une correspondance, le nœud choisit le débit le plus élevé et prélève les données en conséquence. En consultant les gradients relatifs à l'intérêt, le nœud détermine les prochains sauts vers les puits (chacun avec son propre débit).

Lorsqu'un nœud reçoit une donnée, il recherche un intérêt équivalent dans son cache. Si aucune entrée n'est trouvée, le paquet est ignoré. Dans le cas contraire, le nœud utilise son cache de données pour traiter l'information.

Ce cache enregistre les données récemment émises par les voisins. Cela évite l'émission d'informations dupliquées et la création de boucles, en supprimant les données déjà rencontrées. En plus, il permet de garder trace des chemins vers les sources, car la phase d'établissement des gradients ne crée que les routes vers les puits. En consultant la liste des gradients, le nœud relaie la donnée vers ses voisins, suivant le débit de chacun d'eux.

II.5.4. Renforcement des chemins et maintenance des routes

Lorsque le puits reçoit les premières données exploratoires, il renforce un chemin vers l'un de ses voisins émetteurs, en lui réclamant un plus grand débit de capture. Le choix du voisin à renforcer se fait suivant un de ces critères :

- Le voisin renforcé est celui qui a rapporté la donnée en premier,
- Le voisin renforcé est celui qui a rapporté une donnée correspondant à un nouvel évènement dans le réseau,
- Le voisin renforcé est celui qui a plus d'énergie que les autres.

Généralement, c'est le premier critère qui est utilisé.

Á ce stade, la phase d'exploration est clôturée, et la collecte des données va donc être entamée.

Le renforcement ne doit pas s'arrêter au niveau des voisins du puits, mais doit se propager éventuellement jusqu'aux sources. Pour ce faire, lorsqu'un nœud reçoit un message de renforcement, il consulte son cache d'intérêt. Si le débit spécifié dans le message est plus grand que tous les autres débits des gradients présents, le nœud doit renforcer un de ses voisins. Le voisin est choisi en utilisant le cache de données. Ce renforcement est appelé renforcement positif.

En plus du renforcement de chemin initié par le puits, les nœuds intermédiaires dans un chemin renforcé peuvent aussi appliquer les règles du renforcement si nécessaire. Un tel renforcement par les nœuds intermédiaires est très util pour permettre le recouvrement local des chemins défaillants ou dégradés causés par différents facteurs comme la baisse d'énergie d'un nœud ou la présence d'obstacles. Lorsqu'un nœud remarque qu'il y a perte de paquets provenant de son voisin renforcé ou qu'il ne peut plus le joindre ; il doit agir sur place pour renforcer un autre voisin afin d'utiliser un chemin meilleur.

Cependant, en adoptant ce mécanisme, plusieurs chemins seront renforcés engendrant les mêmes problèmes de l'inondation classique. Pour éviter ce cas, DD introduit un renforcement négatif, i.e. : le puits doit choisir des voisins afin de diminuer leur débit. Ce choix peut se faire suivant plusieurs critères :

- Si le nœud n'a pas émis de nouvel événement depuis une certaine période,

- Après expiration d'un temps (time-out) durant lequel le nœud n'a reçu aucun nouveau message de renforcement positif.

Lorsqu'un nœud reçoit un message de renforcement négatif, il doit émettre lui aussi un message de renforcement négatif vers l'un de ses voisins, en utilisant le cache de données.

II.5.5. Analyse

Le paradigme Directed Diffusion ne limite pas le concepteur à un seul usage particulier. En effet, à chaque étape, plusieurs critères de choix sont possibles ; et cela dépend de nature de l'application et de ses exigences.

Avantages

- Consomme moins d'énergie que l'inondation grâce au renforcement d'un seul chemin.
- Permet une bonne latence grâce aux choix du meilleur chemin à renforcer.
- Permet un bon passage à l'échelle grâce aux interactions locales entre voisins (un-à-un) et donc pas besoin de déterminer le chemin complet entre le puits et la source.
- La retransmission périodique de l'intérêt et des données permet une bonne robustesse notamment lors de la mobilité ou de défaillance des nœuds.
- Permet l'agrégation au niveau de chaque nœud. En effet, plusieurs demandes pour le même intérêt initiées par multiples puits, ne seront diffusées qu'une seule fois. De plus, le cache des données permet d'éliminer les redondances et les boucles.
- Facilité de maintenance des routes grâce au multipath.

Inconvénients

- Le traitement du cache consomme une énergie assez importante et occupe beaucoup de mémoire.
- Grand coûts lors de l'établissement des gradients à cause de l'inondation.
- Durée de vie du réseau diminuée, à cause de la diffusion périodique de l'intérêt et des données.
- Épuisement rapide de l'énergie des nœuds du chemin renforcé.

II.5.6. Variantes de DD

Le protocole DD inclue une famille d'algorithmes dont : Two-phase pull diffusion, Push diffusion et One-phase pull diffusion.

Two-phase pull diffusion [Int-00] correspond au premier algorithme de DD décrit par les quatre éléments déjà présentés.

La diffusion initiale de l'intérêt plus la diffusion des messages des données exploratoires, constituent la première phase de two-phase pull diffusion. La deuxième phase comprend le renforcement de chemins et la transmission des données via les chemins renforcées.

Two-phase pull diffusion est souhaitable pour les scénarios avec plusieurs sources et peu de puits [Che-04].

Push Diffusion [Hei-03] est similaire à two-phase pull diffusion, seulement ici les rôles des nœuds sources et des puits sont inversés. Les puits deviennent passifs, en gardant les informations de l'intérêt localement. Par contre, les nœuds sources deviennent actifs ; les données exploratoires sont renvoyées à travers le réseau sans création préalable d'intérêts et de gradients. Lorsque les données exploratoires arrivent au puits, un message de renforcement est généré et est récursivement renvoyé vers la source en créant un gradient renforcé, et les données non exploratoires suivent seulement ces gradients renforcés.

Push diffusion n'est pas très convenable pour des applications avec plusieurs sources générant continuellement des données puisque chaque donnée devrait être renvoyée à travers tout le réseau même si elle ne sera pas utilisée. Un bénéfice de Push diffusion comparée à two-phase pull est qu'elle a seulement un cas où l'information est diffusée a travers le réseau (donnée exploratoire) plutôt que deux (intérêts et données exploratoires).

Push diffusion est préférable pour les applications avec plusieurs puits et peu de sources, les applications à plusieurs puits et sources, où les sources produisent des données occasionnellement et les applications de missions critiques comme le suivi de cible [Zha-02].

One-Phase Pull Diffusion [Hei-03] est un sous système qui évite une des deux phases de diffusion présentées dans two-phase pull diffusion. Ici, Le puits envoie les messages d'intérêts qui seront diffusés à travers le réseau en établissant les gradients. Lorsqu'un intérêt

atteint une source, celle-ci ne va pas marquer ses premiers messages de données comme exploratoires, mais envoie la donnée complète directement via le gradient préféré. Le gradient préféré est déterminé par le premier voisin qui a envoyé l'intérêt correspondant, cela signifie le chemin avec la plus petite latence. One-phase pull diffusion ne requiert pas des messages de renforcement, et le chemin avec la plus petite latence est implicitement renforcé.

One-phase pull diffusion, est aussi préférable pour des scénarios avec plusieurs sources et peu de puits.

II.6. TTDD [FYe-02]

Le protocole TTDD (*Two-Tier Data Dissemination*) [FYe-02] a été conçu afin de répondre aux besoins d'applications employant des puits mobiles.

TTDD est un protocole hiérarchique dynamique, choisissant les chefs suivant leurs positions géographiques en employant une structure appelée grille. Une grille est un ensemble de carrés, nommés cellules, couvrant tout le réseau. Les sommets des cellules représentent des points de dissémination. Le nœud le plus proche d'un point de dissémination est nommé nœud de dissémination, qui représente un chef dans la hiérarchie.

Le protocole TTDD est dépendant de la source, ce qui permet d'établir un partage de charge entre les nœuds. En effet, le rôle de chef (nœud de dissémination) surcharge le nœud avec plus de responsabilité, consommant plus d'énergie. Avec une telle approche, cette responsabilité est partagée par plus de nœuds, car chaque nœud construit sa propre hiérarchie.

TTDD suppose que lorsqu'un évènement surgit dans une zone, un seul capteur est élu pour le représenter. Cependant, le protocole ne spécifie pas la manière d'élection de ce représentant. Le protocole suppose aussi que chaque capteur connaît sa position géographique excepté le puits.

II.6.1. Construction de la grille

Le nœud source L_s, ayant les coordonnées (x, y), commence par la construction de sa grille. La grille est constituée d'un ensemble de points $L_p(x_i, y_j)$, appelés points de dissémination, définis comme suit : $\{(x_i, y_j) / x_i = x + i\alpha, y_j = y + {}_j\alpha; i, j = 0, \pm1, \pm2, ...\}$.

Où α est un paramètre du protocole déterminant la taille des cellules.

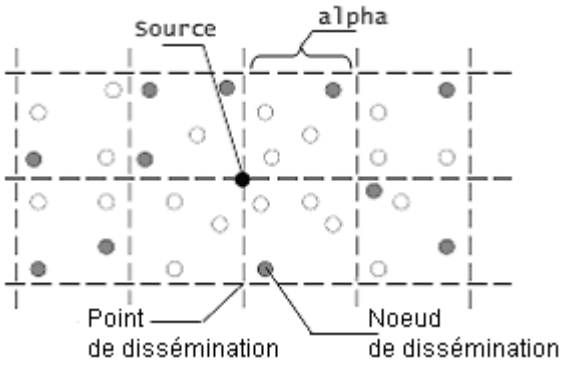

Figure II.3 : Structure de la grille dans TTDD.

La source représente le premier nœud de dissémination. Il émet par la suite un message REQ aux quatre prochains points de dissémination (i.e. (x+α, y+α), (x+α, y−α), (x−α, y+α) et (x−α, y−α)), en utilisant un relais géographique glouton définit comme suit :

– Pour chaque point de dissémination L_p, la source choisit le voisin le plus proche de L_p, et lui émet le message REQ.

– Lorsqu'un nœud reçoit ce message, il vérifie s'il est le plus proche de L_p parmi tous ses voisins :

- Si ce n'est pas le cas, il relaie le message vers le voisin le plus proche.
- Dans le cas contraire, si la distance entre le nœud et L_p est inférieure à (α/2), le nœud devient un nœud de dissémination et réitère le processus, sinon le message est supprimé.

Un nœud de dissémination sauvegarde les coordonnées du point L_p qu'il représente ainsi que l'identificateur du nœud qui a émis le message REQ (i.e. *upstream* vers la source).

II.6.2. Relais de la requête et des données

Quand un puits désire une donnée, il inonde la région locale (d'un diamètre d'environ α) afin d'atteindre le nœud de dissémination le plus proche, qui représentera son nœud de dissémination immédiat NDI.

Le NDI relaie la requête en utilisant l'information du upstream dans chaque nœud, afin d'atteindre la source. En même temps, le chemin inverse est construit, en sauvegardant l'identificateur du *downstream* afin d'acheminer les données vers le puits.

Une agrégation peut avoir lieu lorsque plusieurs puits désirent la même donnée. En effet, un nœud de dissémination qui reçoit la même requête de la part de plusieurs puits, va envoyer seulement une demande vers la source afin de réduire les coûts de communication.

Nous remarquons que le chemin emprunté par TTDD entre deux nœuds n'est pas le chemin optimal (en considérant que le chemin optimal est une droite directe entre les 2 nœuds). En effet, la longueur du chemin emprunté peut atteindre au maximum √2 fois la longueur d'une ligne droite. Cependant, TTDD estime que cette sub-optimalité vaut la peine en gain dans la scalabilité.

II.6.3. Relais des données et traitement de la mobilité du puits

Le but de cette étape est de faire parvenir la donnée au puits, même si ce dernier se déplace.

Chaque puits est associé à un agent primaire AP et un agent immédiat AI. Initialement, le puits choisit un voisin comme AP et AI, et transmet les coordonnées de l'AP dans ses requêtes.

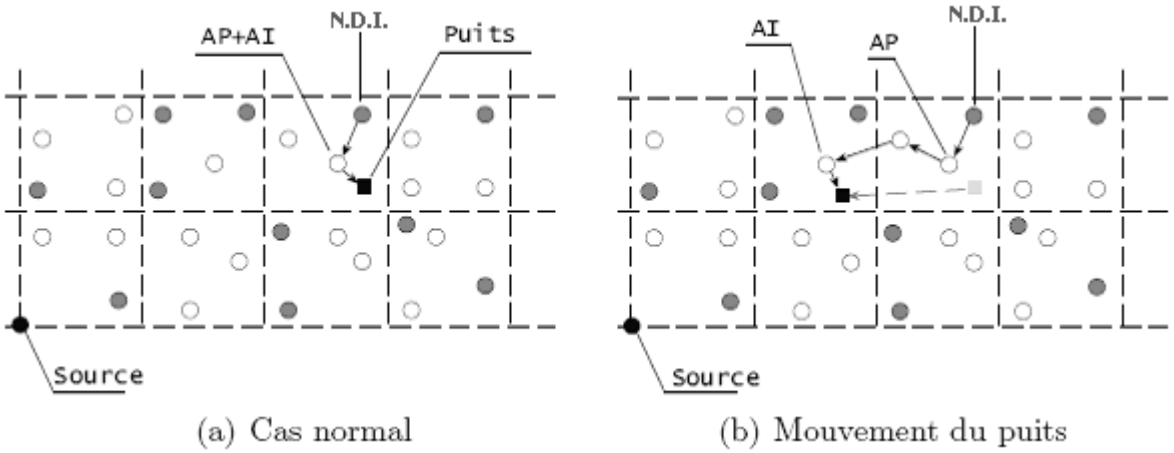

Figure II.4 : Relais des données dans TTDD.

Lorsque la donnée arrive au NDI, il la transmet à l'agent primaire, en utilisant un relais géographique glouton. Lorsque le puits se déplace hors portée de son AI, il doit choisir un

nouvel AI parmi ses voisins, et transmet ses coordonnées à l'agent primaire et à l'ancien AI, en utilisant toujours un relais glouton.

L'AP est le représentant du puits au niveau du NDI ; et l'AI est le représentant du puits au niveau de l'agent primaire.

Cette méthode permet d'implémenter un mécanisme de handoff afin d'éliminer les pertes de données lors du déplacement du puits.

Lorsque le puits s'aperçoit que la distance vers l'AP dépasse un certain seuil, il choisit un nouvel agent primaire, et éventuellement un nouveau NDI.

II.6.4. Maintenance de la grille

Afin de conserver la structure de la grille, TTDD évite de rafraîchir périodiquement la structure et engendrer donc une perte d'énergie considérable. Toutefois, le protocole préfère un mécanisme de duplication d'information. Pour cela, chaque nœud de dissémination, désigne plusieurs nœuds parmi ses voisins afin d'y répliquer les positions de son nœud de dissémination upstream. Lorsqu'un nœud de dissémination tombe en panne, les requêtes venant vers lui vont s'arrêter au niveau d'un de ses voisins. Celui là va donc utiliser l'information dupliquée sur la position du upstream afin de lui transmettre le message. Après, quand la donnée arrive retardée, un nouveau nœud de dissémination est élu en suivant le même mécanisme lors de la construction de la grille.

II.6.5. Analyse : Généralement, les protocoles hiérarchiques tels que TTDD [FYe-02] permettent une bonne gestion du réseau, seulement la construction et la maintenance de la structure hiérarchique est souvent coûteuse notamment pour les WSNs qui sont limités en ressources.

Avantages

- La structure en grille facilite le passage à l'échelle.
- Préservation de l'énergie lors de l'inondation locale de l'intérêt.
- Tolérance aux pannes grâce aux informations dupliquées chez les voisins des NDI.
- Mécanisme de handoff permet d'éviter la perte de données grâce à l'AP et l'AI.
- Agrégation des requêtes et des données lors de l'existence de multiples puits.
- Partage de la charge entre les nœuds grâce à l'utilisation d'une grille différente pour chaque nœud source.

Inconvénients

- Une grille pour chaque source est très coûteuse en construction et en maintenance.
- La grille est basée cible implique la création de nouvelle grille à chaque mouvement de la cible.

II.7. LBDD [Ben-07]

LBDD (*Line-Based Data Dissemination protocol*) [Ben-07] est un autre protocole qui prend en charge la mobilité du puits, et mettant en œuvre une structure virtuelle facilitant le processus de dissémination.

II.7.1. La structure virtuelle

LBDD définit alors une structure virtuelle sous la forme d'une bande, de largeur w, placée au milieu de la surface d'intérêt. Cette bande est également divisée en groupes de taille g (voir figure II.5). Les deux paramètres w et g permettent de limiter la congestion et d'assurer le passage à l'échelle. Les nœuds positionnés à l'intérieur de la bande sont appelés *inline-nodes*. Cette bande représente une zone de rendez-vous pour les requêtes et le stockage des données.

Ce protocole suppose que chaque nœud connaît sa position géographique ainsi que les coordonnées de la zone d'intérêt. Grâce à cette supposition, l'élection des inline-nodes se fait donc facilement. En ce qui concerne l'acheminement des données, l'auteur utilise un routage géographique.

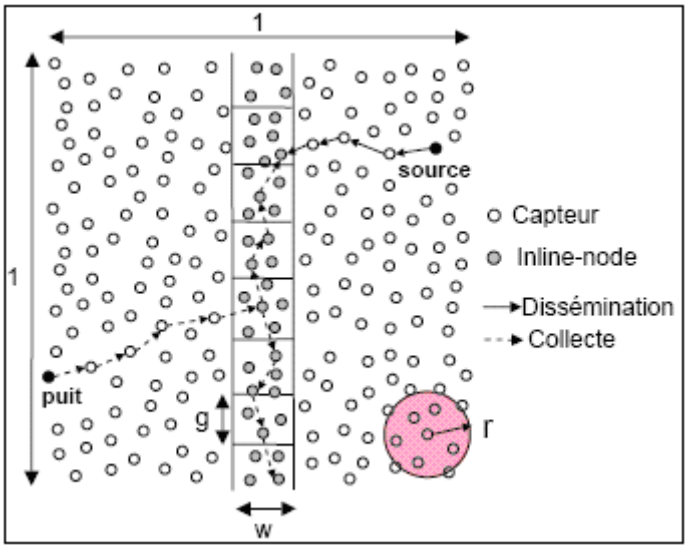

Figure II.5 : Principe de fonctionnement du protocole LBDD.

II.7.2. Les étapes de fonctionnement

Le fonctionnement de LBDD se décompose en deux principales étapes :

- *Dissémination :* dès qu'un nœud détecte un stimulus, une donnée est générée et est envoyée vers le plus proche *inline-node* ou elle sera inondée au sein du groupe ;
- *Collecte :* afin de collecter les différentes données, le nœud puits envoie une requête perpendiculairement à la bande. Le premier *inline-node* qui reçoit la requête la fait propager dans les deux directions de la bande jusqu'à atteindre le ou les nœuds possédant la donnée recherchée. Celle-ci est alors renvoyée directement au puits.

II.7.3. Mobilité du puits

Pour gérer la mobilité du puits, celui-ci élit parmi ses voisins un *home-node* qui va se charger d'émettre les requêtes vers la bande et de retransmettre les données reçues vers le nœud puits. Cette élection est périodique et dépend de la vitesse du puits. Enfin, si le puits s'éloigne de son home-node, un chemin est alors construit grâce au *progressive-footprint-chaining* [Shi-05].

II.7.4. Analyse

Les approches de dissémination basées sur un point de rendez-vous telles que LBDD [Ben-07] et Railroad [Shi-05] paraissent très simples comme solution pour les réseaux de capteurs sans fil, mais leur principale difficulté est la construction de la structure virtuelle.

Avantages

- Améliore la latence en parcourant seulement la moitié du chemin pour atteindre la donnée.
- Mécanisme de handoff permet d'éviter la perte de données lors du déplacement du puits.
- Préserve l'énergie en dirigeant les requêtes et les données directement vers la bande (pas de besoin de diffuser).

Inconvénients

- Concentration de la charge au niveau de la bande engendre l'épuisement en énergie et la saturation des mémoires des nœuds dans la bande.

- Absence de méthode de détermination des paramètres de construction de la structure virtuelle (w et g) notamment lors du passage à l'échelle.
- Absence de mécanisme d'agrégation des données au niveau de la bande.

II.8. Conclusion

Grâce à sa simplicité et sa facilité de déploiement, le paradigme client/serveur est le plus utilisé dans les environnements distribués notamment dans les WSNs. Cependant, cette solution a été critiquée pour être coûteuse en bande passante, en énergie et parfois même en latence. Pour cela, il y eu l'apparition d'un nouveau paradigme dit *Agent Mobile*, qu'on étudiera dans le prochain chapitre.

Chapitre III

Dissémination des données basée Agents Mobiles

Chapitre III. Dissémination des données basée Agents Mobiles

III.1. Introduction

La programmation par agents mobiles est un paradigme de programmation des applications réparties, susceptible de compléter ou de se substituer à d'autres paradigmes plus classiques tel que le passage de messages et l'appel de procédure à distance. Elle est d'un grand intérêt pour la mise en œuvre d'applications dont les performances varient en fonction de la disponibilité et de la qualité des services et des ressources, ainsi que du volume des données déplacées. Le concept d'agent mobile facilite en effet la mise en œuvre d'applications dynamiquement adaptables, et il offre un cadre générique pour le développement des applications réparties sur des réseaux de grande taille qui recouvrent des domaines multiples.

Récemment, les agents mobiles ont été proposés pour une dissémination efficace des données dans les réseaux de capteurs. Dans les architectures traditionnelles client/serveur, les données de différentes sources sont transférées à une destination ; alors que dans les paradigmes avec agents mobiles, un code exécutable spécifique à une tâche traverse les sources pertinentes pour rassembler les données.

III.2. Définition d'un agent mobile

Un AM consiste en un code d'exécution et son état (les valeurs des variables, la prochaine instruction…etc.). Initialement un AM réside dans la machine qui l'a créé. Il est ensuite expédié pour s'exécuter sur une machine distante appelée généralement hôte d'AM ou serveur. Lorsqu'un AM est expédié, le code entier de l'AM et son état d'exécution sont transférés vers l'hôte.

L'hôte fournit à l'AM un environnement d'exécution convenable. L'AM va utiliser les ressources (CPU, mémoire…etc.) de l'hôte pour exécuter sa tâche. Après avoir accompli sa tâche au niveau de l'hôte, l'AM va migrer vers une autre machine. Puisque l'état d'exécution est aussi transféré vers l'hôte, l'AM peut reprendre son exécution au point où elle s'est arrêtée au niveau de l'ancien hôte. De cette manière, l'AM va continuer son parcours jusqu'à la dernière machine de son itinéraire pour enfin revenir vers la machine qui l'a créé.

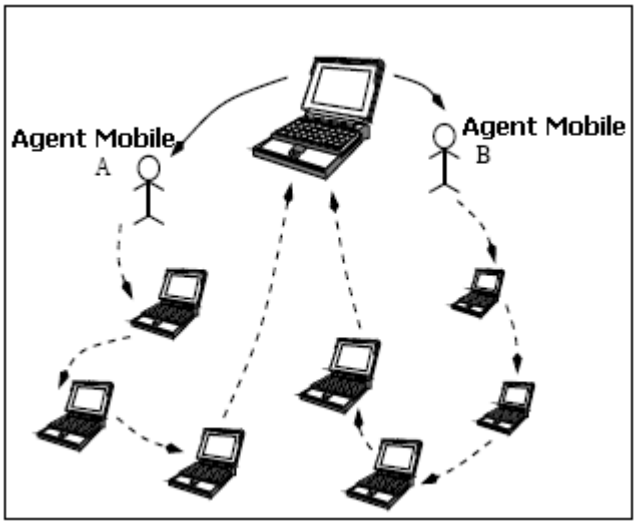

Figure III.1 : Paradigme Agent Mobile.

III.3. Avantages et inconvénients des agents mobiles

Les agents mobiles sont un paradigme pour l'informatique distribuée, particulièrement convenable pour supporter le développement d'applications distribuées, des services et des protocoles dans les systèmes distribuées conventionnels de même que dans les environnements distribués très dynamiques [Yon-05].

Dans [Lan-99], sept bonnes raisons sont exposées pour utiliser les agents mobiles ; nous les éclaircissant en proposant des exemples de contextes dans les WSNs [Aie-08] :

III.3.1. Réduction de la charge du réseau

Les agents mobiles peuvent accéder à des ressources éloignées (ex : BD, capteurs) ou communiquer avec d'autres agents éloignés ou avec d'autres composants, en se déplaçant vers leur site et en interagissant avec eux localement, et économiser ainsi les ressources du réseau comme la bande passante et l'énergie durant la phase d'interaction. Comme exemple dans un WSN, un nœud capteur transmet périodiquement la données captées vers le puits, qui à son tour traite ces données afin d'extraire les informations agrégées. La transmission des données captées consomme de la bande passante durant son acheminement du nœud capteur vers le puits. Un AM incorporant l'algorithme de traitement des données peut migrer vers le nœud capteur et appliquer localement cet algorithme en réduisant ainsi la consommation de bande passante. En effet, l'AM doit aussi transmettre la donnée agrégée vers le puits, mais dans ce cas, la bande passante consommée est beaucoup plus petite que celle consommée

pour la transmission de la donnée brute. Cela implique aussi une réduction de la consommation d'énergie durant la communication, vu que la taille des données transportées par l'AM est plus petite que celles des données brutes.

III.3.2. Surmonter la latence du réseau

Les agents mobiles peuvent se déplacer vers des nœuds éloignés et appliquer localement les contrôles en temps réel pour la régulation physique/logique des dispositifs/composants en évitant ainsi le contrôle à distance qui est lourdement affecté par la latence du réseau. Comme exemple de scénario ; un capteur/actionneur éloigné qui doit être calibré pour l'exécution d'une tâche donnée, et après contrôlé en temps réel pour compléter cette tâche avec succès. Un AM équipé avec le code de calibration et de contrôle peut se déplacer vers le nœud capteur/actionneur et le calibrer localement. Dans ce sens, la latence du réseau n'affectera pas l'opération de control en temps réel qui peut être exécutée aussi dans le cas de panne de la connectivité du réseau avec la station de base.

III.3.3. Encapsulation de protocoles

Les AMs peuvent encapsuler des protocoles et les déployer dynamiquement et surmonter ainsi la standardisation ou les enjeux de mise à niveau. Les protocoles peuvent alors évoluer dynamiquement ou être modifiés selon la demande pour être plus efficaces. Un exemple de cela dans le routage ; un protocole de routage spécifique supportant des chemins multi sauts qui devrait être déployé dans une zone donnée du WSN. Une tâche de coopération d'AMs implémentant le protocole de routage peut être créée et migrée vers les nœuds capteurs de la zone ciblée dans le WSN. Lorsqu'une nouvelle version du protocole est définie, une nouvelle tâche implémentant le nouveau protocole sera lancée pour remplacer la version antérieure du protocole.

III.3.4. Exécution asynchrone et autonome

L'asynchronisation et l'autonomie sont des propriétés distinctives des AMs. Une fois injectés dans le réseau, les AMs peuvent exécuter d'une façon autonome la tâche programmée et supporter les opérations déconnectées en opérant d'une façon asynchrone tout en respectant le processus ou le composant qui l'a généré. Ces propriétés sont très importantes dans les environnements dynamiques où la topologie du réseau change rapidement et les connections sont non stables d'une façon à ce qu'elles peuvent être établies seulement pour de courtes

périodes de temps. Un exemple de tâche asynchrone dans un WSN : périodiquement, l'utilisateur du WSN demande aux nœuds capteurs de lui envoyer leur taux d'énergie courant. Pour cela, un AM créé par l'utilisateur, peut se balader dans le réseau, nœud par nœud, d'une façon autonome pour collecter l'information désirée et finalement rapporter cette information d'une façon asynchrone vers le demandeur.

III.3.5. Adaptation dynamique

Les AMs peuvent capter leur environnement d'exécution et réagir automatiquement aux changements. D'ailleurs les tâches transportées par l'AM peuvent se distribuer à travers les nœuds du réseau pour régler un problème particulier d'une façon coopérative. L'adaptation dynamique peut supporter le comportement autonome d'un WSN ; en fait, les WSNs sont des systèmes à longue exécution dans lesquels les pannes des dispositifs sont accablées. L'adaptation concernant les pannes des dispositifs est déjà devenue un aspect important dans ce domaine. Un exemple est donné dans le point 7.

III.3.6. Orientation à l'hétérogénéité

Les AMs peuvent être utilisés comme des intégrateurs des systèmes hétérogènes. Ils peuvent agir comme des couvertures ou des médiateurs à travers les systèmes basés sur différents matériaux et logiciels. Ils peuvent aussi être convertis pour franchir le bord à travers des systèmes hétérogènes. Un scénario intéressant est : l'intégration d'un WSN avec un réseau IP. Un AM peut être envoyé vers la passerelle entre le WSN et le réseau IP pour agir comme couverture du WSN. N'importe quelle information à collecter du WSN est demandée à l'AM de la part des composants résidants dans le réseau IP. L'AM, à son tour, va traduire une telle requête en une requête spécifique et la soumettre au WSN, et une fois qu'il reçoit la réponse, il va la renvoyer au composant l'ayant demandée.

III.3.7. Robustesse et tolérance aux fautes

Comme les AMs traitent l'habilité à réagir dynamiquement aux situations non favorables et aux événements inattendus, ils peuvent supporter la construction d'un système distribué robuste et tolérant aux fautes. Un scénario fréquent de tolérance aux fautes dans les WSNs est le suivant : Un nœud capteur qui va s'éteindre dû au déchargement de ses batteries, alors l'activité de captage devrait être activée dans un nœud capteur voisin en temps réel. Tous les

AMs s'exécutant sur des nœuds capteurs non chargés vont migrer vers un autre nœud équivalent de façon à continuer leur activité après la migration.

Cependant, Le principal inconvénient des AMs est le risque en sécurité. Par exemple, un AM malicieux peut endommager un hôte particulier, ou alors un hôte malicieux peut modifier le fonctionnement de l'agent…etc.

III.4. Les agents mobiles dans les WSNs

Comme illustré dans la section précédente, l'utilisation d'AMs dans les réseaux informatiques a des avantages mais aussi des inconvénients, tels que le code de mise en cache et la sécurité dans certains scénarios. Malgré tout, ils sont utilisés avec succès dans différentes applications telles que la programmation parallèle, la collecte des données, le e-commerce et l'informatique mobile. Tel que décrit dans [Hai-03], de nombreux avantages inhérents (par exemple la scalabilité et la sensibilisation d'énergie) de l'architecture AM la rend plus adaptée aux WSNs que l'architecture client/serveur. En effet, les agents mobiles peuvent être utilisés pour réduire considérablement le coût de communication, spécialement à travers de faibles bandes passantes, en déplaçant la fonction de traitement vers la donnée plutôt qu'apporter la donnée vers un processeur central.

Les agents mobiles peuvent supporter la programmation des WSNs au niveau de différentes couches [Aie-08]:

- **Au niveau de la couche application :** des agents mobiles peuvent être utilisés pour la conception et la programmation des abstractions par lesquelles les applications du WSN peuvent être effectivement développées ;

- **Au niveau de la couche middleware :** des agents mobiles peuvent être utilisés pour l'implémentation des noyaux des services du WSN comme l'agrégation des données, la fusion et la dissémination, ainsi que le déploiement dynamique de nouveaux services via la dissémination efficace du code ;

- **Au niveau de la couche réseau :** des agents mobiles peuvent être utilisés comme les capsules dans les réseaux actifs [Lev-02] pour le routage multi sauts et d'autres services du réseau.

III.5. MAWSN [Che-06]

L'approche MAWSN (*Mobile Agent based Wireless Sensor Network*) [Che-06] propose d'utiliser le paradigme agents mobiles (AM) pour réduire et agréger les données dans un réseau de capteurs à architecture plate.

III.5.1. Principe du protocole

Dans les scénarios traditionnels, plusieurs demandes d'informations différentes arrivent à des moments différents. MAWSN juge que les applications qui nécessitent plusieurs tâches différentes s'exécutant simultanément deviendront très répandues dans l'avenir. En se basant sur le principe que le coût de communication pour l'envoi d'un message long est généralement moins que celui d'envoi de la même quantité de données en utilisant plusieurs messages courts ; le protocole MAWSN va accomplir plusieurs tâches concurrentes associées avec de petites quantités de données par un seul paquet transportant plusieurs requêtes, et concaténer après leurs résultats dans un seul paquet afin de diminuer le coût de communication.

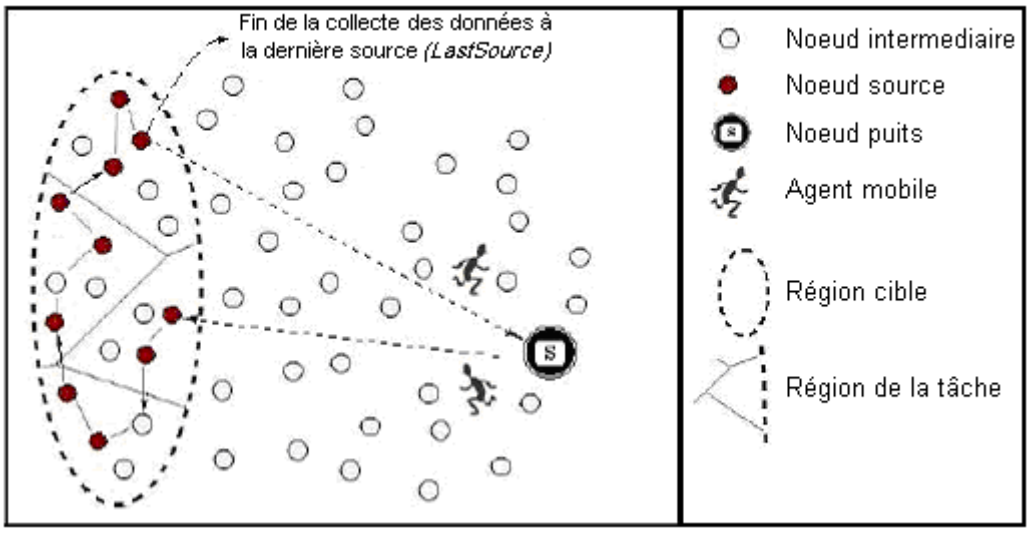

Figure III.2 : Architecture de MAWSN.

Le contexte d'application de MAWSN et sa conception mettent en évidence quelques hypothèses :

- Le puits connaît l'ensemble des nœuds sources qui seront visités par l'AM.
- L'itinéraire de l'AM est déjà conçu avant que le nœud puits ne l'expédie.

Dans cette approche, un puits envoie des requêtes à plusieurs cibles simultanément par le biais d'un agent mobile. Les données dans la région cible sont collectées par l'agent mobile auprès des cibles une par une, et l'ensemble des différentes tâches combinées seront exécutées l'une après l'autre, de sorte que la totalité du traitement prendra plus de temps. Si la qualité de service minimale requise (par exemple, borne de latence) n'est pas violée, en particulier dans le cas où la région cible est éloignée du nœud puits, le gain d'énergie de cette combinaison d'exécution peut s'avérer important.

Pour cela, un modèle de requête de tâches combinées permettant aux applications d'initier plusieurs tâches est nécessaire. En principe, une tâche combinée a trois caractéristiques :

1- Toute tâche appartenant à une tâche combinée peut être traitée par la partie du code de traitement commun d'agent mobile, ainsi un traitement supplémentaire par l'agent mobile ne produira pas un coût de communication supplémentaire.

2- Relativement à la distance entre le centre de la région des tâches combinées et le nœud puits, les régions des tâches sont susceptibles d'être très proches les unes des autres.

3- Toutes les tâches sont demandées simultanément par l'application.

III.5.2. Réduction de la redondance d'information et des coûts de communication

Dans cette partie, nous examinons comment un AM peut être dédié pour réduire la redondance d'information et le surcoût de communication en trois niveaux afin de prolonger la durée de vie du réseau :

III.5.2.1. Niveau nœud : *Élimination de la redondance d'application par traitement local via l'AM*

Comme plusieurs applications sont exécutées, et en raison des contraintes de capacité de stockage, il est impossible de stocker chaque application dans les mémoires locales des capteurs embarqués. Ainsi, une méthode de déploiement dynamique d'une nouvelle application est nécessaire.

Comme illustré dans la figure III.2, le nœud puits attribue à l'AM le code de traitement (comportement) basé sur le besoin d'une application spécifique. Le code transporté par l'AM exige un traitement local des données brutes au niveau des nœuds sources comme l'a demandé

l'application. Ce comportement permet une réduction de la quantité de données transmises en permettant seulement à l'information pertinente d'être extraite et transmise.

III.5.2.2. Niveau tâche : *Élimination de la redondance spatiale par agrégation des données via l'AM*

Le degré de corrélation des données capturées entre les capteurs est étroitement lié à la distance séparant les capteurs, de sorte qu'il est très probable que des capteurs proches l'un à l'autre génèrent des redondances des données capturées. L'AM agrège les données individuelles capturées quand il se rend dans chaque source. Bien que cette technique d'agrégation est généralement utilisée dans les protocoles de dissémination de données basés sur la clusterisation ou l'arbre d'agrégation, l'agrégation assistée par AM ne nécessite aucun coût supplémentaire pour construire ces structures spéciales.

III.5.2.3. Niveau tâche combinée : *Réduction du coût de communication par la concaténation des données de multiples tâches via l'AM*

MAWSN propose une technique d'unification de paquet, qui concatène les données de plusieurs petits paquets en un plus grand paquet afin de réduire les coûts de communication au niveau de la tâche combinée. Grâce à la concaténation des données, les fonctions cycliques et le coût de communication des capteurs intermédiaires peuvent être réduits afin d'augmenter la durée de vie du réseau. Toutefois, ces économies d'énergie peuvent être obtenues généralement au détriment de la prolongation du temps de latence des données.

III.5.3. Analyse

L'approche agent mobile peut s'avérer intéressante pour la dissémination et la collecte des données dans les réseaux de capteurs ; grâce à la réduction des coûts de communication, mais cela reste toujours dépendant de l'environnement d'application notamment les exigences de fiabilité.

Avantages

- Réduction des coûts de communication grâce à l'utilisation d'AM et des tâches combinées.
- Gain en mémoire de stockage des capteurs grâce à l'utilisation d'AM.
- Agrégation importante des données grâce à l'élimination des redondances d'applications et spatiales.

- Partage de la charge entre tous les nœuds du réseau.

Inconvénients

- Grande latence à cause de la combinaison de toutes les tâches.
- Il faut définir à l'avance la séquence de visite des sources et le chemin de l'AM, ce qui n'est pas toujours possible notamment au passage à l'échelle.
- Manque de robustesse, par exemple en cas de défaillance d'un lien ou d'un nœud au moment de la transmission de l'AM.

III.6. MADD [Che-07]

L'approche MAWSN, proposée dans [Che-06] est effectivement intéressante pour minimiser le coût de communication en réduisant l'overhead, l'énergie consommée et la mémoire occupée pour accomplir les tâches du réseau de capteurs.

Cependant, l'approche agent mobile adoptée dans MAWSN repose sur de fortes hypothèses et suppositions irréalistes qui sont elles mêmes considérées comme de grands problèmes dont plusieurs dont plusieurs recherches se font sur ca. En effet, l'approche MAWSN se base sur le fait que l'AM connaît auparavant les nœuds sources à visiter, la séquence de visite de ces nœuds ainsi que le chemin à suivre pour accomplir la tâche du réseau.

Pour cette raison, le concepteur de MAWSN a lui-même proposé une architecture agent mobile combinée avec Directed Diffusion, MADD [Che-07] (*Mobile Agent based Directed Diffusion*), en essayant d'éliminer le maximum d'hypothèses via la combinaison de l'approche Directed Diffusion avec l'approche agent mobile.

L'ordre des nœuds sources à visiter par l'AM peut avoir un impact significatif sur la consommation d'énergie. Trouver une séquence optimale des nœuds sources à visiter est un problème NP-difficile. Dans [QWu-04], une solution basée sur un algorithme génétique pour traiter une solution approximative est présentée. Quoiqu'une optimisation globale puisse être achevée en utilisant un algorithme génétique, ce n'est pas une solution légère pour les réseaux de capteurs qui sont limités en énergie. Pour cette raison, MADD adopte une solution basée sur des gradients pour décider dynamiquement du chemin de l'AM.

III.6.1. Sélection des nœuds sources et établissement des gradients

Le protocole MADD est basé sur l'original DD [Int-00] (two-phase pull diffusion). Dès qu'une nouvelle tâche est reçue en demande par une application, le puits diffuse initialement un paquet d'intérêt afin de localiser les sources qui vont traiter la tâche et établir les gradients.

La diffusion initiale de l'intérêt permet à chaque nœud capteur (nœud intermédiaire ou nœud source) de mettre en place des gradients d'exploration qui seront utilisés pour délivrer les messages d'exploration destinés à la mise en place des routes et leur maintenance. Lorsque les nœuds sources dans la région cible reçoivent les paquets d'intérêt, ils diffuseront des données d'exploration vers le puits individuellement.

Puisque le but final est la détection d'événements dans le réseau de capteurs, le nœud puits doit arrêter le traitement des messages d'exploration qui s'écoulent s'il considère que le nombre de nœuds sources est assez grand pour atteindre l'objectif de détection de l'événement. Ainsi, tous les nœuds sources ou seulement un sous ensemble de ces nœuds seront choisis pour être visités par l'AM. Le nœud puits décide donc de la liste des nœuds sources qui seront visités par l'AM sans avoir à déterminer leur séquence de visite, celle-ci sera décidée dynamiquement au fur et à mesure que l'AM se déplace entre les nœuds. Dans cette liste, il y a seulement deux nœuds sources dont les positions sont importantes, à savoir le premier et le dernier nœud source que l'AM devrait visiter.

III.6.2. Génération et expédition de l'AM

Après avoir décidé des nœuds sources à visiter, le puits va générer un AM, et l'expédie vers le premier nœud source. En même temps, le puits renforce le chemin vers le dernier nœud source. Lorsque l'AM arrive au premier nœud source, il sera sauvegardé dans sa mémoire locale. On divise la période de la tâche entière en plusieurs cycles séquentiels, où chaque cycle requiert l'AM pour visiter tous les nœuds ciblés et retourner le résultat des données collectées vers le puits.

L'AM commence son parcours à partir du premier nœud source (ou à partir du puits seulement au premier cycle) jusqu'à arriver au dernier nœud source. Finalement, l'AM va transporter le résultat des données vers le puits via le chemin renforcé. Dans le premier cycle, en plus que l'AM se déplace d'une source à l'autre pour collecter et agréger l'information, il copie aussi le code de traitement dans la mémoire locale de chaque nœud source. Au début de

chaque cycle, le premier nœud source va construire un autre AM à partir de sa mémoire et l'expédie pour initialiser un nouveau cycle. Puisque le code de traitement a déjà résidé dans la mémoire de chaque nœud source après le premier cycle, l'AM ne va plus transporter ce code de traitement dans les prochains cycles. Lorsque la tâche entière sera terminée, tous les nœuds sources vont se débarrasser du code de traitement.

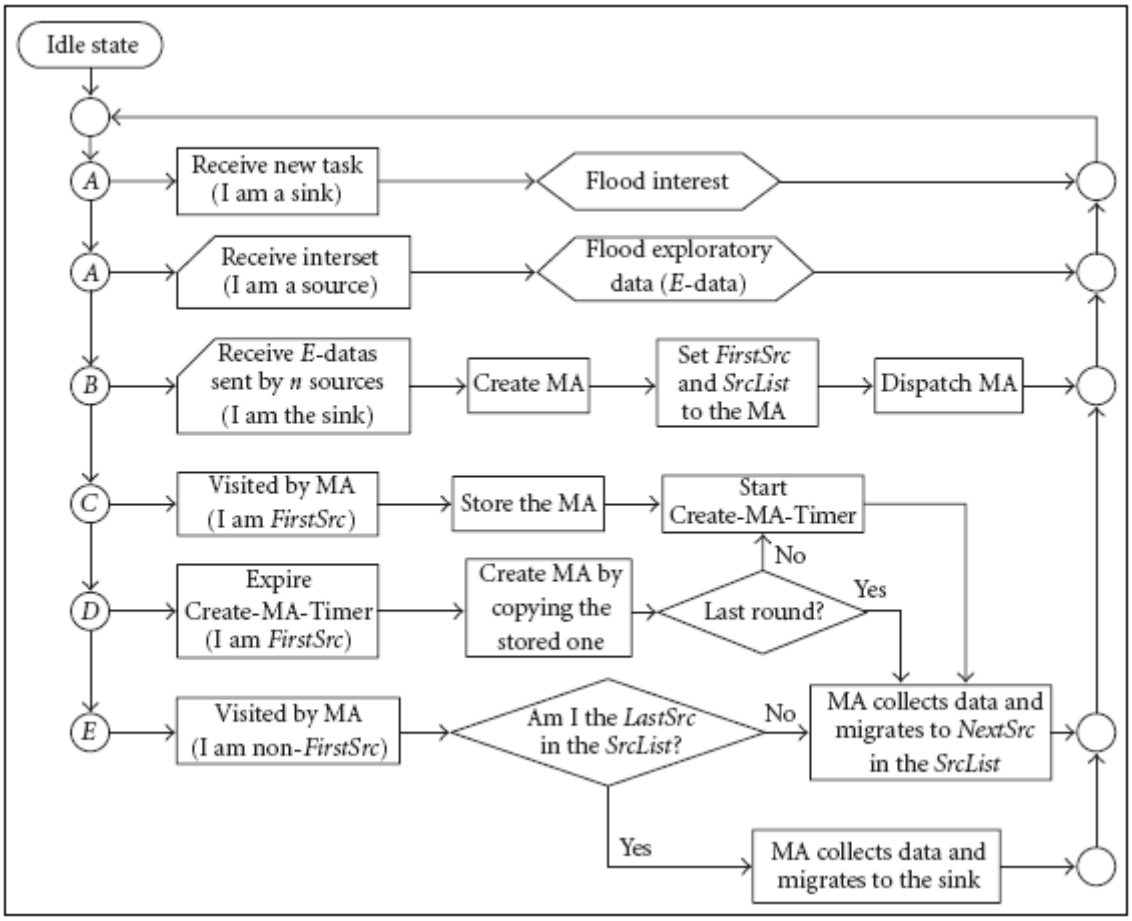

Figure III.3 : Organigramme du protocole MADD.

III.6.3. Séquence de visite des nœuds sources et routage de l'AM

Dans MADD, les messages d'exploration diffusés par les nœuds sources ciblés permettent aux nœuds capteurs de mettre en place une structure *ToSourceEntry*, qui est un type de gradient vers chaque nœud source ciblé. ToSourceEntry est utilisé par l'AM pour parcourir les nœuds sources. Parmi tous les voisins d'un nœud capteur, seul le voisin qui a transmis en premier le message d'exploration d'une source ciblée spécifique va être choisi comme prochain saut dans le ToSourceEntry.

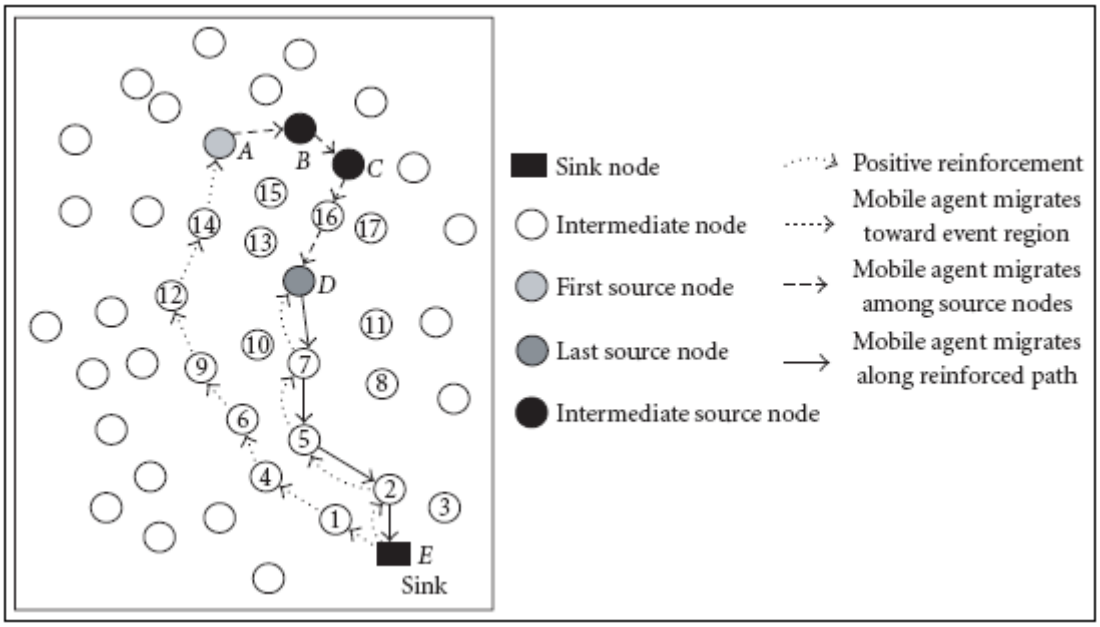

Figure III.4 : Routage de l'AM dans MADD.

En se basant sur les gradients et ToSourceEntry, la route de migration de l'AM est décidée grâce aux 3 éléments suivants :

Choisir FirstSource et LastSource : La taille d'un AM est au minimum dans FirstSource alors qu'elle devient au maximum dans LastSource. Donc, pour réduire l'overhead total de communication, FirstSource doit être le plus loin capteur ciblé à partir du puits, alors que LastSource devrait être le plus proche. Dans MADD, la source ciblée qui est la dernière (première) à envoyer les messages d'exploration, au puits est choisie comme FirstSource (LastSource).

Décider de la séquence de visite des sources : à l'exception de firstsource et LastSource qui sont choisi par le puits, la séquence de visite des autres nœuds sources est décidée dynamiquement par chaque capteur ciblé dans SrcList. Par exemple, lorsqu'un AM arrive au nœud A dans la figure III.4, le nœud va choisir le prochain nœud le plus proche dans sa ToSourceEntry montré dans la première colonne de la figure III.5. Puisque la plus basse latence du nœud B est la plus petite, il en conclu que le nœud B est le nœud source le plus proche du nœud A et il est choisi comme NextSrc.

ToSourceEntry (exploratory message SeqNum = 5)					
	Source	*A*	*B*	*C*	*D*
A	*NextHop*	—	*B*	*B*	14
	Lowest Latency (ms)	—	4.46	8.24	16.32
B	*NextHop*	*A*	—	*C*	*C*
	Lowest Latency (ms)	4.47	—	4.43	12.89
C	*NextHop*	*B*	*B*	—	16
	Lowest Latency (ms)	8.16	4.32	—	8.52
16	*NextHop*	15	*C*	*C*	*D*
	Lowest Latency (ms)	9.65	7.56	4.86	5.08
D	*NextHop*	10	16	16	—
	Lowest Latency (ms)	14.15	12.67	8.73	—

Figure III.5 : La structure ToSourceEntry dans MADD.

Trouver le prochain nœud saut pour router l'AM le long de la route entière du puits à la source, d'une source à l'autre et de la source au puits : Expédié par le puits, un AM migre de FirstSource de la même manière que les messages de renforcement sont expédiés dans l'original DD. Lorsque l'AM migre le long des sources ciblées, son prochain nœud saut sera décidé suivant le ToSourceEntry courant du nœud. L'AM va retourner au puits en utilisant un chemin renforcé (par exemple le chemin D-7-5-2-E dans la figure III.4).

III.6.4. Élimination de la redondance d'application et agrégation des données

L'approche agent mobile de MADD, de la même manière que MAWSN, permet d'éliminer la redondance d'application grâce au traitement local assisté par AM. Cette procédure permet une réduction dans la quantité des données à transmettre puisque seulement l'information pertinente sera extraite et transmise. Aussi, Le degré de corrélation des données capturées entre les capteurs est étroitement lié à la distance séparant les capteurs, de façon à ce qu'il est très probable pour des capteurs proches de générer des données redondantes. D'ailleurs, dans DD, différents paquets de données qui sont complètement ou partiellement redondants, sont transmis au puits via différents chemins avec une faible probabilité d'être agrégés. Cette technique d'agrégation peut être considérée comme une agrégation opportunistique. Par contre, dans MADD, l'AM agrège les données capturées individuellement lorsqu'il visite chaque nœud source.

III.6.5. Analyse

La combinaison des deux approches DD [Int-00] et agent mobile permet de combler plusieurs problèmes. Ceci permet de bénéficier des avantages de chacune de ces deux approches, mais aussi d'hériter de leurs inconvénients.

Avantages

- Gain en mémoire de stockage des capteurs grâce à l'utilisation d'AM.
- Choix de FirstSource et LastSource selon leur éloignement du puits permet d'économiser l'énergie de communication.
- Agrégation importante des données grâce à l'élimination des redondances d'applications et spatiales.
- Améliore la latence grâce au choix de la source la plus proche dans la séquence de visite des sources par l'AM.
- Permet un bon passage à l'échelle grâce à la décision dynamique du chemin de l'AM et donc pas besoin de déterminer le chemin complet de l'AM avant son expédition.

Inconvénients

- Le problème d'overhead dans DD durant la première phase apparaît toujours dans MADD lors de la diffusion initiale de l'intérêt et des données.
- Séquence de visite des nœuds source décidée au fur et à mesure par le choix du nœud source le plus proche n'assure pas que le chemin complet est meilleur (problème NP-difficile).
- Non prise en considération de :
    - Possibilité de changement de topologie (mobilité, défaillance de nœuds ou de liens).
    - Possibilité d'apparition de nouvelles ou de meilleures sources dans le réseau.

III.7. AbDD [Sha-08]

Un des inconvénients de DD, comme tout modèle client/serveur, est que chaque nœud a besoin de sauvegarder chaque code d'application possible dans sa mémoire, ce qui est irréaliste pour un capteur embarqué avec contrainte d'énergie et de mémoire. Aussi, l'agrégation des données dans DD est minime. En effet, différents paquets de données qui sont complètement ou partiellement redondants sont expédiés vers le puits avec une

agrégation opportunistique. Pour cette raison, l'approche AbDD [Sha-08] (Agent based Directed Diffusion) emploi le modèle agents combiné avec Directed Diffusion afin d'assurer une meilleure durée de vie du réseau.

L'approche AbDD se base sur les hypothèses suivantes :

- Chaque nœud capteur opère sur le principe de localisation, i.e. : chaque nœud connaît son coût de routage, son énergie restante, le coût du routage de ses voisins et leur énergie restante, ceci via un simple protocole Hello entre voisins [Liu-04].

- La distance entre deux nœuds sources est toujours plus petite que la distance entre un nœud source et le nœud puits.

III.7.1. Routage

Afin d'assurer qu'un chemin de routage ne sera pas épuisé très tôt, AbDD adopte un schéma combiné [Liu-04], qui prend en considération et le coût du routage Ci (nombre de sauts vers le destinataire) et l'énergie restante Ei d'un nœud Ni. L'agent mobile au nœud Ni sélectionne le nœud voisin Nj, en se basant sur la valeur H, en utilisant la formule suivante :

$$H_1 = \underset{j \in FT_i}{MAX} \frac{E_j}{C_j}$$

Pour effectuer ces calculs, chaque nœud met en place une table d'expédition des données (FT) basée sur les informations locales obtenues dans la première phase. En mettant en place cette table, le système peut garantir l'arrivée du paquet de données vers le puits.

Afin que les nœuds mettent à jour leur table d'expédition, ils utilisent un transfert de paquet simple en utilisant un protocole Hello [Liu-04] pour échanger périodiquement entre voisins les informations concernant l'énergie restante, le coût pour atteindre le puits et les nœuds sources.

L'objectif principal de cette approche est de fournir un routage efficace en énergie dans le réseau de capteurs, pour obtenir des chemins de routage optimaux afin de prolonger la durée de vie des capteurs le long de chaque chemin.

III.7.2. Cycles de vie des agents

La figure III.6 montre l'environnement qui inclue le système basé agents. Cela consiste en deux types d'agents ; Agents Stationnaires (AS) et Agents Mobiles (AM).

L'AS réside dans chaque nœud source. Il surveille toutes les activités pour le nœud et met à jour ses connaissances en conséquence. Ces connaissances sont critiques et sont utilisées par l'AM pour prendre des décisions de sélection du prochain saut. L'AS maintient le cache d'intérêt, les modèles des autres, son modèle et les coûts d'initialisation du routage.

Chaque item dans le cache correspond à un intérêt distinct. Les principaux éléments constituant le cache d'intérêt sont :

- Gradient : L'état de direction envers un voisin duquel un intérêt a été reçu.
- Intervalle : Il contient un intervalle demandé par le voisin spécifié pour expédier un AM.
- Durée : C'est la durée de vie de l'intérêt.

De plus, chaque nœud maintient un coût de routage pour l'intérêt correspondant. Le coût de routage d'un nœud est le nombre de sauts requis pour atteindre le puits et le nœud source. Pour chaque intérêt, chaque nœud maintient le coût de route de ses voisins, i.e. : à combien de sauts est éloigné le puits et le nœud source ?

Les cycles de vie des agents et les processus d'interaction sont décrits via les trois étapes suivantes :

- Première étape : Identification des nœuds sources qui satisfont l'intérêt et établissement des tables de coûts.
- Deuxième étape : Distribution des codes d'applications spécifiées au nœud source.
- Dernière étape : Agrégation et envoie des données des nœuds sources vers le puits via l'AM. épargnée

III.7.2.1. Étape initiale : Lorsque le puits reçoit une nouvelle tâche par une application, il expédie un AM pour trouver les nœuds sources qui vont exécuter la tâche et pour initialiser le coût pour chaque nœud source.

Le puits envoie les AMs équipés avec l'intérêt spécifié vers ses voisins immédiats. Les ASs résidants dans les nœuds voisins prennent l'intérêt à partir des AMs et les enregistrent dans

leur cache d'intérêt. Le cache d'intérêt consiste en un tuple d'intérêts reçus à partir des nœuds voisins. Un message d'intérêts est une requête ou interrogation qui spécifie ce que veut l'utilisateur. Chaque intérêt contient une description d'une tâche de captage qui est supportée par le réseau de capteurs pour acquérir la donnée. Un AS mets en place des gradients pour chaque AM le visitant. Un gradient est une direction créée au niveau de chaque nœud recevant un intérêt. La direction du gradient est mise envers le nœud voisin duquel l'AM a migré. Si un AS dans un nœud, trouve qu'il satisfait les contraintes de l'intérêt, il renvoie l'AM le long du gradient. Sinon, l'AS expédie simplement l'AM vers son voisin immédiat.

Une fois l'AM arrive vers un nœud source qui satisfait l'intérêt, il reprend son chemin de retour vers le puits le long des gradients établis. Les nœuds sources mettent l'identifiant de coût (Ci) de l'AM à la valeur 1 et le renvoie en retour le long des gradients.

Ci est le coût requis pour un AM pour atteindre le nœud source. Chez un nœud donné N, lorsque l'AM migre d'un saut vers le voisin du nœud N en retour vers le puits ; sa valeur Ci est incrémentée par 1. Le nœud source met aussi son unique identificateur dans l'AM, afin de s'identifier envers le puits. Ensuite, chaque nœud met à jour les coûts de ses voisins. Aussitôt que l'AM atteint de retour le puits, la phase initiale de découverte de nœuds sources et l'initialisation de la table de coûts est terminée.

III.7.2.2. Deuxième étape : le puits expédie l'AM vers les nœuds sources avec le code d'application spécifié. À la différence de l'approche client/serveur, où le code de l'application est sauvegardé dans chaque nœud capteur, le code de traitement dans AbDD est sauvegardé dans les nœuds sources seulement jusqu'à expiration du temps d'intérêt (le temps d'exécuter la requête dans le réseau). La séquence de visite des nœuds sources est déterminée par le puits avant d'expédier l'AM. Ce dernier va se déplacer à travers les nœuds capteurs pour atteindre les nœuds sources. À chaque saut, il consulte les modèles des autres nœuds et fait sa décision pour sélectionner le prochain saut (voisin), en se basant sur **le niveau de batterie maximum** et **le coût de route minimum**.

Lorsqu'un AM atteint le premier nœud source, il distribue la copie du code d'application spécifiée à l'AS du nœud source. L'AM va se cloner au niveau du premier nœud source et démarre alors son Timer (Intervalle), puis se déplace jusqu'au dernier nœud source en distribuant une copie du code d'application spécifiée à chaque nœud source. L'AM expire dès qu'il délivre le code source de l'application au dernier nœud source.

III.7.2.3. Dernière étape : celle-ci dépend de l'intervalle de fréquence (un paramètre de l'intérêt spécifiant le taux d'envoie d'évènement), l'AM au premier nœud source va se cloner après l'intervalle spécifié. Ensuite, il collecte la donnée, restore le Timer, puis migre vers le prochain nœud source. Á chaque nœud source, l'AM agrège la donnée sensorielle, si possible. De cette manière, l'AM continue son parcours vers le puits pour rapporter la donnée agrégée. À son arrivée vers le puits, il va faire passer la donnée transportée des nœuds sources vers le puits et après il meurt. Le nombre de fois que le Timer est réinitialisé dépend du champ Durée de l'intérêt. Une fois la Durée de l'intérêt expire, l'AM supprime tous les codes d'applications spécifiées de tous les nœuds sources.

III.7.3. Analyse

L'approche AbDD [Sha-08] utilise les informations d'énergie et du coût de routage pour permettre un routage de l'AM efficace en énergie. Cette technique hérite aussi des caractéristiques des deux approches DD [int-00] et agent mobile.

Avantages

- Gain en mémoire de stockage des capteurs grâce à l'utilisation d'AM.
- Prolongation de la durée de vie du réseau en utilisant un routage efficace en énergie grâce à la formule du coût de routage.
- Permet de s'adapter aux changements de la topologie du réseau en échangeant périodiquement les informations concernant le routage par l'utilisation des messages Hello.

Inconvénients

- Le problème d'overhead dans DD durant la première phase apparaît toujours dans AbDD lors de la diffusion initiale de l'AM et l'établissement des tables des coûts.
- Sélection des nœuds sources une seule fois, donc les sources resteront les mêmes jusqu'à la fin de la tâche. Cependant, le mouvement de la cible et le changement de l'événement peut mener à de nouvelles sources qui ne seront donc pas identifiées pour la tâche courante.
- Absence de méthode de construction de la séquence de visite des nœuds sources.

III.8. Conclusion

En déplaçant le code de traitement vers la donnée, un AM peut éviter la transmission des données intermédiaires dans le réseau, continuer le travail même en présence de déconnexions dans le réseau, et compléter donc l'ensemble de la tache plus rapidement que dans les solutions traditionnelles client/serveur.

Cependant, un AM n'est pas toujours meilleur qu'une solution client/serveur. Par exemple, si le code de l'agent est plus grand que la totalité des données intermédiaires, l'AM sera donc moins performant dans ce cas, puisqu'il va transférer plus d'octets à travers le réseau par rapport à la solution client/serveur.

De plus, si le réseau est assez rapide, l'agent sera moins performant même si le code est plus petit. Avec un réseau rapide et fiable, l'interprétation de l'agent au niveau des capteurs est moins rapide que la transmission des données intermédiaires vers le puits. Cependant, si la vitesse du réseau et sa fiabilité chutent, ou la taille des données augmente, le cas change considérablement.

Décider donc d'une meilleure approche (client/serveur ou agent mobile) pour les réseaux de capteurs sans fil, n'est pas un choix fixe ; mais cela dépend fortement des caractéristiques et de la topologie du réseau ainsi que de son application.

Chapitre IV

MLBDD (Mobile Line Based Data Dissemination)

Chapitre IV : MLBDD (*Mobile Line Based Data Dissemination*)

IV.1. Introduction

Le paradigme de communication dans les réseaux de capteurs est en général de type N-vers-1, où les capteurs collectent des données de l'environnement qui les entoure, et les disséminent vers le nœud puits [Ben-07]. Cependant, ce n'est pas toujours évident pour le puits de localiser les nœuds sources, et aussi pour les nœuds sources de localiser le puits pour lui envoyer les données, notamment lorsque celui-ci est mobile. Des solutions intuitives [Int-00] [Che-07] utilisent des inondations pour atteindre les sources et le puits, mais cela risque de surcharger le réseau et de créer des problèmes de congestion. Afin d'optimiser la dissémination, nous avons opté pour une solution agent mobile [Che-06] basée sur une zone de rendez-vous inspirée de l'approche LBDD [Ben-07].

L'intérêt de la zone de rendez-vous est d'éviter la recherche aveugle des données et du puits dans le champ de déploiement. Pour cela, nous allons attribuer à certains nœuds, situés dans la zone de rendez-vous, le rôle de sauvegarde des données générées par tous les capteurs du réseau. Ainsi, chaque nœud source qui détecte un évènement dans le champ de capture, va générer la donnée correspondante et l'envoyer directement vers la zone de rendez-vous, où elle sera stockée. Par la suite, Lorsqu'un nœud puits désire collecter une certaine information, il va générer une requête qui sera aussi dirigée directement vers la zone de rendez-vous, où toutes les données y résident. De cette façon, on évite le problème de surcharge du réseau par l'élimination des diffusions des messages de recherche des données dans tout le réseau.

Afin de collecter les données stockées dans la zone de rendez-vous d'une manière efficace, nous utiliserons un agent mobile qui aura pour mission d'agréger les données collectées dans le but de réduire la taille du paquet transmis vers le puits et de diminuer ainsi la consommation d'énergie et la bande passante qui représentent des ressources critiques dans les réseaux de capteurs sans fil.

IV.2. Environnement et hypothèses

Notre approche est utilisée pour des applications continues où les nœuds génèrent périodiquement des données, et aussi pour des applications event-driven où les nœuds génèrent des données lorsqu'un évènement surgit dans le champ de capture. La solution

proposée va accomplir plusieurs tâches simultanées associées avec de petites quantités de données, transportées par un seul paquet. Le résultat sera concaténé dans un seul paquet en permettant une meilleure agrégation des donnes afin de diminuer les coûts de communication [Che-06].

Afin de construire la zone de rendez-vous, nous allons utiliser une structure sous forme d'une bande rectangulaire placée dans la zone d'intérêt. Pour cela, nous supposons que le réseau est muni d'un système de localisation, par exemple un GPS [Hof-97]. De cette manière, chaque nœud est supposé connaître sa position géographique ainsi que les coordonnées de la zone d'intérêt, ce qui implique que chaque nœud peut savoir à n'importe quel instant s'il appartient à la zone de rendez-vous.

De plus, les nœuds doivent être synchronisés, afin de pouvoir faire le changement de la zone de rendez-vous d'une façon périodique.

IV.2.1. Synchronisation des capteurs par GPS

La solution la plus connue, et aussi la plus précise pour synchroniser des dispositifs sans fil, est celle du GPS. Tout module GPS fournit chaque seconde deux informations au processeur auquel il est connecté : la trame de géolocalisation (contenant la localisation et l'heure absolue GMT) et le signal PPS. Ce dernier est un simple signal carré périodique dont la période est de 1 s ± 50 ns, c'est-à-dire une période d'une seconde extrêmement précise et stable. À tout instant, tous les modules GPS situés dans un même cône d'influence satellitaire délivrent des tops PPS précis et parfaitement synchrones (moins de 50 ns d'écart).

La solution par module GPS présente de nombreux intérêts, en premier lieu une très bonne précision, de l'ordre de la µs, voire moins, indépendamment du nombre de capteurs du système, de la qualité du réseau et de sa topologie, ainsi que des distances entre capteurs. Cette haute précision, sans dérive et calée sur l'heure universelle, rend le système et ses algorithmes parfaitement déterministes d'un point de vue temporel.

Les inconvénients de cette solution résident dans la nécessité d'ajouter un petit module GPS qui consomme de l'énergie et augmente un peu le volume du capteur et dans le fait que le capteur ne peut être synchrone qu'en extérieur (outdoor). Mais les développements récents donnent chaque jour des réponses appropriées à ces limitations. Tout d'abord les modules GPS sont toujours plus miniaturisés, au point d'être désormais intégrables directement à la

carte mère du capteur comme un composant parmi d'autres. De plus, leur consommation est toujours plus faible (les constructeurs tenant compte du phénomène de nomadisme) et, pour des utilisations intérieures pour lesquelles les signaux satellitaires seraient filtrés par les caissons en béton des tabliers de pont par exemple, des solutions de réémetteur GPS sont disponibles [LeC-08].

IV.2.2. Routage géographique

Un routage est dit géographique lorsque les décisions de routage sont basées sur les positions des nœuds [Erm-04].

Puisque notre solution utilise un GPS, ainsi les positions des nœuds sont disponibles et il est donc plus convenable et plus qualifié d'utiliser un routage géographique pour acheminer les données dans la zone d'intérêt.

Dans un routage géographique, un nœud qui doit envoyer un paquet sélectionne comme prochain saut le nœud qui est « dans la direction » du destinataire et qui produit « le meilleur » avancement vers lui [Nat-08].

Le mécanisme de relais généralement suit le paradigme connu glouton (Greedy Forwarding) [Med-09]: lorsqu'un nœud capteur a un paquet à transmettre, le nœud de relais le plus proche de la destination est choisi pour transférer le paquet. Ce principe glouton est appliqué jusqu'à ce que la destination finale soit atteinte.

Messages Hello : le protocole de routage sans fil utilise généralement un processus de découverte de voisinage qui est réalisé grâce à l'envoi périodique de messages Hello. Chacun de ces messages contient l'identifiant du nœud émetteur et éventuellement d'autres informations (position géographique, énergie restante, mémoire libre…). Cet échange périodique permet de construire et de maintenir une table de voisinage contenant la liste des nœuds à portée de communication.

IV.3. Principe

LBDD [Ben-07] définit une structure sous la forme d'une bande, de largeur g, placée au milieu de la zone d'intérêt. Celle-ci est utilisée comme une zone de rendez-vous pour les requêtes envoyées par les puits et les données générées par les nœuds sources. Cependant, comme déjà cité dans le chapitre 2 section II.7.4 ; le principal problème causé par cette

approche est la surcharge des nœuds placés dans la zone de rendez-vous. En effet, LBDD suppose que cette zone est placée au milieu de la zone d'intérêt, et elle est donc statique. De ce fait, la concentration de la charge au milieu de la zone d'intérêt va engendrer un affaiblissement rapide du taux d'énergie de ces nœuds par rapport aux autres nœuds du réseau n'appartenant pas à la zone de rendez-vous. Cela risque de partitionner le réseau en deux parties injoignables, ce qui va immobiliser le système et diminuer donc la durée de vie du réseau.

Comme le facteur d'énergie constitue l'une des contraintes les plus importantes qui guident la conception des protocoles dans les réseaux de capteurs, ils doivent donc intégrer des mécanismes qui permettent aux utilisateurs de prolonger la durée de vie du réseau en entier.

Pour cette raison, nous avons conçu l'approche MLBDD (*Mobile Line Based Data Dissemination*) qui a pour principal objectif la prolongation de la durée de vie du réseau, en utilisant une approche de dissémination des données équitable en énergie qui limite le coût en énergie et en communication tout en assurant un équilibre de la consommation d'énergie des nœuds du réseau.

L'approche MLBDD emploie une zone de rendez-vous sous forme d'une bande rectangulaire. Les nœuds positionnés à l'intérieur de la bande sont appelés *inline-nodes*. Contrairement à LBDD où cette bande est placée au milieu de la zone d'intérêt, MLBDD utilise une bande mobile (périodiquement, un nouvel ensemble de nœuds devra assurer le rôle des inline-nodes) afin d'éviter la surcharge des nœuds placés au milieu de la zone d'intérêt et d'équilibrer donc la charge dans tout le réseau. Il s'agit donc de proposer un mécanisme efficace pour la gestion d'une ligne dynamique mobile.

Aussi, MLBDD utilise un agent mobile inspiré de [Che-06] pour collecter les données dans la zone de rendez-vous en employant des fonctions d'agrégation afin de réduire la quantité des données transmises et d'économiser ainsi l'énergie de communication et la bande passante utilisée.

IV.4. Fonctionnement de MLBDD

IV.4.1. Mobilité de la zone de rendez-vous

Dans notre approche, la zone d'intérêt est divisée en N bandes rectangulaires qui ne s'interfèrent pas (voir figure IV.1) et chaque bande est également divisée en groupes. Á un instant donné, une seule bande i (i € [1,N]) va représenter la zone de rendez-vous actuelle pour les données prélevées par les capteurs et les requêtes des nœuds puits. Après l'écoulement d'une période T, La bande i+1 sera activée comme zone de rendez-vous actuelle au lieu de la bande i, et ainsi de suite. Ce processus sera exécuté périodiquement afin d'équilibrer la charge entre tous les nœuds du réseau et d'améliorer donc la durée de vie du réseau.

La largeur de la bande rectangulaire représentant la zone de rendez-vous et la taille des groupes sont des paramètres déterminants. Nous fixons ces deux valeurs (largeur de la zone de rendez-vous et taille d'un groupe) à la longueur de la portée radio des nœuds capteurs. Cela permet d'atteindre, en un seul saut seulement, tous les nœuds du même groupe.

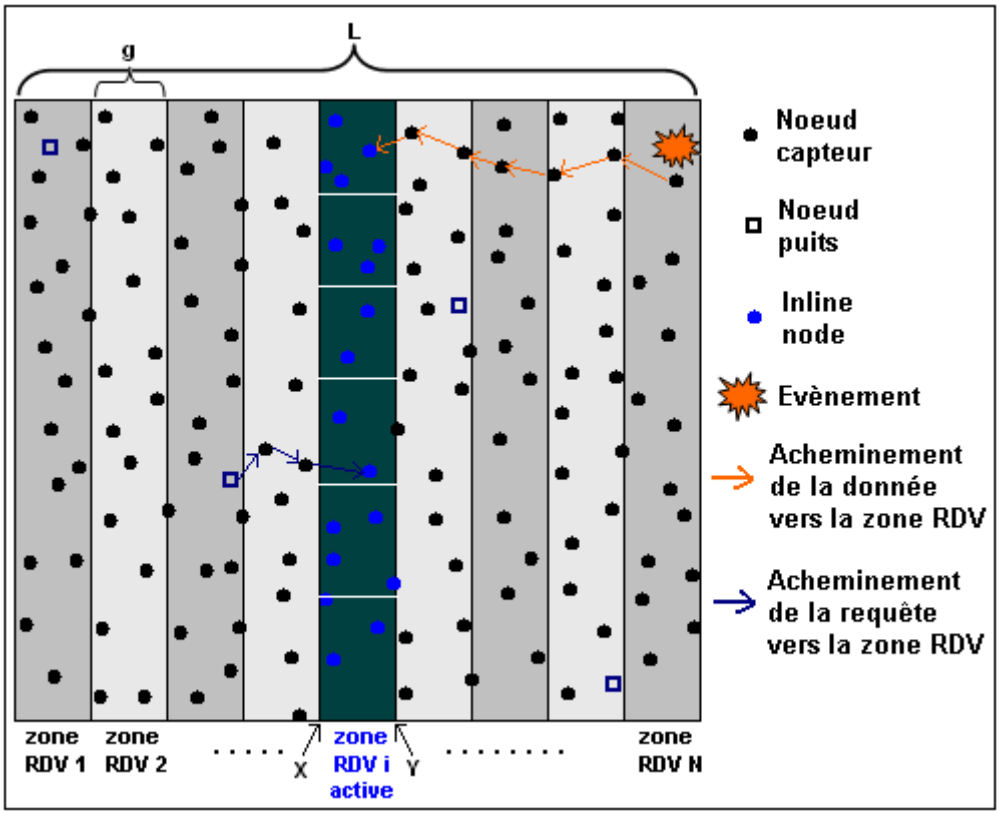

Figure IV.1 : Principe de fonctionnement de MLBDD.

Dès que le changement de zone de rendez-vous se fait, les clusterheads de chaque groupe doivent être spécifiés et reconnus. Le clusterhead de chaque groupe doit avoir assez d'énergie pour pouvoir recevoir l'intérêt du puits et les descripteurs (MétaData) des données à collecter.

Les messages Hello de la zone de rendez-vous vont inclure le taux d'énergie restant ainsi que la capacité de mémoire libre pour chaque inline-node. Ces informations serviront à la sélection des clusterheads et des nœuds hébergeurs des données.

Pour élire un clusterhead, nous nous basons sur le taux d'énergie restant pour chaque inline-node. Le facteur de mémoire libre n'est pas un paramètre à prendre en considération pour l'élection du clusterhead car tous les inline-nodes n'ont encore aucune donnée stockée à l'instant de changement de zone de rendez-vous. Ainsi, dès que le changement de zone de rendez-vous se fait à l'instant T, le nœud ayant le maximum d'énergie dans chaque groupe (le taux d'énergie est une information transmises dans les paquets Hello) sera élu clusterhaed et va propager cette information dans son groupe.

Algorithme :

Chaque nœud du réseau possède dans son cache les coordonnées de la zone de rendez-vous. Périodiquement, chaque nœud va recalculer les coordonnées de la zone de rendez-vous de façon à ce que tous les nœuds du réseau se dirigent vers la même zone de rendez-vous à n'importe quel instant.

T : période de changement de zone de rendez-vous

RDV : représente la zone rectangulaire par 4 variables x, y, v et w.

L : largeur de la zone d'intérêt (toute la zone de déploiement)

g : largeur de la bande rectangulaire (zone de rendez-vous)

P : variable booléenne pour indiquer le sens de déplacement de la zone de rendez-vous (1 : de gauche à droite, 0 : de droite à gauche).

```
Début
Pour chaque période T ;
  Faire :
        Si (RDV.y = L)  alors    P=0
```

```
        Sinon  si (RDV.x = 0)    alors    P=1
        Fsi
        Si (P = 1)          alors    // Remplacer la zone de RDV actuelle par celle de droite.
                                     RDV.x=RDV.y
                                     RDV.y=RDV.x+g
        Sinon                        // Remplacer la zone de RDV actuelle par celle de gauche.
                                     RDV.y=RDV.x
                                     RDV.x=RDV.y-g
        Fsi
  Fait
Fin.
```

De plus, grâce à l'utilisation du système de localisation et à la synchronisation des nœuds, le changement de la zone de rendez-vous ne génère aucun message supplémentaire. En effet, comme les nœuds sont synchronisés, chacun d'eux et à n'importe quel instant, a l'information d'une façon autonome sur la zone de rendez-vous à utiliser pour l'envoie des données et des requêtes.

IV.4.2. Dissémination des données prélevées

Lorsqu'un évènement surgit dans la zone d'intérêt, les nœuds sources l'ayant capté vont générer les données correspondantes ainsi que leur descripteur et les envoyer directement vers la zone de rendez-vous active.

Le descripteur d'une donnée contient les informations nécessaires pour reconnaitre cette donnée par les intérêts du puits. Chaque descripteur doit contenir :

- Le type de la donnée
- Le temps de prélèvement de la donnée
- Identifiant du nœud capteur de cette donnée
- Position du nœud capteur de cette donnée
- …etc.

Puisque les nœuds sont munis d'un système de localisation, les paquets des données seront routés vers la zone de rendez-vous en utilisant le routage géographique glouton. Chaque

nœud va router le paquet d'une façon perpendiculaire à la bande rectangulaire jusqu'à atteindre le premier inline-node. Cela va équilibrer la charge entre les nœuds, éviter les congestions et assurer une construction efficace du chemin d'accès à la donnée.

IV.4.3. Stockage des données

Notre solution MLBDD a pour but de prolonger la durée de vie du réseau en minimisant les coûts de communications et en équilibrant la charge entre les nœuds du réseau.

Pour cela, Nous distinguons entre deux types de données :

- les données volumineuses, dont la taille du descripteur est plus petite que la taille de la donnée prélevée elle-même (photo, vidéo, son, …) ;
- et les données légères, dont la taille du descripteur est plus grande ou égale à la taille de la donnée prélevée elle-même (pression, température, …).

Lorsqu'une donnée volumineuse arrive chez le premier inline-node, il va choisir parmi ses voisins appartenant à son groupe, le nœud ayant un maximum d'énergie restante et un espace suffisant pour héberger la donnée en question.

P : Premier inline-node qui reçoit la donnée disséminée vers la Z-RDV.

{V1, V2, …, Vn} : ensemble des inline-nodes appartenant au groupe du nœud P.

N : L'inline-node qui va héberger la donnée.

Lorsque l'inline-node P reçoit une donnée de taille TD:

```
Max = V1.energie_restante
Pour tout inline-node Vi du groupe
Faire
Si Vi.memoire_libre > TD
        alors si Vi.energie_restante > max
                alors   max = Vi.energie_restante
                        N=Vi
Fait
Envoyer la donnée au nœud hébergeur N
```

Lorsque le nœud hébergeur va recevoir la donnée en question, il va la stocker dans sa mémoire, et envoyer son descripteur au clusterhead de son groupe.

Cependant, lorsqu'une donnée légère arrive chez le premier inline-node, il va l'envoyer directement chez le clusterhead, qui va donc stocker dans sa mémoire la donnée prélevée elle-même et son descripteur.

De cette manière, chaque clusterhead aura dans sa mémoire une table contenant tous les descripteurs des données de son groupe.

Cette solution de stockage permet de minimiser l'overhead. En effet, le coût engendré par le stockage des données dans la zone de rendez-vous se résume simplement en l'envoi des données à un seul saut uniquement et éventuellement l'envoi du descripteur au clusterhead.

Comme un inline-node est à la portée de communication de tous les autres inline-node appartenant à son groupe, cela implique qu'un inline-node peut atteindre, en un seul saut, tous les autres inline-nodes de son groupe. Cela permet de faire participer tous les nœuds de la zone de rendez-vous à la tâche de stockage des données.

IV.4.4. Collecte des données

IV.4.4.1. L'agent mobile et son itinéraire

Lorsqu'un nœud puits désire collecter certaines informations du réseau, il va créer un agent mobile et l'envoyer directement vers la zone de rendez-vous active, en utilisant le même routage géographique employé pour disséminer les données vers la bande rectangulaire. L'agent mobile contient l'intérêt du puits ainsi que le code de traitement des données à collecter (code d'agrégation qui pourrait être différent pour chaque type de données). L'AM va transporter un paquet contenant plusieurs requêtes [Che-06] où chaque requête correspond à une application différente.

Lorsque l'AM arrive chez le premier *inline-node* (nœud appartenant à la zone de rendez-vous actuelle), il sera dirigé vers le clusterhead qui va générer un deuxième AM comportant la liste des requêtes à collecter pour la première fois dans la zone de rendez-vous actuelle pour l'envoyer vers la zone de rendez-vous précédente (voir paragraphe IV.4.4.3.).

Lorsque le clusterhead reçoit un AM, il fait propager l'intérêt du puits dans les deux directions de la bande afin d'atteindre tous les clusterheads de la zone de rendez-vous. Chaque clusterhead recevant l'intérêt, va consulter sa table de descripteurs des données pour en tirer les descripteurs qui correspondent à l'intérêt du puits et les envoyer à l'AM.

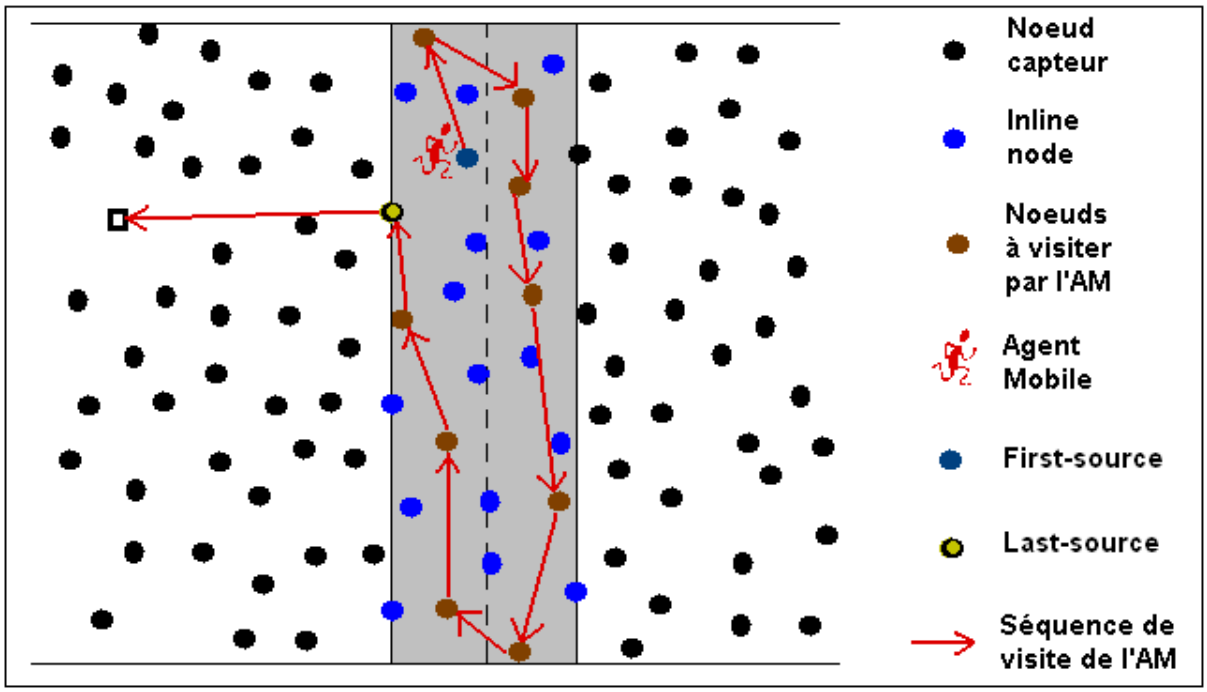

Figure IV.2 : Séquence de visite de l'AM.

Après réception des descripteurs, l'AM va décider des nœuds cibles à visiter pour la collecte des données, et il va établir une séquence logique de visite de ces nœuds, en se basant sur leur position géographique comme suit (figure IV.2) :

- Diviser la bande rectangulaire en deux sous bandes égales.
- Pour chaque sous bande, trier les inline-nodes à visiter en se basant sur leur position géographique, de façon à tracer une ligne d'un bout à l'autre.
- Concaténer les deux sous séquences établies de façon à faire un tour complet de tous les inline-nodes ciblés.
- Choisir le last-source : la taille d'un AM est au minimum dans first-source alors qu'elle devient au maximum dans last-source. Donc, pour réduire l'overhead de communication, last-source doit être le plus proche inline-node ciblé à partir du puits [Che-07].
- Choisir le first-source : puisque les nœuds à visiter par l'AM sont triés en un tour complet et le last-source est déjà choisi en premier, il nous reste donc que deux choix pour le first-source : soit le nœud qui précède immédiatement le last-source soit celui qui le suit immédiatement. Pour la même raison de minimisation de l'overhaed, on élit

comme first-source le nœud le plus loin du last-source entre ces deux nœuds : celui qui précède et qui suit le last-source.

IV.4.4.2. Agrégation des données

Après avoir établie la séquence de visite des *inline-nodes*, l'AM va parcourir les nœuds cibles en fusionnant les données au fur et à mesure qu'il les collecte [Che-06] :

IV.4.4.2.1. ***Elimination de la redondance d'application :*** Avec le développement des WSNs, "un déploiement, plusieurs applications" est une tendance en raison de la nature des applications spécifiques des réseaux de capteurs. En général, en raison des contraintes de capacité de stockage, il est impossible de stocker chaque application dans les mémoires locales des capteurs embarqués. Le noeud puits attribue à l'AM le code de traitement (comportement) basé sur le besoin d'une application spécifique. Le code transporté par l'AM exige un traitement local des données brutes au niveau des inline-nodes comme l'a demandé l'application. Ce comportement permet une réduction de la quantité des données transmises en permettant seulement à l'information pertinente d'être extraite et transmise.

Exemple [Che-07]*:* Associer l'AM avec un code de traitement d'image. L'utilisateur peut ne pas être intéressé par toute l'image prélevée par le capteur, mais seulement par une partie de l'image. Par exemple, la partie du visage seulement.

IV.4.4.2.2. ***Elimination de la redondance spatiale :*** Le degré de corrélation des données captées entre les capteurs est étroitement lié à la distance séparant les capteurs, de sorte qu'il est très probable que des capteurs proches l'un à l'autre génèrent des redondances des données prélevées. L'AM agrège les données individuellement quand il se rend dans chaque inline-node. Bien que cette technique d'agrégation est généralement utilisée dans les protocoles de dissémination des données basés sur la clusterisation ou l'arbre d'agrégation, l'agrégation assistée par AM ne nécessite aucun coût supplémentaire pour construire ces structures spéciales.

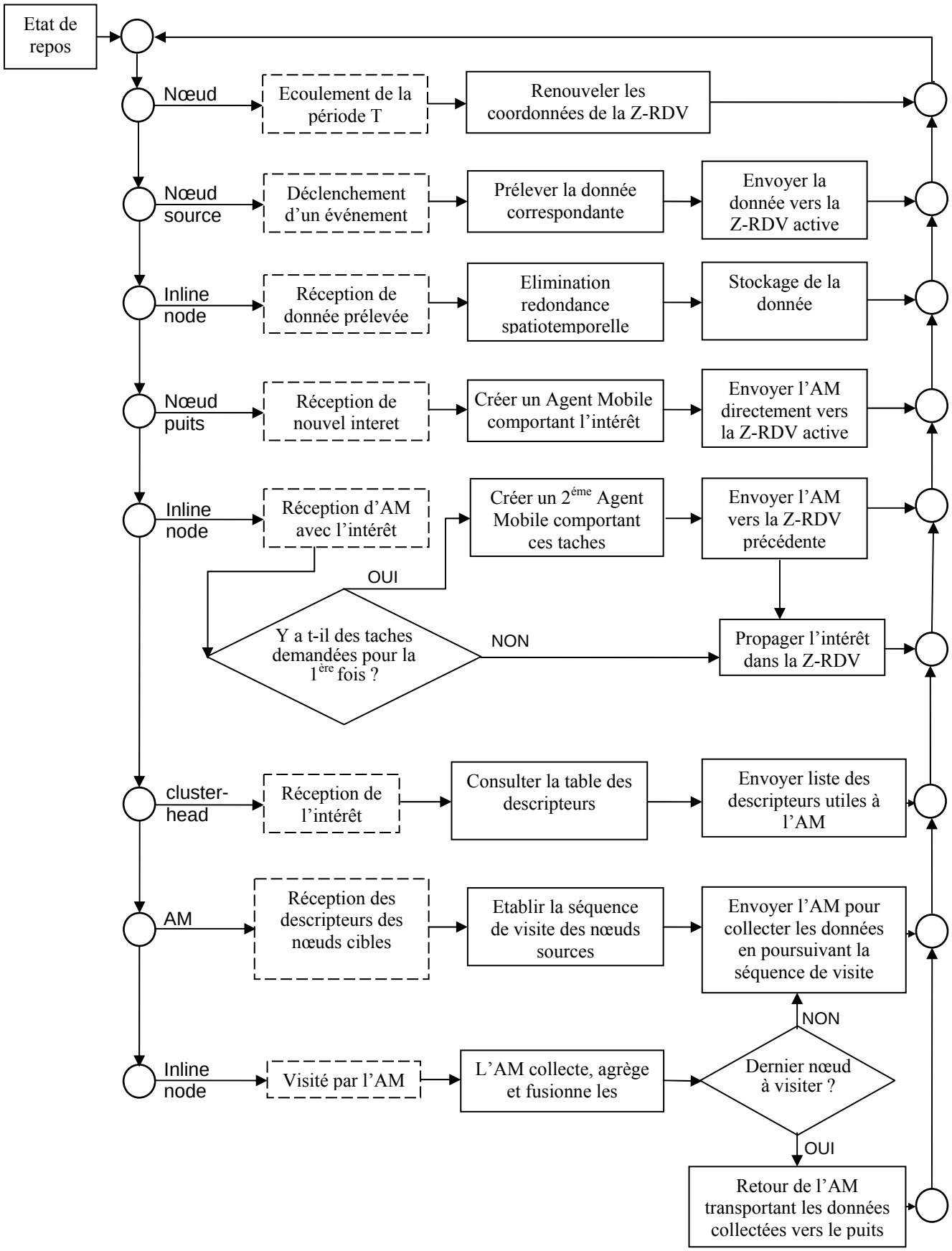

Figure IV.3 : Organigramme récapitulatif de l'algorithme de MLBDD.

IV.4.4.2.3. ***Concaténation des données :*** En se basant sur le principe que le coût de communication pour l'envoi d'un message long est généralement moins que celui d'envoi de la même quantité de données en utilisant plusieurs messages courts, nous utilisons une technique d'unification de paquets, qui concatène les données de plusieurs petits paquets en un plus grand paquet afin de réduire les coûts de communication. Grâce à la concaténation des données, le coût de communication des capteurs intermédiaires peut être réduit afin d'augmenter la durée de vie du réseau. Toutefois, ces économies d'énergie peuvent être obtenues généralement au détriment de la prolongation du temps de latence.

IV.4.4.3. Changement périodique de la zone de rendez-vous

Un nœud puits peut envoyer une requête de collecte des données à n'importe quel instant vers la zone de rendez-vous. Puisque la zone de rendez-vous est mobile et change périodiquement, le système peut tomber dans le cas où la requête de collecte est générée juste après le changement de la zone de rendez-vous. C'est-à-dire que la collecte va se faire au niveau de la nouvelle zone de rendez-vous (qui est la zone de rendez-vous actuelle), alors que les données résident toujours dans l'ancienne zone et n'ont pas encore été collectées. Pour éviter cela, le puits va ordonner la collecte des données dans les deux zones de rendez-vous : l'ancienne zone et la zone actuelle en même temps. Ceci va se faire uniquement lors de la première collecte de chaque type de données pour chaque période de changement de zone de rendez-vous. C'est-à-dire qu'à chaque changement périodique de zone de rendez-vous, la collecte de chaque type de données se fera au niveau des deux zones (actuelle et ancienne) ; mais juste après cette première collecte, les collectes suivantes de ce type de données vont se faire normalement dans la zone actuelle seulement (voir figure IV.4).

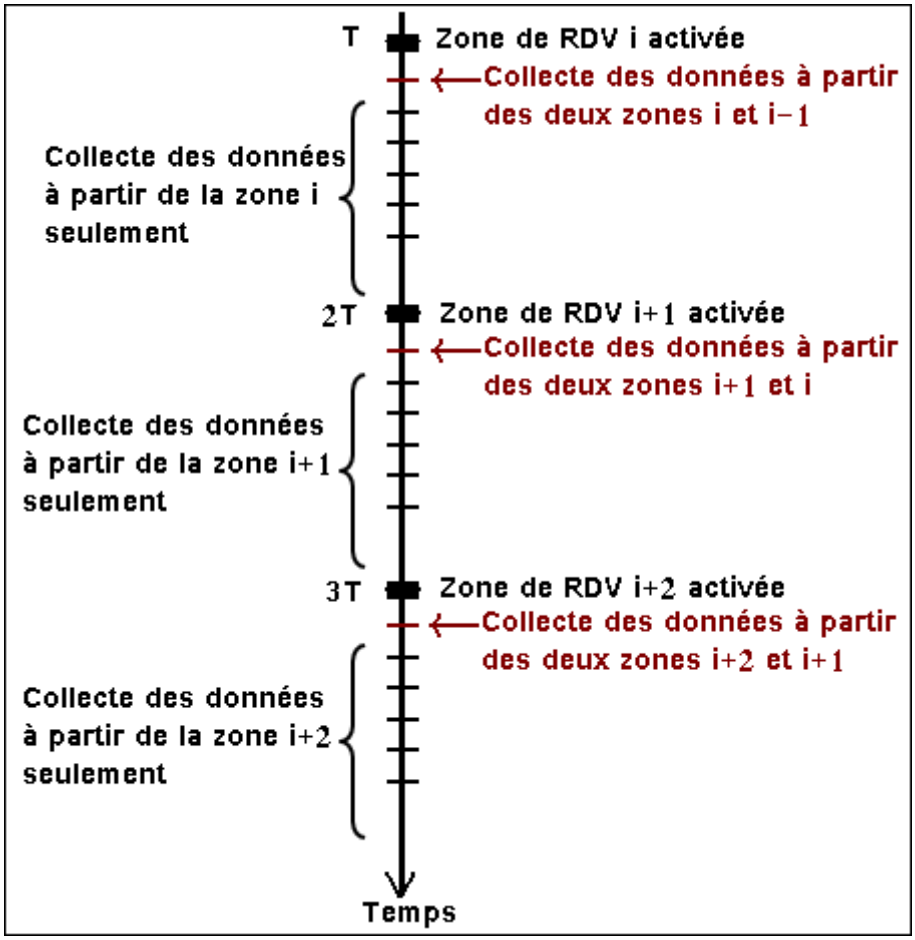

Figure IV.4 : Collecte des données adaptée au changement de la zone de RDV.

IV.4.4.4. Mobilité du puits

Le but de cette étape est de faire parvenir la donnée au puits, même si ce dernier se déplace durant l'exécution d'une tâche donnée.

Chaque puits est associé à un *home-node*. Initialement, le puits choisit un voisin comme home-node, et transmet les coordonnées du home-node dans ses requêtes. Le home-node va se charger d'émettre les requêtes vers la bande et de retransmettre les données reçues vers le nœud puits. Cette élection va se faire avant le lancement de chaque tâche. Lorsque le puits se déplace hors portée de son home-node, il transmet ses coordonnées au home-node en utilisant toujours un routage géographique.

Cette méthode offre un mécanisme de handoff afin d'éliminer la perte de données lors du déplacement du puits.

IV.5. Conclusion

Dans ce chapitre, nous avons proposé un protocole de dissémination des données dans les réseaux de capteurs sans fil à base de zone de rendez-vous et d'agent mobile. Ce protocole a été appelé MLBDD (Mobile Line Based Data Dissemination). MLBDD est un protocole hybride inspiré des approches proposées par les travaux de LBDD (Line Based Data Dissemination) [Ben-07] et MAWSN (Mobile Agent Based Wireless Sensor Network) [Che-06]. Notre protocole propose une amélioration qui tente d'apporter des solutions aux principales contraintes des travaux précédents. La contrainte énergétique qui limite la durée de vie du réseau est sans doute la plus importante. L'idée motrice a été de permettre une définition de zone de rendez-vous qui change dynamiquement dans le temps de sorte à équilibrer la consommation d'énergie fournie par chaque capteur pour l'effort de dissémination des données sur le champ de captage. MLBDD a pour objectif de proposer un protocole efficace et à moindre coût pour la mobilité de la zone de rendez-vous.

Afin de confirmer l'efficacité de MLBDD, nous avons effectué une évaluation des performances et des comparaisons.

Chapitre V

Evaluation des performances

Chapitre V. Evaluation des performances

V.1. Introduction

Pour évaluer les performances de la solution MLBDD conçue dans le chapitre précédent, et la comparer avec d'autres travaux similaires, nous avons effectué des simulations.

L'objectif de la simulation est de modéliser le monde réel et de mettre en évidence les caractéristiques et les interactions entre les activités du système modélisé. Elle permet aussi d'observer le comportement du système et de suivre son évolution dans le temps.

Pour ce faire, nous avons choisi le simulateur Glomosim (*Global Mobile Information System Simulator*). Ce choix a été fait grâce à la spécialisation du simulateur aux environnements sans fil, et aussi à la diversité des fonctionnalités qu'il offre pour la gestion du système modélisé et des résultats de simulation.

V.2. Glomosim

GloMoSim est un simulateur réseau modulaire pour la simulation et l'évaluation des performances des réseaux sans fil (locaux ou étendus) [Bag-98]. Il utilise la simulation parallèle à événements discrets fournie par le langage PARSEC (*Parallel Simulation Environment for Complex System*) développé par PCL (*Parallel Computing Laboratory*) de l'université UCLA (*University of California at Los Angles*). Le noyau de la simulation PRSEC permet à GloMoSim d'avoir une grande vitesse d'exécution et un bon passage à l'échelle en exécutant des simulations de centaines de milliers de nœuds mobiles.

V.2.1. Architecture

La plupart des systèmes réseaux sont construits en utilisant une approche basée sur les couches qui est semblable à l'architecture à sept couches de la norme OSI.

Des APIs (*Application Programming Interface*) standards sont utilisées entre les différentes couches du simulateur. Ceci autorisera l'intégration rapide et facile des modèles et protocoles développés aux différentes couches par différents développeurs.

Le tableau suivant montre les différentes couches du simulateur ainsi que les protocoles disponibles dans chaque couche.

Couche Application (CBR, HTTP, Telnet, FTP)
Couche Transport (TCP, UDP)
Couche Réseau (IP avec AODV, Flooding, Bellman-Ford, OSPF, DSR, WRP)
Couche Mac (CSMA, MACA, MACAW, FAMA, 802.11)
Couche Radio (Free Space, Rayleigh, Ricean, SIRCIM)
Couche Mobilité (Random drunken et Random waypoint)

Figure V.1 : Architecture en couches de Glomosim.

Le transfert de messages dans GloMoSim d'un nœud I vers un nœud J est montré dans la figure ci-dessous.

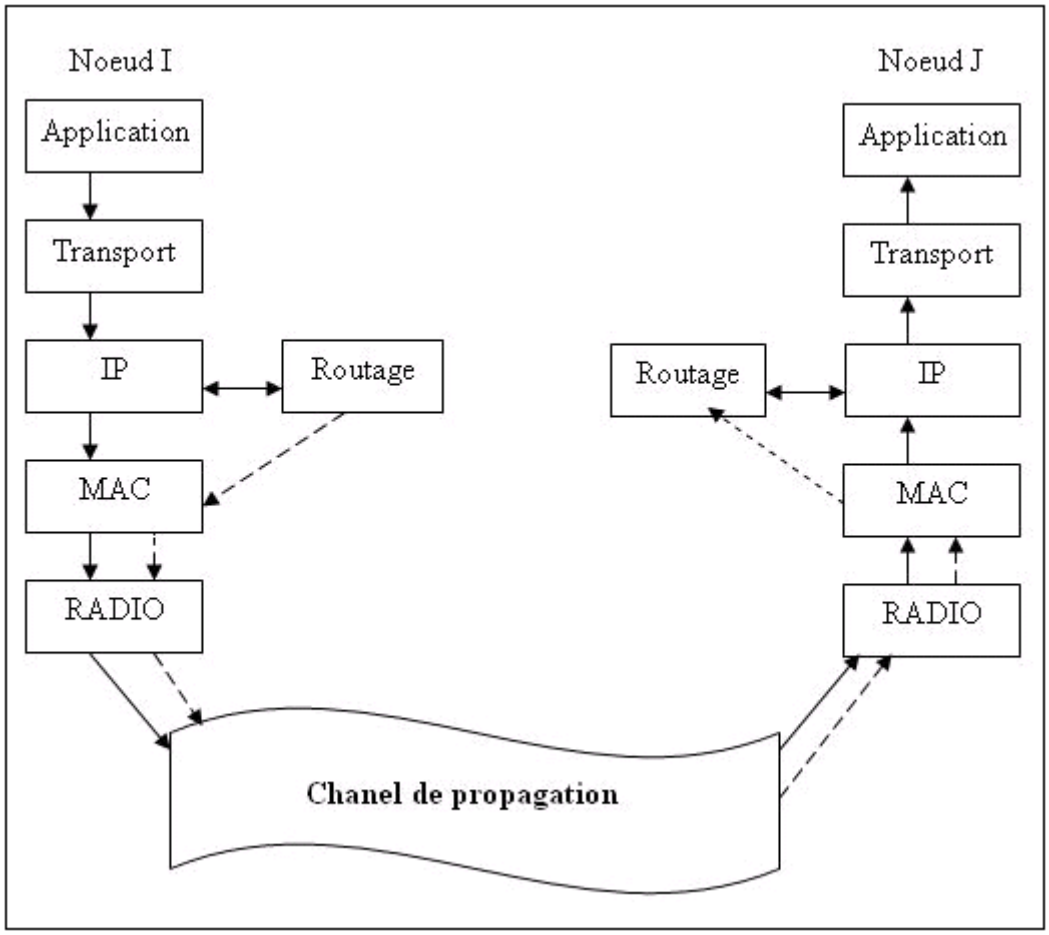

Figure V.2 : Transfert des paquets dans Glomosim.

V.2.2. Paramètres de configuration du simulateur

Avant de commencer la simulation, il faut d'abord préciser les paramètres de l'environnement simulé. En effet, des paramètres de configuration sont injectés au modèle de simulation à partir d'un fichier d'entrée nommé ***config.in*.** Ce fichier contient un ensemble de variables de paramètres de simulation avec les valeurs correspondantes. Parmi ces variables on trouve :

- SIMULATION-TIME : Le temps de simulation. Il est exprimé en jours, heures, minutes, secondes, microsecondes et nano-secondes. Si on ne précise pas l'unité, la seconde est utilisée par défaut.
- TERRAIN-DIMENSIONS (X , Y) : Dimensions du terrain de simulation en mètres. X et Y représentent respectivement la largeur et la longueur du terrain**.**
- NUMBER-OF-NODES : nombre de nœuds dans le réseau à simuler.
- NODE-PLACEMENT et NODE-PLACEMENT-FILE : décrivent les emplacements des nœuds dans le terrain : (aléatoire, uniforme, rangés, en groupe, etc.).
- MOBILITY : spécifie si les nœuds seront mobiles ou pas.
- RADIO-BANDWIDTH : Largeur de la bande passante utilisée par les nœuds pour transmettre des messages.
- POWER-RANGE : représente la portée de communication de chaque nœud. Un nœud peut échanger des messages directs avec les nœuds se trouvant dans sa portée de communication (échange de messages à un saut).
- MAC-PROTOCOL : Protocole de la couche Mac (802.11, CSMA, FAMA (*Floor Acquisition Multiple Acces*), MACA (*Multiple Access Collision Avoidance*)).
- ROUTING-PROTOCOL : protocole utilisé pour le routage des paquets. On trouve : BELLMANFORD, AODV, DSR, LAR1, WRP, ZRP, FISHEYE ou STATIC.
- APPLICATION : FTP, CBR, HTTP ou TELNET.
- TRANSPORT-PROTOCOL : UDP ou TCP.
- PROPAGATION-PATHLOSS : modèle de propagation: FREE-SPACE, TWO-RAY, PATHLOSS-MATRIX.

Il existe aussi d'autres paramètres comme la température, le bruit et les paramètres de statistiques.

V.3. Configuration de notre modèle de simulation

En ce qui concerne notre modèle de simulation, la configuration des paramètres cités ci-dessus est comme suit:

V.3.1. Modèle de propagation

Le signal se propage de l'émetteur vers le récepteur suivant le modèle TWO-RAY qui prend en considération les obstacles entre les nœuds, où le signal ne faiblit pas seulement en fonction de la distance séparant le nœud source et le nœud destination.

V.3.2. Protocole de la couche MAC

Le protocole choisit est le protocole 802.11, ce protocole est généralement utilisé dans la simulation des réseaux sans fil. Dans ce protocole, chaque paquet transmis doit être suivi par un acquittement de réception. L'absence de cet acquittement après plusieurs transmissions indique la défaillance du lien entre le nœud émetteur et le nœud récepteur.

V.3.3. Couche application

Dans cette couche, chaque nœud peut générer des données prélevées par les capteurs, et chaque puits peut générer des demandes de données. Pour ce faire, Nous avons implémenté les applications suivantes :

- **Donnée** <Idf_nœud > <Taille_donnée> <Date_création> <Type_donnée> : Cette application crée une nouvelle donnée de type « Type_donnée » au moment indiqué par « Date_création » d'une taille précisée par « Taille_donnée » sur le nœud identifié par « Idf_nœud ».

- **Demande** <Idf_puits > <Types_données> <Date_dem> : Cette application représente une requête du puis identifie par <Idf_puits > demandant plusieurs données de types désignés par « Types_données » à l'instant « Date_dem ».

Puisque nous utilisons un simulateur, les positions des nœuds sont donc disponibles et c'est sur celles-ci que nous nous sommes basés pour implémenter le protocole de routage géographique (lorsqu'un nœud a un paquet à transmettre, le nœud de relais le plus proche de la destination est choisi pour transférer le paquet).

V.3.4. Autres paramètres

Dimensions du terrain	200m x 200m
Nombre de nœuds	400
Durée de simulation	15 minutes
Portée de communication de chaque nœud	37.67 mètres
Emplacement des nœuds (les nœuds sont immobiles durant toute la simulation)	aléatoire
Période de changement de zone de rendez-vous	3 minutes
Période de création des données	1 seconde
Période de création des demandes par le puits	1 minute
Taux d'élimination de la redondance d'application par l'AM	10%
Facteur de fusion de l'AM	1

V.4. Paramètres d'étude

Comme dit précédemment, le simulateur Glomosim permet de simuler des réseaux sans fil selon plusieurs paramètres. Ainsi, les caractéristiques par rapport auxquelles nous avons évalué notre système sont les suivantes :

V.4.1. La structuration en zones

Comme notre approche se base sur la division de la zone d'intérêt en plusieurs zones rectangulaires, ce paramètre permet d'étudier et de comparer le comportement des nœuds dans chaque zone.

Puisque la largeur de la bande rectangulaire est de 37.67m (équivalente à la portée des nœuds, voir paragraphe IV.4.1), et la largeur du terrain est de 200m, alors nous obtiendrons au total 5 zones rectangulaires dans toute la zone d'intérêt.

V.4.2. La charge du réseau

La charge du réseau est indiquée par le nombre des données et le nombre des demandes générées par unité de temps. Ce paramètre est en relation directe avec la consommation de la bande passante et l'augmentation du trafic. En effet, une grande charge dans le réseau

conduit à un grand risque de collisions et peut être une dégradation du système. C'est donc important d'étudier le comportement de notre solution par rapport à ce paramètre. Nous ferons varier la charge de 0,5 jusqu'à 4 données/sec et de 0,5 jusqu'à 4 demandes/min.

V.4.3. Passage à l'échelle

Ce paramètre est important dans les réseaux de capteurs, car ils sont généralement denses. Nous allons étudier notre solution en faisant varier le nombre des nœuds capteurs entre 200 et 600 nœuds.

V.5. Métriques à mesurer

En se basant sur les paramètres mentionnés précédemment, nous allons mesurer les métriques importantes pour évaluer les performances de notre approche:

V.5.1. L'énergie

Ce paramètre est le plus important dans les réseaux de capteurs sans fil. En effet, les capteurs sont munis d'une faible source d'énergie généralement non rechargeable. Pour cela, la solution proposée doit être efficace en énergie afin de permettre une plus longue durée de vie du réseau de capteurs.

Comme nous faisons des comparaisons avec LBDD [Ben-07], et puisque tous deux MLBDD et LBDD utilisent un système de localisation (GPS), alors nous n'avons pas inclus le coût énergétique pour la communication avec le GPS.

V.5.2. Le délai

Ce paramètre représente le temps écoulé depuis le lancement d'une demande d'intérêt par le puits jusqu'à la réception des données demandées. Ce paramètre pourrait être très important pour certaines applications critiques comme la détection d'incendies ou de catastrophes naturelles …etc.

V.6. Résultats de la simulation

Afin d'évaluer les performances de notre solution (MLBDD), nous avons implémenté les deux approches MLBDD et LBDD avec le même simulateur (glomosim) et nous avons effectues plusieurs exécutions afin de comparer leurs résultats.

V.6.1. L'énergie

V.6.1.1. L'énergie en fonction de la structuration en zones

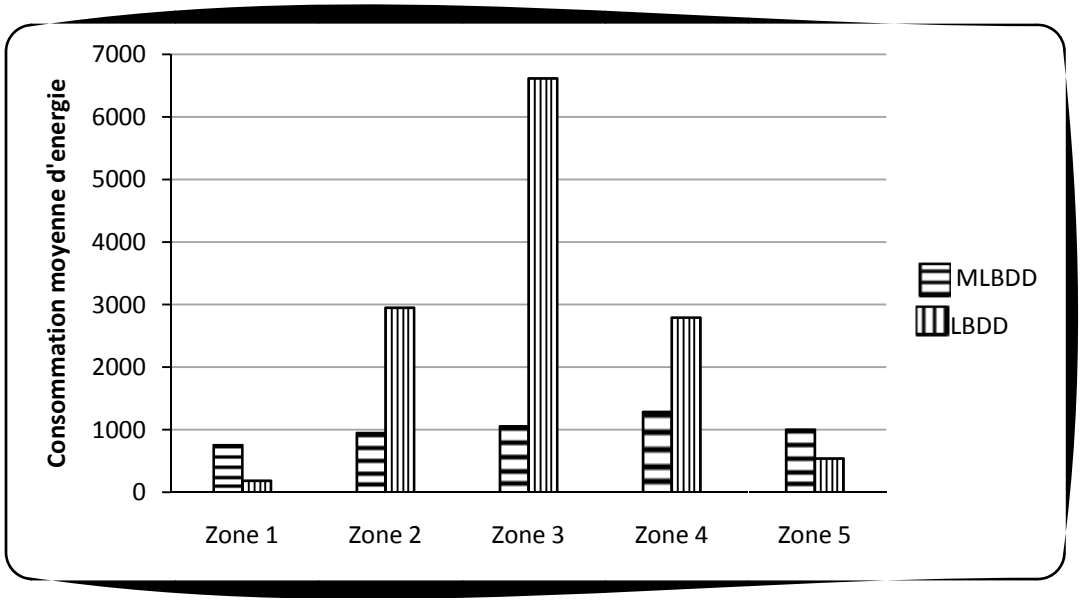

Figure V.3 : Energie / Structuration en zones.

L'histogramme ci-dessus nous montre la consommation d'énergie dans chaque bande de la zone d'intérêt. En effet, LBDD concentre la charge dans la zone du milieu puisqu'il emploi une zone de rendez-vous statique, alors que MLBDD répartie la charge entre toutes les zones. Ceci va influer directement sur la durée de vie du réseau. En effet, la durée de vie du réseau dans LBDD dépend directement de la durée de vie des nœuds placés au milieu de la zone d'intérêt alors que dans MLBDD la durée de vie ne dépend pas spécialement de certains nœuds du réseau.

V.6.1.2. L'énergie en fonction de la charge des données

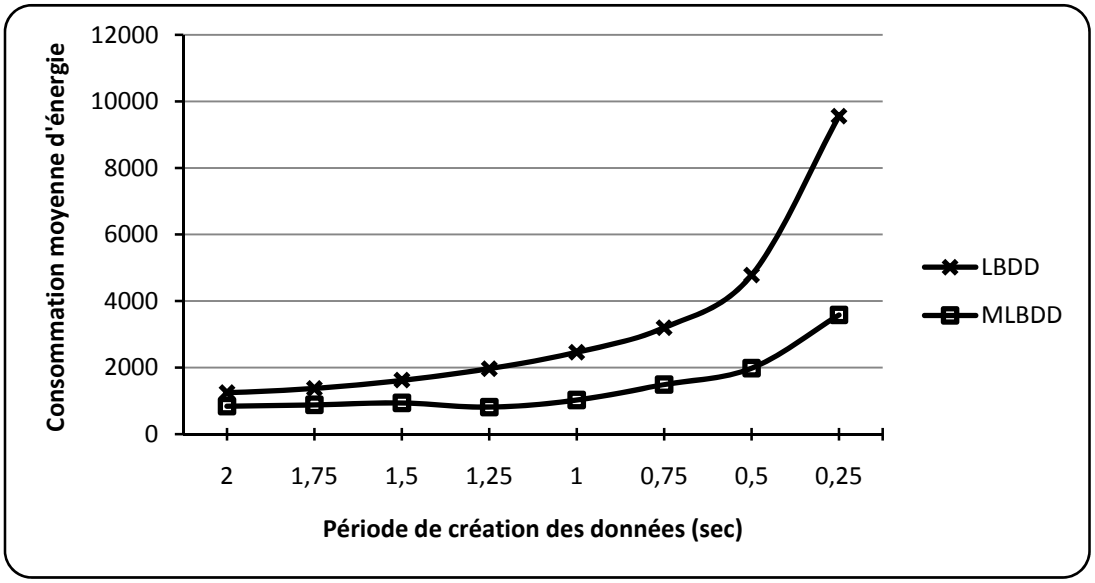

Figure V.4 : Energie / Charge des données.

D'après le graphe ci-dessus, nous constatons que la consommation d'énergie augmente lorsque nous augmentons la fréquence de création des données. Ceci est dû à l'augmentation du trafic engendré par la dissémination et le stockage des données. Cependant, la consommation de l'énergie de LBDD est toujours supérieure à celle de MLBDD, ceci est dû au fait que LBDD stocke les données dans tout le groupe au niveau de la zone de rendez-vous alors que MLBDD stocke chaque donnée chez un seul nœud appartenant à la zone de rendez-vous en plus de l'agrégation des données assistée par AM dans MLBDD qui diminue la taille des paquet à transmettre au puits. De plus, la différence de consommation d'énergie augmente avec l'augmentation de la charge des données. En effet, le taux d'amélioration de la consommation d'énergie dans MLBDD atteint 133% par rapport à LBDD dans le cas ou la période de génération des données est de 0,25 secondes. Cela favorise énormément MLBDD dans des réseaux avec une charge de données élevée.

V.6.1.3. L'énergie en fonction de la charge des demandes

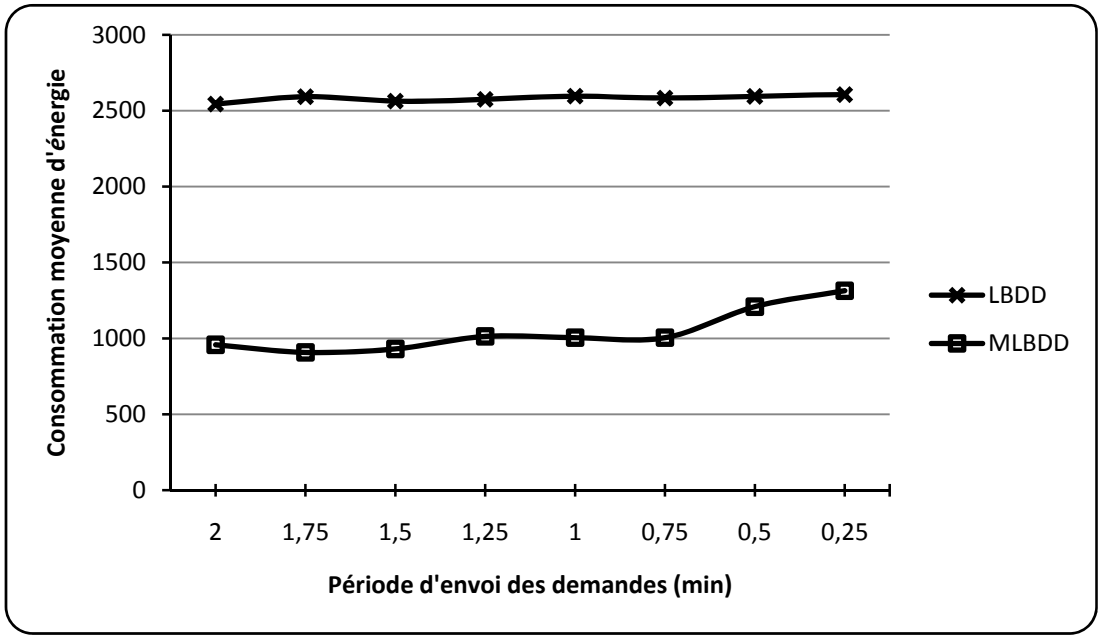

Figure V.5 : Energie / Charge des demandes

Le graphe ci-dessus nous montre que la consommation d'énergie de LBDD reste presque stationnaire lorsque nous augmentons la fréquence des demandes de collecte des données, alors qu'elle augmente concernant MLBDD. L'augmentation de la consommation de l'énergie dans MLBDD est due à l'envoi de l'AM pour chaque demande d'intérêt du puits ; donc plus la fréquence des demandes d'intérêt augmente, plus on envoi des AMs transportant le code d'agrégation et plus on engendre un trafic supplémentaire (trafic de l'AM). Il est donc préférable que les périodes de demande d'intérêt soient éloignées dans MLBDD pour que l'AM soit plus efficace en énergie.

Cependant, la consommation d'énergie de MLBDD reste toujours inferieure à celle de LBDD, ceci est dû à la phase de stockage des données (voir paragraphe V.6.1.1).

V.6.1.4. L'énergie en fonction du passage à l'échelle

Dans la figure V.6 nous remarquons pour les deux approches LBDD et MLBDD, que la consommation d'énergie augmente lorsque le nombre des nœuds augmente dans le réseau, car plus le nombre des nœuds augmente et plus le trafic augmente. Cependant, la consommation d'énergie dans LBDD est toujours supérieure à celle de MLBDD. Ceci est dû au fait que LBDD stocke chaque donnée dans tout le groupe (plus le nombre de nœuds augmente plus la taille des groupes augmente) de plus que MLBDD utilise l'AM pour

agréger les données collectées ce qui diminue les couts de collecte et de stockage dans MLBDD par rapport à LBDD.

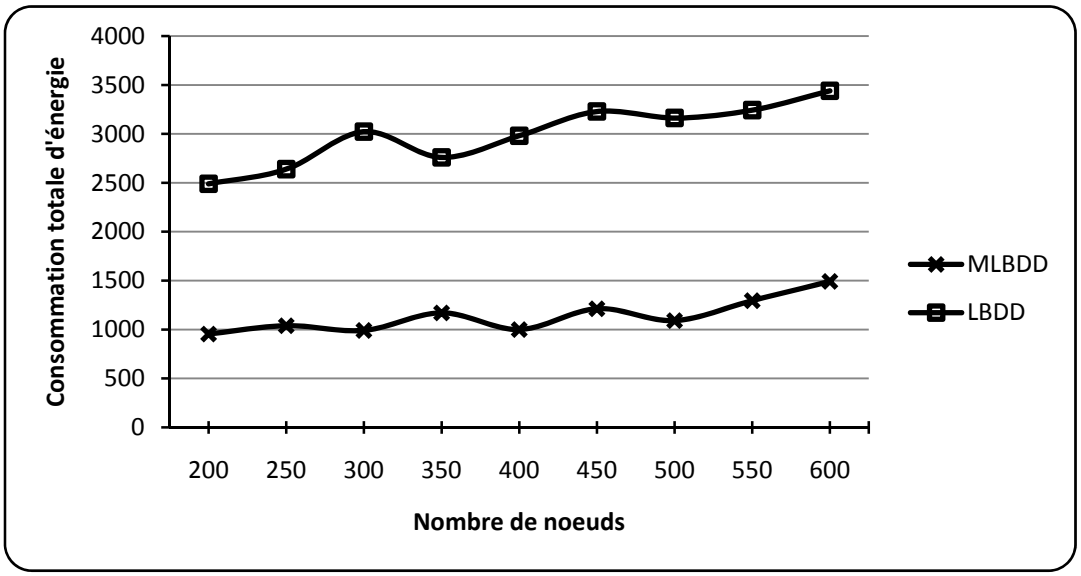

Figure V.6 : Energie / Passage à l'échelle

V.6.2. Le délai

V.6.2.1. Le délai en fonction de la charge des données

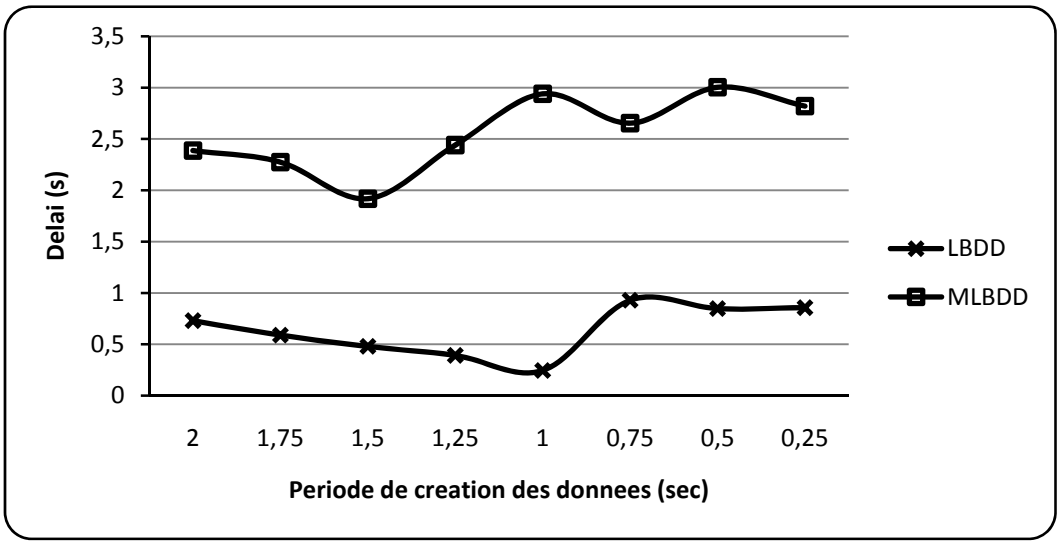

Figure V.7 : Délai / Charge des données.

Le graphe de MLBDD nous montre que plus on des données dans le réseau, plus le délai de collecte augmente. En effet il varie entre 1.9 et 3 secondes. Ceci est dû à l'utilisation de l'AM qui parcourt tous les nœuds cibles pour collecter les données concernées. Cependant,

LBDD a un délai (inferieur à 1 seconde) considérablement inférieur à celui de MLBDD, car lors de la réception de l'intérêt, chaque nœud cible dans LBDD envoi directement les données concernées au puits puisque celles-ci sont disponibles chez n'importe quel nœud dans les groupes.

V.6.2.2. Le délai en fonction du passage à l'échelle

D'après la figure V.8, nous constatons que plus le nombre de nœuds augmente dans le réseau, plus le délai de collecte augmente. Ceci se justifie par le fait que la communication augmente dans le réseau et donc on aura plus de collisions entre les nœuds et plus de retransmissions et donc une augmentation des délais. Le graphe montre que le délai de collecte de LBDD est meilleur que celui de MLBDD dans le cas où le nombre de nœuds est inferieur à 550, cependant la collecte dans MLBDD avec 600 nœuds prend moins de temps que dans LBDD.

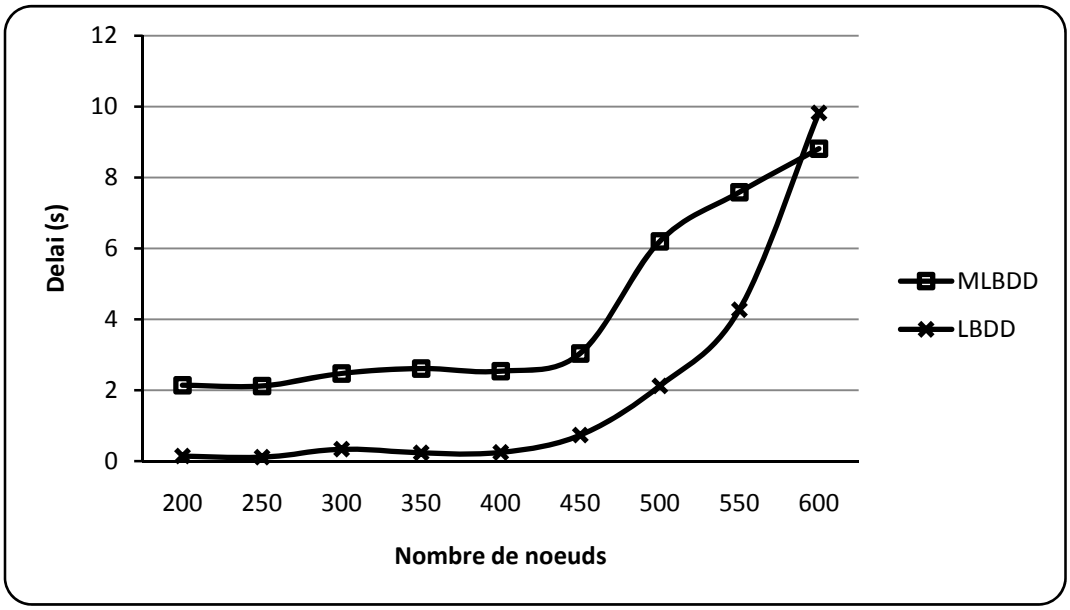

Figure V.8 : Délai / Passage à l'échelle

V.6.3. Energie X délai

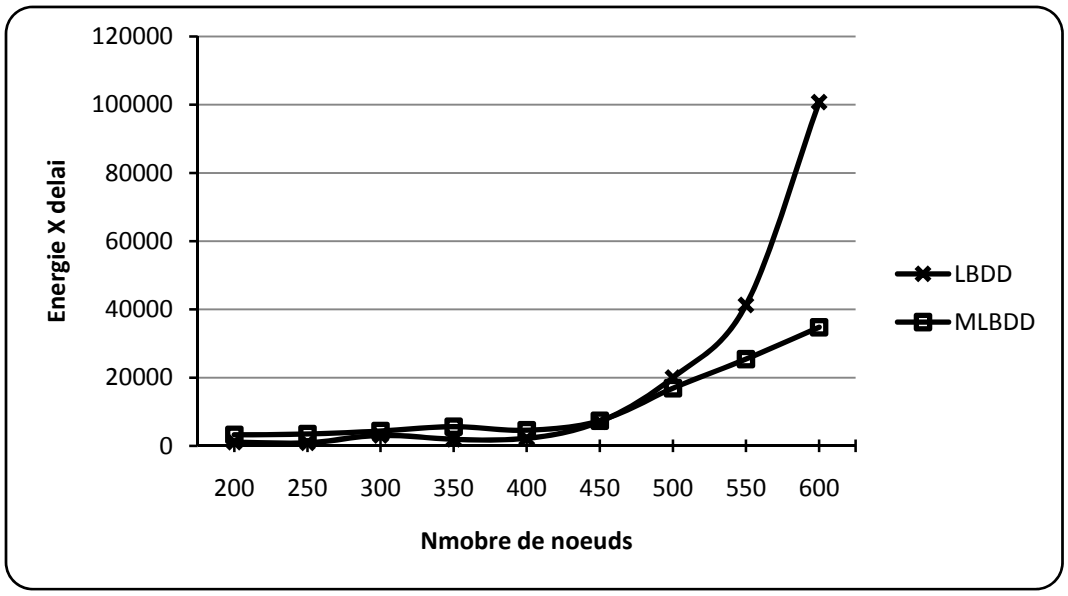

Figure V.9 : Energie X délai / Passage à l'échelle

Le graphe ci-dessus est obtenu en multipliant les valeurs des deux graphes des deux figures V.6 et V.8. On constate que lorsque le nombre de nœuds varie entre 200 et 450, les graphes de LBDD et MLBDD sont presque superposés (LBDD est légèrement meilleur que MLBDD en terme d'« énergie X délai »), alors que MLBDD est plus efficace que LBDD lorsque le nombre de nœuds dépasse les 450. De ce fait, MLBDD passe mieux à l'échelle que LBDD.

V.7. Conclusion

Dans ce chapitre, nous avons évalué la méthode proposée MLBDD en la comparant à LBDD [Ben-07]. Les résultats de simulation ont montré que MLBDD améliore largement la consommation d'énergie ainsi que la durée de vie du réseau. LBDD reste meilleur en terme de délai de collecte des données, mais dans le cas de passage à l'échelle MLBDD s'avère intéressant même en délai.

Conclusion générale

Conclusion générale

La conception d'un protocole de dissémination de données dans un RCSF se révèle un défi intéressant. En raison de leurs diverses propriétés (contraintes physiques, nature du déploiement, paradigmes de communication…), il est difficile de trouver un protocole idéal qui répond à toutes les exigences de ce type de réseaux.

Dans ce projet de recherche, nous avons classifié les approches de dissémination des données en deux grandes parties : celles basées sur le paradigme Client/Serveur et celles basées sur des Agents Mobiles. Les performances de ces deux paradigmes dans les réseaux de capteurs sans fil qui sont très spécifiques en contraintes matérielles diffèrent par rapports aux réseaux classiques. En effet, le paradigme Client/Serveur offre une meilleure latence alors que l'agent mobile permet de réduire considérablement la consommation énergétique.

Dans ce travail, nous avons proposé une nouvelle solution MLBDD (Mobile Line Based Data Dissemination) qui est une fusion et une amélioration de deux approches : MAWSN (Mobile Agent Based Wireless Sensor Network) [Che-06] et LBDD (Line Based Data Dissemination) [Ben-07].

Comme le facteur d'énergie constitue l'une des contraintes les plus importantes qui guident la conception des protocoles dans les réseaux de capteurs sans fil, ils doivent donc intégrer des mécanismes qui permettent aux utilisateurs de prolonger la durée de vie du réseau en entier. Pour cette raison, nous avons conçu l'approche MLBDD (*Mobile Line Based Data Dissemination*) qui a pour principal objectif la prolongation de la durée de vie du réseau, en utilisant une approche de dissémination des données équitable en énergie qui limite le coût en énergie et en communication tout en assurant un équilibre de la consommation d'énergie des nœuds dans tout le réseau. Aussi, MLBDD utilise un agent mobile [Che-06] pour collecter les données dans la zone de rendez-vous en employant des fonctions d'agrégation afin de réduire la quantité des données transmises et d'économiser ainsi l'énergie de communication et la bande passante utilisée.

L'évaluation des performances a été faite en comparant MLBDD à LBDD par des simulations en utilisant le simulateur Glomosim spécifique aux réseaux sans fil. Les résultats de simulation ont montré que MLBDD améliore largement la consommation énergétique et équilibre la charge entre tous les nœuds du réseau. Toutefois, LBDD reste meilleur en ce qui

concerne le délai de collecte des données. De ce fait, MLBDD est convenable pour les applications exigeant une longue durée de vie où le champ de déploiement est inaccessible comme les applications de surveillance de l'environnement. MLBDD est aussi meilleur pour les réseaux a grande échelle et ayant de grandes charges de données.

Comme perspectives, le rajout d'un module pour la tolérance aux pannes serait intéressant notamment concernant la transmission du paquet de l'agent mobile. Ce serait aussi meilleur de pouvoir mettre en place notre protocole en le déployant sur un vrai réseau de capteurs.

Bibliographie

Références bibliographiques

[Aie-08] : F. Aiello, G. Fortino et A. Guerrieri. « Using Mobile Agents as Enabling Technology for Wireless Sensor Networks ». The Second International Conference on Sensor Technologies and Applications, 2008.

[Aky-02] : I.F. Akyildiz, W. Su, Y.Sankarasubramaniam et E. Cayirci. « Wireless sensor networks: a survey ». Computer Networks: The International Journal of Computer and Telecommunications Networking, v.38 n.4, pp.393-422, 2002.

[Bac-08] : J. Bacon, A. R. Beresford, D. Evans, D. Ingram, N. Trigoni, A. Guitton, et A. Skordylis. « TIME: An open platform for capturing, processing and delivering transport-related data ». In Proceedings of the 5th IEEE Consumer Communications and Networking Conference (CCNC), pp.687-691, 2008.

[Ben-07] : E. Ben Hamida, A. Ziviani et M. Dias de Amorim. « Dissémination dans les réseaux de capteurs avec puits mobiles ». Dans 9ème Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications pp.85-88 (INRIA-00176965, version 1), 2007.

[Ber] : Site Internet de l'université de Berkeley. <http:// berkeley.edu/ >.

[Bok-04] : T. Bokareva, N. Bulusu et S. Jha. « A performance comparison of data dissemination protocols for wireless sensor networks ». Proceedings of Globecom 2004, IEEE Press, pp.85-89, 2004.

[Bus-06] : Y. Busnel. « Systèmes d'information pair-à-pair pour les réseaux de capteurs larges échelles ». Manuscrit auteur, publié dans "Lettre de la fondation Metivier 1, 2 : 3--5 ", 2006.

[Car-05] : J. Cartigny, F. Ingelrest, D. Simplot-Ryl et I. Stojmenovic. « Localized LMST and RNG based minimum energy broadcast protocols in ad hoc networks ». Ad Hoc Networks, 3(1):1–16, Janvier 2005.

[Cas-08] : C. Castelluccia et A. Francillion. « Protéger les réseaux de capteurs sans fil ». SSTIC2008, Renne, France, Juin 2008.

[Che-04] : W-P. Chen. « A data-quality driven framework for data dissemination in wireless sensor networks ». Thèse de doctorat, Université d'Illinois à Urbana-Champaign, 2004.

[Che-06] : M. Chen, T. Kwon, Y. Yuan et V.C.M. Leung. « Mobile Agent Based Wireless Sensor Networks ». Journal of computers, vol. 1, NO. 1, Avril 2006.

[Che-07] : M. Chen, T. Kwon, Y. Yuan, Y. Choi et V.C.M. Leung. « Mobile agent-based directed diffusion in wireless sensor networks ». EURASIP Journal on Advances in Signal Processing, Article ID 36871, 13 pages, 2007.

[Chi-05] : M. Chilowicz et I.El Otmani.« Regroupement économe en énergie par vote de capteurs ». exposé de Master2,
< http://monge.univ-mlv.fr/%7Echilowi/misc/m2/adhoc.pdf >. 2005.

[Chl-05] : I. Chlamtac, I. Carreras et H. Woesner. « From internets to bionets: biological kinetic service oriented networks ». The case study of Bionetic Sensor Networks. CREATE-NET Research Consortium, Trento, Italy, 2005.

[Cho-03] : J. Chou, D. Petrovic et K. Ramchandran. « A Distributed and Adaptive Signal Processing Approach to Reducing Energy Consumption in Sensor Networks ». IEEE INFOCOM 2003.

[Cro] : http:// www.xbow.com/ site Internet de Crossbow Technology, Inc.

[Erm-04] : E. Ermel. « Localisation et Routage géographique dans les réseaux sans fil hétérogènes ». Thèse de Doctorat de l'université Paris VI Pierre et Marie CURIE, Spécialité systèmes informatiques. Juin 2004.

[Est-02] : D. Estrin, et. al. <http://nesl.ee.ucla.edu/tutorials/mobicom02>.

[Fug-98] : A. Fuggetta, G. P. Picco et G. Vigna. « Understanding code mobility ». IEEE Trans. on Software Engineering, 24(5):342{361, 1998.

[FYe-02] : F. Ye, H. Luo, J. Cheng, S. Lu, et L. Zhang. « A two-tier data dissemination model for large-scale wireless sensor networks ». MobiCom 02: Proceedings of the 8th annual international conference on Mobile computing and networking (New York, NY, USA), ACM Press, pp.148–159, 2002.

[Hai-03] : Q. Hairong, Xu. Yingyue et W. Xiaoling. « Mobile-Agent Based Collaborative Signal and Information Processing in Sensor Networks ». In Proceeding of the IEEE, Vol. 91, NO. 8, pp.1172-1183, Août 2003.

[Hal-01] : D.L. Hall et J. Llinas. « Handbook of Multisensor Data Fusion ». CRC Press, 2001.

[Hei-03] : J. Heidemann, F. Silva et D. Estrin, « Matching Data Dissemination Algorithms to Application Requirements », The First ACM Conference on Embedded Networked Sensor Systems (Sensys'03), Los Angeles, CA, USA, Novembre 2003.

[Hof-97] : W. Hofmann, H. Lichtenegger et J. Collins. « Global Positioning System: Theory and Practice ». Springer; 4th Rev edition (May 1997).

[Int-00] : C. Intanagonwiwat, R. Govindan, et D. Estrin. « Directed diffusion: a scalable and robust communication paradigm for sensor networks ». In Proceedings of the 6th Annual ACM/IEEE (MOBICOM ’00), pp.56-67, USA, Août 2000.

[JLu-07] : J. Lu et F. Valois. « On the Data Dissemination in WSNs ». In the Third IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2007) 0-7695-2889-9/07, 2007.

[Khe-04] : L. Khelladi et N. Badache. « Les réseaux de capteurs : état de l'art ». Rapport de recherche N° LSI-TR0304, USTHB, Alger, Février 2004.

[Khe-05] : I. Khemapech, I. Duncan et A. Miller. « A survey of wireless sensor networks technology ». In PGNET, Proceedings of the 6th Annual PostGraduate Symposium on the Convergence of Telecommunications, Networking & Broadcasting, Juin 2005.

[Lan-99] : D.B. Lange et M. Oshima, « Seven Good Reasons for Mobile Agents ». Communications of the ACM, 42(3), 1999.

[LeC-08] : V. Le Cam et al. « Applications des réseaux de capteurs intelligents et de la communication sans fil à l'instrumentation des structures de génie civil ». Bulletin des laboratoires des ponts et chaussées, ISSN 1269-1496, Nº. 273, pp.9-37, 2008.

[Lev-02] : P. Levis et D. Culler, « Maté: A Tiny Virtual Machine for Sensor Networks » Proc. of the 10th Int'l Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS X). San Jose (CA), USA, Oct 2002.

[Liu-04] : J. Liu, et X. Jin. « Agent-based, energy efficient routing in sensor networks». In Proceedings of the third international joint conference on autonomous agents and multiagent systems (AAMAS' 04) (Vol. 1, pp.472-479). 2004.

[LLi-01] : L. Li et J. Y.Halpern. « Minimum-energy mobile wireless networks revisited ». IEEE International Conference on Communications ICC '01, Helsinki, Finland, 2001.

[Mad-02] : S. Madden, M. J. Franklin, J. M. Hellerstein, et W. Hong. « TAG: A Tiny Aggregation service for ad-hoc sensor networks ». In Proceedings of the 5th Symposium on Operating System Design and Implementation (OSDI 2002), Boston, Massachusetts, Decembre 2002.

[Med-09] : S. Medjiah, T. Ahmed, F. Krief, P. Gélard. « AGEM : Un Protocole de Routage Géographique Angulaire Adaptatif ». Colloque francophone sur l'ingénierie des protocoles, Strasbourg, France, CFIP'2009.

[Mou-07] : S. Moussaoui. « Contribution à l'amélioration de l'accessibilité dans les réseau mobile ad hoc », Thèse d'Etat en Informatique, USTHB 2007.

[Nat-08] : M. Nati. « Geographic routing: Mission possible ». Thèse Ph.D, Sapienza, Université de Rome. Italie, Mars 2008.

[Nic-05] : D. Niculescu. « Communication paradigms for sensor networks ». Communications Magazine, IEEE 43, no. 3, pp.116-122, 2005.

[QWu-04] : Q. Wu, N.S.V. Rao, J. Barhen, et al., « On computing mobile agent routes for data fusion in distributed sensor networks ». IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 6, pp.740-753, 2004.

[Ros-98] : K.N. Ross et R.D. Chaney. « Mobile Agents in Adaptive Hierarchical Bayesian Networks for Global Awareness ». Proc. IEEE Int'l Conf. Systems, Man, and Cybernetics, pp.2207-2212, 1998.

[Sch-01] : C. Schurgers et M. Srivastava. « Energy efficient routing in wireless sensor networks». In MILCOM proceedings on communications for network-centric operations: creating the information force, pp. 357–361. Vienna, VA, 2001.

[Sen-06] : S. Sentilles. « Architecture logicielle pour capteurs sans fil en réseau ». rapport de stage master recherche : Technologies de l'Internet, Université de Pau et des Pays de l'Adour, Juin 2006.

[Sha-02] : R.C. Shah, et J.M. Rabaey. « Energy aware routing for low energy ad hoc sensor networks ». In Proceedings of the IEEE wireless communications and networking conference (WCNC '02). 2002.

[Sha-08] : E. Shakshuki, H. Malik et M.K. Denko. « Software agent-based directed diffusion in wireless sensor network ». Telecommunication Systems 38(3-4): pp.161-174. 2008.

[Shi-05] : J.H. Shin, J. Kim, K. Park, et D. Park. « Railroad: virtual infrastructure for data dissemination in wireless sensor networks ». Proceedings of the 2nd ACM International Workshop (PE-WASUN'05), pp.168-174, Octobre 2005.

[Sko-06] : A. Skordylis, A. Guitton et N. Trigoni. « Correlation-Based Data Dissemination in Traffic Monitoring Sensor Networks ». ACM CoNEXT'06, 2nd Conference on Future Networking Technologies, Lisboa, Portugal. Décembre 2006.

[SNi-99] : S. Ni, Y. Tseng, Y. Chen, et J. Sheu. « The broadcast storm problem in a mobile ad hoc network ». Proc. of the International Conference on Mobile Computing and Networking (MobiCom), Seattle, USA, pp.151-162, Août 1999.

[Tri-06] : N. Trigoni, A. Guitton et A. Skordylis. « Routing and processing multiple aggregate queries in sensor networks ». SenSys pp.391-392. 2006.

[Val-05] : M. Val Machado, O. Goussevskaia, R. A. F. Mini, C. G. Rezende, A. A. F. Loureiro, G. R. Mateus, et J. M. S. Nogueira. « Data Dissemination in Autonomic Wireless Sensor Networks ». IEEE Journal on selected areas in communications, vol. 23, NO. 12, Déccembre 2005.

[Vig-04] : G. Vigna. « Mobile Agents: Ten Reasons For Failure ». Proceedings of MDM 2004, pp.298-299 Berkeley, CA Janvier 2004.

[Yao-02] : Y. Yao et J. Gehrke. « Cougar Approach to In-Network Query Processing in Sensor Networks ». SIGMOD Record, Vol. 31, No. 3, Septembre 2002.

[Yon-05] : E. Yoneki et J. Bacon. « A survey of Wireless Sensor Network technologies: research trends and middleware's role ». Technical Report UCAM-CL-TR-646, University of Cambridge, UK, Sept. 2005.

[Zha-02] : F. Zhao, J. Shin et J. Reich, « Information-Driven Dynamic Sensor Collaboration for Tracking Applications ». IEEE Signal Processing Magazine, Mars 2002.