Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire

Université Abou Bakr Belkaid– Tlemcen

Faculté desSciences

Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique

Option: Réseaux et Systèmes Distribués (R.S.D)

Thème

Une version de LEACH adaptée à un environnement non-idéal représenté par le modèle lognormal shadowing

Réalisé par :

BOUABANE Manel

Présenté le 25 Juin 2015 devant le jury composé de:

- *Mr BENMAMMAR Badr* — *Président*
- *Mr LEHSAINI Mohamed* — *Encadrant*
- *Mme TABET HELLEL Chifaa* — *Co-encadrante*
- *Mr MANA Mohammed* — *Examinateur*
- *Mr BENZIAN Yaghmoracene* — *Examinateur*

Année universitaire:2014-2015

Remerciements

Si ce document a pu voir le jour, c'est grâce aux actions conjuguées de plusieurs personnes à qui je dis sincèrement merci pour leur soutien, leur présence, leurs conseils et leur disponibilité

Je tiens à remercier tous ceux qui ont collaboré à la réalisation de ce modeste travail, fruit de longues années de labeur.

J'exprime mes remerciements et ma gratitude à mon encadreur Monsieur LEHSAINI Mohamed pour avoir accepté de m'encadrer, tout particulièrement pour l'excellence de son accompagnement ainsi que pour la confiance qu'il m'a accordé .Les mots ne sont pas suffisants pour lui exprimer toute ma reconnaissance envers lui, je lui dis simplement que les personnes avec un aussi grand cœur sont rares de nos jours et que dans mes rêves les plus fous je n'ai rêvé de tomber sur un encadreur aussi dévoué qui ne comptent ni les heures ni les encouragements ni les conseils, ainsi que ma co-encadreur M^{me} Tabet Chifaa, pour toute la disponibilité dont elle a fait preuve, ainsi qu'à la confiance qu'elle m'a attribuée tout au long de mon travail. Aussi pour ses compétences si précieuses qui ont fait avancer ce travail.

Mes sincères remerciements vont également à tous les enseignants qui nous ont formées durant ces cinq dernières années.

J'exprime ma gratitude à Mr BENMAMMAR Badr , pour l'honneur qu'il me fait de présider notre jury de soutenance je lui exprime ma gratitude profonde.

Dédicaces

SOMMAIRE

Liste des Figures

Liste des Tableaux

Introduction générale

Introduction générale

Les développements technologiques dans divers domaines liés à la micro-électronique et à la technologie des réseaux sans fil, ont donné naissance à des dispositifs miniaturisés peu coûteux, de faible puissance, multifonctionnels, et qui peuvent communiquer entre eux, appelés "nœuds capteurs". Ces derniers coopèrent entre eux pour former une infrastructure de communication appelée réseau de capteurs sans fil (RCSF).

Les nœuds capteurs peuvent interagir avec leur environnement et ils sont responsables de la collecte des données environnementales et les acheminer vers un point de collecte distant appelé "puits ou station de base" directement ou via une communication multi-sauts. Ces capteurs sont limités en termes de capacité de stockage, de calcul et surtout d'énergie puisqu'ils sont alimentés par des batteries qui sont généralement non rechargeables et non remplaçables en raison de l'endroit de déploiement de ces nœuds capteurs.

Dans certaines applications les capteurs sont déployés dans des zones hostiles dont le changement de batteries est quasiment impossible et même ils sont menacés par une destruction physique qui peut être accidentelle ou intentionnelle par des ennemis, ce qui les rend inutilisables suite à la panne reproduite.

Les RCSF sont soumis à des pannes puisqu'ils sont composés de dispositifs sans fil qui sont souvent peu fiables à cause des changements dans l'environnement (haute température, pluie, présence d'obstacles, bruit, etc..). Ces facteurs peuvent provoquer une fluctuation du signal radio et par conséquent des erreurs seront injectées dans les messages échangés. En outre, les capteurs sont alimentés par des batteries dont la durée de vie est limitée et ils peuvent être endommagés accidentellement par des animaux ou intentionnellement par des êtres humains.

Pour faire face à ces pannes, des techniques pour la fiabilité de routage dans les RCSF telles que [1-3] ont été proposées, mais pour qu'elles soient efficaces, elles doivent dépendre d'une topologie de réseau qui assure des chemins alternatifs vers la station de base et de prendre en considération la qualité de lien entre les nœuds communicants. Cela exige que le déploiement des RCSF soit planifié pour s'adapter à tout changement de topologie, de telle sorte que lorsque des pannes se produisent ou la qualité de lien se dégrade les protocoles de routage peuvent toujours continuer à assurer une livraison fiable.

Notre travail est une solution qui permet d'assurer la fiabilité de livraison dans le protocole de routage LEACH (Low-Energy Adaptive Clustering Hierarchy) [4] par l'utilisation de nœuds relais supplémentaires quand la qualité de lien descend d'un certain seuil prédéfini.

Par ailleurs la version originale de LEACH a été développée dans un modèle idéale représenté par le modèle du disque unitaire (UDG: Unit Disk Graph) [5]. Ce modèle ne prend pas en considération la fluctuation du signal radio et considère que le message arrive sans erreur au destinataire tant que la distance séparant les deux nœuds communicants est inférieure ou égale à la portée de communication. De ce fait, pour illustrer l'impact de la fluctuation du signal sur les performances de LEACH, nous avons évalué LEACH dans un environnement non-idéal représenté par le modèle LogNormal Shadowing (le modèle LNS) [6]. Puis, on a proposé une version améliorée de LEACH en impliquant la qualité de lien c'est-à-dire avant toute communication entre nœuds une évaluation préliminaire de la probabilité de réception sans erreur entre les deux nœuds communicants est faite. Si cette probabilité dépasse le seuil prédéfini les deux nœuds peuvent communiquer directement sinon un nœud relais est impliqué dans cette communication.

Ce mémoire est organisé en trois chapitres :

Dans le premier chapitre, on présente des généralités sur les réseaux de capteurs sans fil pour illustrer leurs caractéristiques et leurs spécificités.

Dans le deuxième chapitre, une présentation de la tolérance aux pannes et les causes de pannes dans les RCSF est donnée, suivie par une présentation du protocole de routage LEACH pour bien comprendre son mode de fonctionnement.

Dans le troisième chapitre, on a évalué LEACH dans le modèle LNS et on a proposé une version de LEACH qui s'adapte à un environnement non-idéal représenté par le modèle LNS.

Enfin, on conclut ce mémoire par un récapitulatif sur le travail réalisé dans le cadre de notre projet de fin d'études.

Chapitre I
Généralités sur les réseaux de capteurs sans fil

Chapitre I
Généralités sur les réseaux de capteurs sans fil

I.1 Introduction

Les RCSF sont souvent caractérisés par leur déploiement dense et à grande échelle avec une limitation de ressources telles que la capacité de stockage, de traitement et d'énergie.

Dans ce chapitre, nous donnons une brève présentation des RCSF, leurs caractéristiques et aussi la nécessité d'incorporer la tolérance aux pannes dans ce type de réseaux.

I.2 Les réseaux de capteurs sans fil

Un réseau de capteurs sans fil se composent généralement d'un grand nombre de petits dispositifs appelés "nœuds capteurs ou motes" qui sont dispersés aléatoirement ou d'une manière prédéfinie dans un champ d'intérêt appelé zone de captage en vue de collecter des données et les router directement ou via une communication multi-sauts à un nœud collecteur appelé "station de base ou puits" qui est connecté à un centre de contrôle distant via internet ou bien satellite, comme le montre la figure 1.

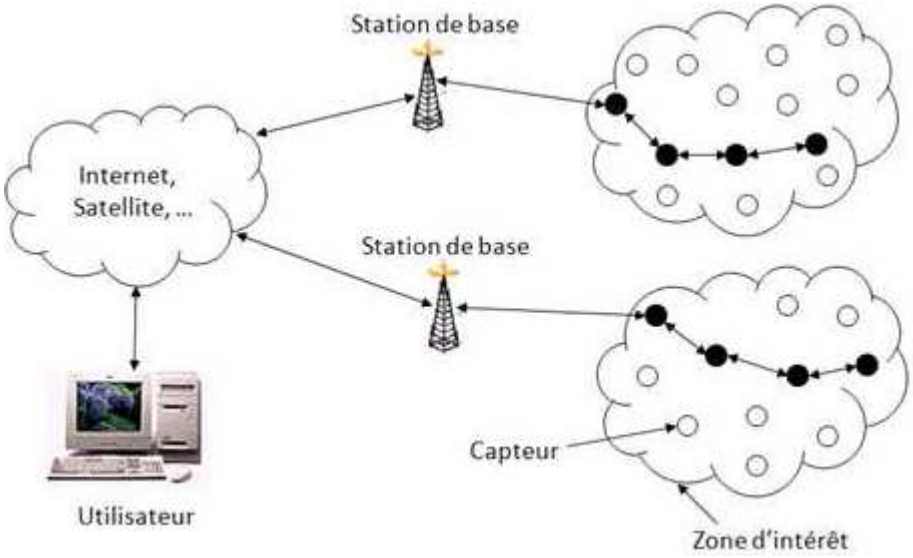

Figure 1: Exemple de RCSF

Les RCSF sont contraints en termes d'énergie, mémoire, capacité de calcul. Ils sont aussi contraints par une bande passante réduite due à la nature du canal radio partagé. Le canal de communication radio est moins fiable qu'un médium filaire. Les capteurs peuvent aussi être mobiles, ce qui nécessite des algorithmes adaptatifs au changement de la topologie réseau. Néanmoins, le vrai défi critique dans ce type de réseaux est l'énergie car les capteurs sont dotés souvent de batteries non rechargeables et irremplaçables surtout lorsque ces derniers sont déployés dans des zones d'intérêt. Ainsi, l'objectif principal dans ces réseaux est de minimiser la consommation d'énergie tout en assurant une livraison fiable de données.

La portée de transmission des nœuds capteurs dépend de cinq facteurs: la puissance d'émission, la fréquence, la modulation, la localisation et les conditions météorologiques [7]. Cette portée peut aller de quelques mètres à des centaines de mètres. Par exemple, la portée de transmission de TelosB est de 75m à 100m en outdoor et de 20m à 30m en indoor [8]. De ce fait, en fonction de déploiement des nœuds capteurs, ils ne seront pas forcément à la portée radio de la station de base. D'où, ils doivent collaborer entre eux pour envoyer les données collectées jusqu'à la station de base via un schéma de routage multi-sauts.

I.2.1 Notion de nœud capteur

Un capteur sans fil est un petit dispositif électronique capable de mesurer une valeur physique environnementale (température, lumière, pression, etc.) ou physiologique (glycémie, tension artérielle, etc.), et de la communiquer à un centre de contrôle distant via une station de base. Il est composé de quatre unités de base [9] (figure 2) :

- **L'unité d'acquisition**: est généralement composée de deux sous-unités : les capteurs et les convertisseurs analogique-numériques (ADCs). Les capteurs obtiennent des mesures numériques sur les paramètres environnementaux et les transforment en signaux analogiques. Les ADCs convertissent ces signaux analogiques en signaux numériques.
- **L'unité de traitement**: est composée de deux interfaces: une interface avec l'unité d'acquisition et une autre avec le module de transmission. Elle contrôle les procédures permettant au nœud de collaborer avec les autres nœuds pour réaliser les tâches d'acquisition, et stocke les données collectées.
- **Un module de communication (Transceiver)**: il est responsable de toutes les communications via un support de communication radio qui relie le nœud au réseau.
- **Batterie**: C'est une source d'énergie par laquelle tous les composants du capteur sont alimentés. Elle correspond généralement à une pile ou une batterie qui s'épuise par le temps.

La consommation d'énergie est devenue un facteur critique que doit prendre en compte toutes les applications de capteurs due aux limitations des ressources énergétiques des nœuds. Pour résoudre le problème d'énergie une réalisation récente d'une unité d'alimentation en utilisant des panneaux solaires [10]. Cependant, ce problème persiste puisqu'on n'a pas toujours la possibilité de s'alimenter en énergie solaire. D'où, dans presque tous les travaux de recherche on traite la problématique concernée conjointement avec la consommation d'énergie.

Il existe des capteurs qui sont dotés d'un système de localisation tel que le GPS (Global Positioning System) pour être repérer ou pour repérer les capteurs qui se trouvent dans leur voisinage et d'un mobilisateur lui permettant le déplacement tel que les drones.

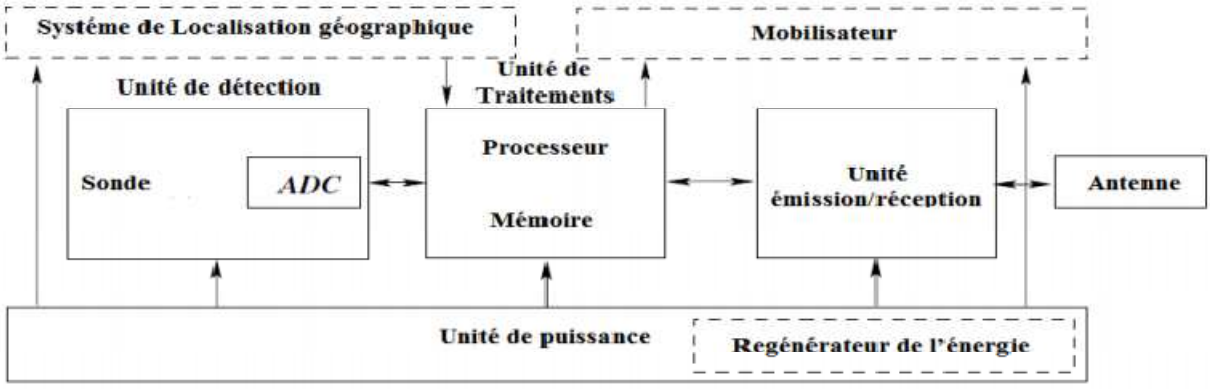

Figure 2: Architecture d'un capteur sans fil [11]

I.2.2 Catégories de communications dans les RCSF

La manière d'acheminer les messages dans les réseaux de capteurs peut changer selon l'application qui en est faite. On distingue les catégories de communication suivantes dans les RCSF :

a) Scénario périodique

Dans les scénarios impliquant la collecte de données périodiques, les capteurs transmettent régulièrement leurs mesures à la station de base. Ce processus peut être continu ou pourrait suivre une certaine distribution (géométrique, gaussienne, ...), déterministe ou probabiliste.

La connaissance du processus d'envoi peut également influencer les périodes de sommeil des capteurs non impliqués dans l'envoi de données. Ce comportement est typique pour effectuer des mesures statistiques d'une certaine métrique, par exemple l'étude du climat.

b) Selon la demande

Lorsque nous souhaitons avoir l'état de la zone de couverture à un moment T, le centre de contrôle émet des brodcasts vers toute la zone pour que les capteurs remontent leur dernier relevé vers le puits. Les informations sont alors acheminées par le biais d'une communication multi-sauts comme s'est illustré par la figure 3.

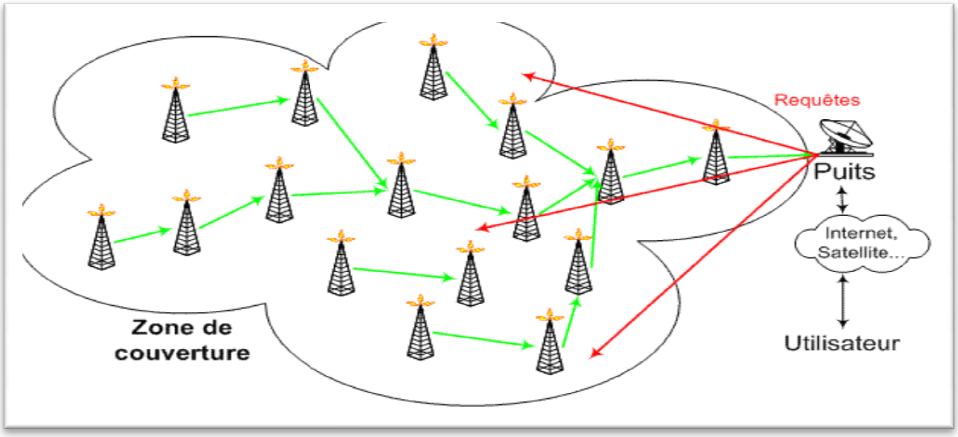

Figure 3: Collecte d'informations à la demande

c) Scénario orienté événement (event-driven)

Dans les scénarios orientés événement (event-driven), les capteurs doivent seulement transmettre une alerte quand un événement pertinent survient ou quand un changement brusque se produit. Dans ce type de scénarios, la réception de message doit être assurée et le délai de transmission doit être limité. Néanmoins, certains mécanismes d'alerte peuvent être désactivés à cause de la défaillance d'un lien radio. Par conséquent, il est recommandé d'envoyer l'alerte plusieurs fois. La figure 4 illustre ce type de scénarios.

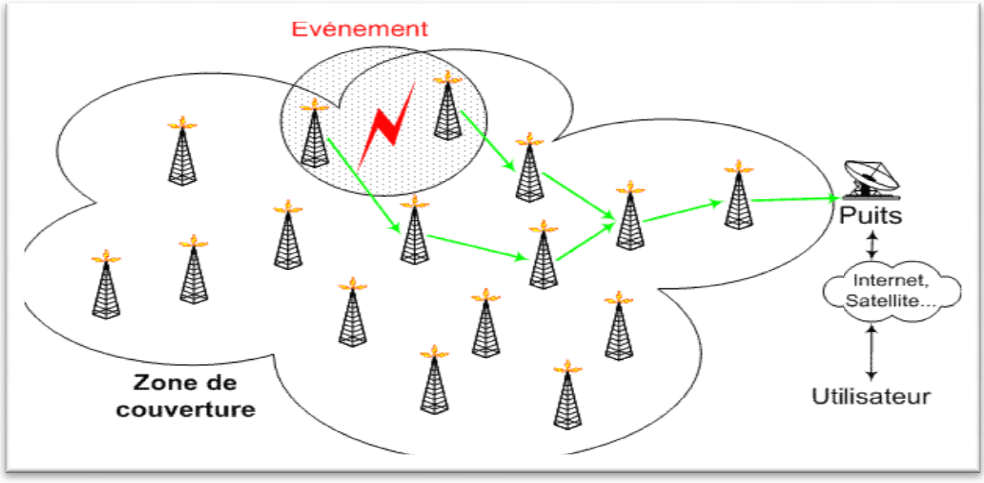

Figure 4: Scénario orienté événement

I.3 Les différents facteurs de conception d'un RCSF

La conception des RCSF est influencée par plusieurs facteurs qui doivent être traités d'une manière appropriée :

I.3.1 La durée de vie du réseau

La durée de vie d'un RCSF dépend de la période du temps durant laquelle le réseau est opérationnel et elle est liée à la durée de vie des nœuds capteurs qui le constituent.

Par ailleurs la consommation d'énergie par un nœud capteur correspond à la détection, la communication et le traitement de données [9]. Plusieurs définitions ont été proposées dans la littérature pour la durée de vie d'un RCSF. Par exemple dans [12], les auteurs ont supposé que la durée de vie d'un RCSF représente le temps pendant lequel le premier nœud épuise son énergie complètement, et dans [13], cette durée coïncide avec le temps pendant lequel le premier clusterhead (CH) épuise toute son énergie alors que dans [14], c'est la période pendant laquelle tous les capteurs épuisent leur énergie.

I.3.2 La tolérance aux pannes

Les nœuds capteurs sont sujets à des pannes dues à un épuisement d'énergie, à un problème physique ou une interférence. Par conséquent, ces pannes peuvent causer un dysfonctionnement du réseau et entraver ce dernier à effectuer sa mission dans des conditions favorables. D'où il faudrait instaurer des mécanismes qui permettent de garantir le bon fonctionnement du réseau et assurer la continuité de service quand un ou plusieurs capteurs cessent de fonctionner. En outre pour assurer la fiabilité de livraison il faut impliquer la qualité du lien et choisir les liens les plus fiables pour toute communication.

I.3.3 Topologie du réseau

La forte densité des RCSF dans les zones à surveiller nécessite une maintenance continue de la topologie par les nœuds capteurs qui doivent être capables d'adapter leur fonctionnement. Cette maintenance consiste généralement en trois phases : le déploiement (d'une manière aléatoire ou prédéfinie), le post-déploiement (les capteurs peuvent se déplacer ou cessent de fonctionner) et le redéploiement (l'ajout de nouveaux capteurs peut changer la topologie du réseau).

I.3.4 La consommation d'énergie

La limitation des ressources énergétiques rend ce genre de réseaux contraint en termes d'énergie puisque le rechargement ou bien le remplacement de la batterie est souvent impossible. C'est pour cette raison que l'économie d'énergie est la problématique majeure dans de nombreuses applications. Des mesures expérimentales ont montré que, la transmission des données est l'opération qui consomme plus d'énergie alors que les calculs consomment très peu d'énergie [15].

I.3.5 La sécurité

Les RCSF peuvent être déployés dans des endroits critiques où l'absence de la sécurité fait appel à des attaques qui peuvent perturber leur fonctionnement et même exploiter des informations importantes collectées par les capteurs de ces réseaux ou injecter de fausses informations. Dans ce cas de figure, la sécurité est l'une des contraintes essentielles surtout dans les applications critiques telles que les applications militaires ou les applications médicales.

I.3.6 Le lien radio

Les capteurs possèdent le matériel nécessaire pour effectuer des communications par ondes radio. Toutefois, la diffusion de l'information via ces liens est peu aisée à cause de l'instabilité et du manque de fiabilité qu'ils présentent. En plus, l'utilisation d'un médium de communication partagé pour faire face aux interférences radio, réduit considérablement la capacité d'exploitation du canal.

I.3.7 Auto-configuration

Les réseaux de capteurs sont généralement déployés aléatoirement dans des zones d'intérêt hostiles (capteurs largués par un avion). Par conséquent, aucune intervention humaine ne peut être requise pour assurer leur organisation. L'auto-configuration de ces réseaux s'avère nécessaire pour leur bon fonctionnement.

I.3.8 Passage à l'échelle

La plupart des protocoles sont conçus pour des réseaux de capteurs d'une grande taille. Cependant, ces protocoles sont dits efficaces si les performances des réseaux ne doivent pas chuter d'une manière drastique quand le nombre de capteurs augmente dans le réseau.

I.4 Domaines d'applications des réseaux de capteurs

La miniaturisation, l'adaptabilité, le faible coût et la communication sans fil permettent aux réseaux de capteurs d'envahir plusieurs domaines d'applications. Ils permettent aussi

d'étendre le domaine des applications existantes. Parmi ces domaines où ces réseaux se révèlent très utiles et peuvent offrir de meilleures contributions, on peut citer :

I.4.1 Applications militaires

Le déploiement rapide, l'auto-organisation et la tolérance aux pannes sont des caractéristiques qui ont rendu les réseaux de capteurs efficaces pour les applications militaires. Plusieurs projets ont été lancés pour aider les unités militaires dans un champ de bataille et protéger les villes contre des attaques, telles que les menaces terroristes. Le projet DSN (Distributed Sensor Network) [16] au DARPA[1] était l'un des premiers projets dans les années 80 ayant utilisés les réseaux de capteurs pour rassembler des données distribuées. Les chercheurs du laboratoire national Lawrence Livermore ont mis en place le réseau WATS[2] [17]. Ce réseau est composé de détecteurs des rayons gamma et des neutrons pour détecter et dépister les dispositifs nucléaires. Il est capable d'effectuer la surveillance constante d'une zone d'intérêt. Il utilise des techniques d'agrégation de données pour les rapporter à un centre intelligent.

Un réseau de capteurs peut être déployé dans un endroit stratégique ou hostile, afin de surveiller les mouvements des forces ennemies, ou analyser le terrain avant d'y envoyer des troupes (détection des armes chimiques, biologiques ou radiations). L'armée américaine a réalisé des tests dans le désert de Californie.

I.4.2 Applications à la surveillance

L'application des réseaux de capteurs dans le domaine de la sécurité peut diminuer considérablement les dépenses financières consacrées à la sécurisation des lieux et des êtres humains. Ainsi, l'intégration des capteurs dans de grandes structures telles que les ponts ou les bâtiments aidera à détecter les fissures et les altérations dans la structure suite à un séisme ou au vieillissement de la structure. Le déploiement d'un réseau de capteurs de mouvement peut constituer un système d'alarme qui servira à détecter les intrusions dans une zone de surveillance.

I.4.3 Applications environnementales

Le contrôle des paramètres environnementaux par les réseaux de capteurs peut donner naissance à plusieurs applications. Par exemple, le déploiement des thermo-capteurs dans une forêt peut aider à détecter un éventuel début de feu et par suite faciliter la lutte contre les feux de

[1] DARPA: Defense Advanced Research Projects Agency
[2] WATS: Wide Area Tracking System

forêt avant leur propagation. Le déploiement des capteurs chimiques dans les milieux urbains peut aider à détecter la pollution et analyser la qualité d'air. De même leur déploiement dans les sites industriels empêche les risques industriels tels que la fuite de produits toxiques (gaz, produits chimiques, éléments radioactifs, pétrole, etc.).

Dans le domaine de l'agriculture, les capteurs peuvent être utilisés pour réagir convenablement aux changements climatiques par exemple le processus d'irrigation lors de la détection de zones sèches dans un champ agricole. Cette expérimentation a été réalisée par le laboratoire "Laboratory and Agriculture and Agri-Food Canada" sur une vigne.

Figure 5: Exemple d'applications environnementales

I.4.4 Applications médicales

Dans le domaine de la médecine, les réseaux de capteurs peuvent être utilisés pour assurer une surveillance permanente des organes vitaux de l'être humain grâce à des micro-capteurs qui pourront être avalés ou implantés sous la peau (surveillance de la glycémie, détection de cancers, ..). Ils peuvent aussi faciliter le diagnostic de quelques maladies en effectuant des mesures physiologiques telles que : la tension artérielle, battements du cœur, ... à l'aide des capteurs ayant chacun une tâche bien particulière. Les données physiologiques collectées par les capteurs peuvent être stockées pendant une longue durée pour le suivi d'un patient [18]. D'autre part, ces réseaux peuvent détecter des comportements anormaux (chute d'un lit, choc, cri, ...) chez les personnes dépendantes (handicapées ou âgées).

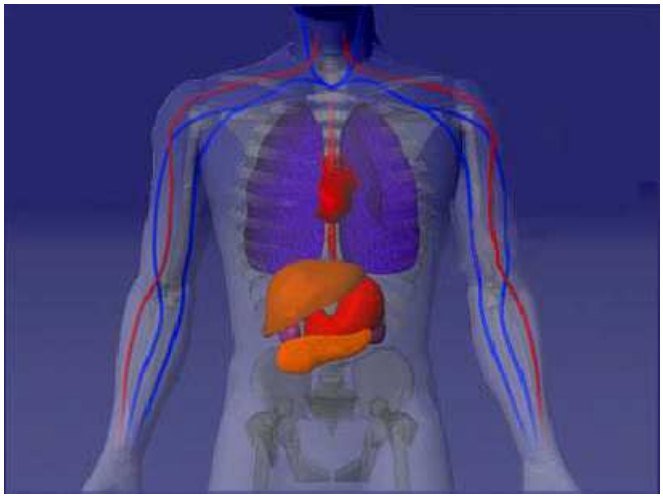

Figure 6: Applications médicales [18]

I.4.5 La domotique

Avec le développement technologique, les capteurs peuvent être embarqués dans des appareils, tels que les aspirateurs, les fours à micro-ondes, les réfrigérateurs, les magnétoscopes, ... [19]. Ces capteurs embarqués peuvent interagir entre eux et avec un réseau externe via Internet pour permettre à un utilisateur de contrôler les appareils domestiques localement ou à distance.

Le déploiement des capteurs de mouvement et de température dans les futures maisons dites intelligentes permet d'automatiser plusieurs opérations domestiques telles que : la lumière s'éteint et la musique se met en état d'arrêt quand la chambre est vide, la climatisation et le chauffage s'ajustent selon les points multiples de mesure, le déclenchement d'une alarme par le capteur anti-intrusion quand un intrus veut accéder à la maison.

I.5 Conclusion

Dans ce chapitre, nous avons présenté les réseaux de capteurs sans fil, leurs caractéristiques et les concepts nécessaires à la compréhension des réseaux de capteurs. En outre, nous avons mis le point sur quelques facteurs permettant la conception des RCSF en particulier la tolérance aux pannes.

Dans le chapitre qui suit, nous présentons quelques protocoles de routage tolérants aux pannes conçus pour les RCSF et nous détaillons le fonctionnement du protocole de routage LEACH qui fait l'objet de projet de fin d'études.

Chapitre II
Les protocoles de routage tolérants aux pannes dans les RCSF

Chapitre II
Les protocoles de routage tolérants aux pannes dans les RCSF

II.1 Introduction

Dans les RCSF, on doit assurer la fidélité de détection, de routage et de livraison. La fidélité de détection consiste à assurer la couverture de tout point de la zone d'intérêt par au moins un capteur. La fidélité de routage consiste à instaurer au moins un chemin entre tout capteur et la station de base alors que la fidélité de livraison permet d'assurer que les messages arrivent aux destinataires sans erreur. Cette caractéristique de routage nous a permis de concevoir plusieurs protocoles de routage tolérants aux pannes.

Dans ce chapitre, on présente les critères de performances des protocoles de routage tolérants aux pannes et les principaux protocoles de routage tolérants aux pannes dédiés aux RCSF. Puis, on met le point sur le protocole de routage LEACH qu'on va l'évaluer dans un environnement idéal et un environnement non-idéal représenté par le modèle lognormal shadowing pour illustrer sa robustesse dans un tel environnement. Malheureusement cette évaluation nous a montré que les performances de LEACH se dégradent dans un environnement non-idéal. Dans cette optique, nous avons proposé une version améliorée de LEACH qui s'adapte à ce type d'environnement.

II.1 Critères de performance des protocoles de routage dans les RCSF

La spécificité des réseaux de capteurs a permis d'instaurer des critères bien particuliers pour les protocoles de routage conçus à ce type de réseaux. Parmi ces critères de performance, nous citons [20]:

- **Évolutivité:** l'évolutivité est un facteur important dans les RCSF. Une zone de réseau n'est pas toujours statique, elle change selon les besoins des utilisateurs. Tous les nœuds dans le domaine du réseau doivent être évolutifs ou être en mesure de s'adapter aux changements de la structure du réseau. En plus, les capteurs sont sujets à des pannes. Ce qui permet de changer la topologie du réseau lors de l'occurrence des pannes.

- **L'énergie**: Les capteurs présentent une autonomie d'énergie et les batteries dont ils disposent sont rarement rechargeables et remplaçables. De ce fait, chaque nœud doit économiser la consommation de l'énergie pour assurer ses activités telles que la détection, le traitement, le stockage et la transmission. Pour acheminer les paquets vers la station de base, il devra utiliser un protocole de routage performant en termes d'énergie. Ce protocole de routage doit en particulier minimiser le nombre de paquets transmis puisque la transmission est l'opération qui consomme plus d'énergie.
- **Le temps de traitement**: il se réfère au temps pris par le nœud dans le réseau pour assurer l'ensemble des opérations commençant par la détection, le traitement des données, leur stockage, leur transmission ou leur réception sur le réseau. En outre, l'information devrait y arriver au poste de contrôle dans un laps de temps raisonnable en particulier pour les applications orientées événement.
- **Le schéma de transmission**: la transmission de données par les capteurs vers la destination ou la station de base se fait par un schéma de routage à un seul saut ou à multi-sauts. Par exemple, dans un réseau plat le schéma de routage se fait de nœud en nœud jusqu'à l'arrivée à la station de base. Cependant, dans les protocoles de routage hiérarchiques à l'instar de LEACH les clusterheads communiquent directement l'information à la station de base.
- **Synchronisation :** dans les communications radio entre les nœuds de capteurs d'un RCSF, les capteurs écoutent en permanence les transmissions et consomment de l'énergie s'ils ne sont pas synchronisés les uns des autres. Pour cela, un nœud doit savoir gérer son temps de veille en temps compte des périodes de veille de ces voisins.
- **Overhead**: Avant l'établissement des routes, les capteurs échangent des paquets de contrôle. De ce fait, un protocole de routage performant doit minimiser son overhead. En outre, lors de l'échange de données, on assiste à éviter les collisions pour minimiser les envois multiples.
- **Fiabilité de livraison** : la plupart des protocoles de routage ont été conçus dans un environnement idéal. Néanmoins, la présence des obstacles pour y avoir un impact négatif sur la qualité des messages reçus. De ce fait, il est recommandé de prendre en considération la qualité des liens avant toute communication.

II.3 La tolérance aux pannes dans les RCSF

Après le déploiement des RCSF, un nœud capteur peut cesser de fonctionner à n'importe quel moment durant son déploiement. Cette défaillance peut être due à un épuisement d'énergie, à une destruction physique accidentelle ou intentionnelle, ou à une interférence. La panne d'un nœud capteur aura éventuellement un impact sur le fonctionnement global du réseau. Pour remédier à cette anomalie, il faut instaurer la tolérance aux pannes qui permet de garantir le fonctionnement du réseau sans interruption même en présence de pannes.

II.3.1 Les pannes

La panne d'un système se produit lorsque son état devient inactif ou lorsqu'il fournit un résultat erroné. Elle est la conséquence d'une ou de plusieurs erreurs où une erreur représente un état invalide du système à partir duquel la poursuite de l'exécution est susceptible de conduire à une panne. En outre, une faute est la première cause de l'erreur qui pourrait provoquer la panne du système. La figure 1.6 montre la chronologie faute, erreur et panne.

Figure 7: Chronologie Faute/ Erreur/ Panne

On distingue quatre classes de pannes :

- La panne accidentelle: le composant soit cesse de fonctionner ou bien poursuit son fonctionnement mais dans un état invalide.
- La panne d'omission: le composant arrête de fonctionner définitivement.
- La panne de synchronisation : le composant réalise son traitement normalement mais fournit le résultat en retard.
- La panne byzantine: cette panne est arbitraire et imprévisible. Elle peut être due à des attaques malicieuses.

II.3.2 Les sources de pannes dans les RCSF

Les pannes sont inévitables dans les RCSF puisque ces derniers sont souvent déployés dans des zones hostiles et sont objets à des menaces. Ces pannes peuvent intervenir au niveau des différentes couches du système [21].

Par ailleurs, une panne au niveau d'une couche a la possibilité de se propager à des niveaux supérieurs. Par exemple, la panne d'un capteur qui est sur un chemin de routage se répercute sur l'ensemble des nœuds du même chemin puisque leurs messages ne peuvent pas être transmis jusqu'à ce que le chemin soit rétabli. La panne peut se produire au niveau des nœuds, du réseau et au niveau de la station de base.

a) La panne des nœuds

Un nœud est composé de plusieurs composants matériels et logiciels qui peuvent produire des dysfonctionnements à cause d'un épuisement de la batterie, une destruction physique, ou même coté logiciel puisqu'un capteur peut fournir des lectures erronées. Par exemple, dans [22], des expériences ont été menées pour tester la robustesse des capteurs quand ils sont en contact direct avec de l'eau ou un milieu très humide. Les résultats ont montré que les capteurs sont devenus défaillants à cause des courts-circuits. En outre, si le niveau de la batterie d'un nœud devient faible alors ses composants ne peuvent plus fonctionner correctement.

b) La perturbation des réseaux

Le routage est une opération fondamentale dans les RCSF. Il est essentiel de transmettre, recevoir les données, distribuer des mises à jour logicielles et de coordonner entre les nœuds. En outre, il pourrait y avoir des applications de routage spécifiques, par exemple suivre les objets en mouvements car une défaillance au niveau de la couche de routage peut générer des messages perdus ou à des retards de transmission de données inacceptables.

Dans les RCSF, les liens de communication entre les nœuds sont peu fiables à cause des interférences radio ou la présence d'obstacles. Par exemple, dans [23] le taux d'erreur est 42% en présence de perturbations dans le réseau et dans [24] l'instabilité des liens radios entre les nœuds a conduit à des changements fréquents dans les chemins de routage.

c) La panne de la station de base

La station de base est aussi sujet à des pannes de ses composants. Lorsque ce dispositif échoue une défaillance massive du réseau se produit puisque les données collectées par les nœuds capteurs ne sont plus exploitables sauf si des dispositifs de tolérance aux pannes sont mis en place pour faire face à ce type de pannes qui pourra y avoir une grande conséquence sur le fonctionnement du réseau en entier.

Par ailleurs, la station de base peut être déployée dans des zones où aucune alimentation permanente n'est présente. Dans certaines applications les batteries sont dotées de cellules

solaires pour être alimentées [25] alors que dans la plupart des applications telles que l'expédition glaciaire rapportée dans [26], cette technique traditionnelle s'est avérée inefficace.

II.4 Les principaux protocoles de routage tolérants aux pannes

La tolérance aux pannes peut être réalisée en appliquant les protocoles de routage au niveau de la couche réseau dans le cas où certains nœuds présentent une défaillance.

On distingue deux architectures de protocoles de routage: architecture plate, architecture hiérarchique auxquelles il y avait des protocoles de routage tolérants aux pannes sous-jacents.

Dans cette section, on présente les principaux protocoles de routage tolérants aux pannes pour chaque catégorie ainsi que leurs fonctionnalités.

II.4.1 Les protocoles de routage plats tolérants aux pannes

Dans cette classe, les capteurs communiquent via un routage multi-sauts jusqu'à ce qu'ils atteignent leurs destinations. Ils jouent tous le même rôle et collaborent entre eux pour accomplir la tâche de détection. Par ailleurs, la panne d'un nœud dans cette classe est généralement tolérée par la technique de multi-chemins. Dans ce qui suit, on présente les fonctionnalités de certains protocoles.

a) Fault-Tolerant Multilevel Routing Protocol (FMS)

Ajay et al. [27] ont proposé un protocole de routage multi-niveaux tolérant aux pannes avec un ordonnancement de l'activité des capteurs appelé FMS. Le but de ce protocole est de maintenir la connectivité du réseau même si un nœud épuise son énergie.

FMS assurer la fiabilité de livraison de données pour les applications orienté-événement. Il est conçu pour les RCSF denses dont les nœuds sont déployés aléatoirement et chaque nœud possède son propre identifiant (ID_r) ainsi que la communication entre les nœuds voisins est bidirectionnelle. En outre, FMS propose un ordonnancement de sommeil aléatoire afin de conserver l'énergie des capteurs tel que la mise en veille d'un nœud ne prévient pas ce dernier de détecter des grandeurs dans son voisinage puisque l'unité d'acquisition sera toujours allumée et la radio sera en mode éteint.

Le fonctionnement du protocole FMS s'effectue en deux phases. La première phase est identique à celle du protocole décrit dans [28] et la deuxième phase consiste à effectuer le processus d'ordonnancement de sommeil périodiquement et la transmission de données. Au cours de cette deuxième phase, un ensemble de nœuds est sélectionné aléatoirement et mis en

mode veille tandis que l'autre ensemble de nœuds est mis en mode actif. Cette opération est exécutée périodiquement et durant chaque période, seulement les nœuds actifs sont responsables de la transmission de données. Si l'énergie résiduelle d'un nœud actif est inférieure à un seuil prédéfini, ce dernier diffuse un message de notification à tous ses nœuds fils qui vont chercher un autre nœud parent qui a plus d'énergie. De ce fait, les chemins alternatifs sont établis et la connectivité du réseau reste toujours maintenue.

b) ENhanced FAult-Tolerant AODV routing protocol (ENFAT-AODV)

ENFAT-AODV [29] est un protocole de routage multi-sauts tolérant aux pannes destiné aux RCSF. Il permet d'établir rapidement des chemins efficaces entre les nœuds communicants. En outre, ce protocole associe à chaque chemin principal de livraison de données un chemin alternatif. De ce fait, c'est une transmission de données sur le chemin principal échoue, le chemin alternatif sera emprunté. Par ailleurs, le plus court chemin est considéré comme chemin principal et le second sera considéré comme chemin alternatif.

Pour l'établissement de routes, ENFAT-AODV utilise des messages de contrôle similaires à ceux utilisés par AODV [30] tels que RREQ pour la demande de routes et REP comme message de réponse à cette demande.

Les principales opérations du protocole ENFAT-AODV sont :

- **La découverte du chemin principal**: le nœud source effectue le processus de découverte du chemin principal en diffusant le message de demande de route principale (main RREQ) pour atteindre le nœud destinataire. Quand les nœuds intermédiaires reçoivent ce message, ils créent une route inverse vers le nœud source. Si ce message est reçu pour la première fois par un nœud qui n'est pas le nœud destinataire, ce dernier le diffuse à tous ses voisins. En outre, si c'est le nœud destinataire ou s'il possède une route principale vers la destination, il génère une réponse de route principale (main RREP). Ensuite, le message (Main RREP) est transmis saut par saut au nœud source et en ce moment tous les nœuds intermédiaires créent une route vers la destination. Quand le nœud source reçoit ce message, il enregistre la route dans sa table de routage principale.
- **La construction de chemin alternatif**: Durant le processus de réponse de route principale (main RREP), les nœuds qui appartiennent au chemin principal, créent une route alternative vers le nœud destinataire en diffusant un message de demande de route alternative "backup RREQ". Ensuite l'émetteur attend une réponse de route alternative "backup RREP" de la destination ou bien des nœuds intermédiaires qui n'appartiennent pas à la route alternative.

- **Maintenance des routes**: Durant la phase de transmission de données, si le chemin principal a des liens brisés ou bien la destination du paquet transmis n'est pas active sur la route principale, le nœud utilise immédiatement sa route alternative pour transmettre le prochain paquet de données sans interrompre la transmission de paquet de données. Par suite, le nœud sur le nouveau chemin principal et qui ne possède pas une route alternative effectue un processus de découverte de route alternative "Backup route discovery" pour trouver une nouvelle route alternative.

En résumé on pourra dire que le protocole ENFAT-AODV assure la fiabilité de routage et la disponibilité de routes vers les nœuds destinataires comparativement à la version originale d'AODV.

c) Fault-Tolerant Directed Diffusion for wireless sensor networks (FaT2D)

FaT2D [31] est un protocole tolérant aux pannes basé sur le protocole "Directed Diffusion" [32]. Il permet de garantir une tolérance aux pannes contre les nœuds défaillants en utilisant le mécanisme multi-chemins, l'exploration périodique et la technique de renforcement positif/négatif. En outre, FaT2D définit une nouvelle technique qui permet la détection rapide de pannes avec une récupération rapide du chemin malgré la défaillance des nœuds et le changement de topologie et il s'exécute selon les phases suivantes :

- **Détection de pannes** : Un délai de détection de pannes "Tfd" a été introduit en vue de réduire le délai de recouvrement de panne et par conséquent accélérer la détection de défaillance d'un nœud et la réparation locale du chemin. Si le temps "Tfd expire, le protocole FaT2D transmet immédiatement un nouveau message appelé "ExploreRequest" pour notifier la détection de pannes et demander une nouvelle exploration pour trouver un chemin fiable qui remplace celui qui est défaillant. Ainsi, chaque nœud dans le chemin défectueux élimine le gradient correspondant pour remédier aux défaillances.
- **Recouvrement rapide du chemin**: Quand Tfd expire, cela notifie la défaillance d'un nœud. De ce fait FaT2D lance le processus pour réparer le chemin défectueux en transmettant un message de demande d'exploration spéciale nommé "ExploreRequest" qui contient des informations sur le chemin défectueux. Le message "ExploreRequest" sera envoyé afin d'atteindre la source principale des informations en rapport avec la route défectueuse sans invoquer des boucles de transmission ou bien la recherche des nœuds inappropriés. Lorsque le nœud source reçoit la demande d'exploration, il arrête de le transmettre et il commence une exploration par inondation comme dans le protocole "Directed Diffusion". Ce mécanisme

génère une exploration rapide pour trouver une nouvelle route fiable avec les mêmes règles du protocole "Directed Diffusion".

- **Elimination de pannes**: Pour chaque nœud intermédiaire qui reçoit le message "ExploreRequest", le protocole FaT2D vérifie si ce nœud appartient au chemin défectueux ou non. Si ce dernier appartient à ce chemin, il va renforcer négativement son gradient. En outre, chaque nœud exécute un renforcement négatif local à tous ses voisins en amont afin de supprimer le chemin défectueux et arrêter l'envoi des données sur ce chemin.

II.4.2 Les protocoles de routage hiérarchiques tolérants aux pannes

Dans cette classe de protocoles de routage tolérants aux pannes, il est généralement considéré que les clusterheads sont sujets à des pannes dues à plusieurs tâches et cette panne est couverte par le CH adjoint (CH_v) qui le remplace. Ainsi, si le CH tombe en panne le cluster devient paralysé et donc le réseau perd ses fonctionnalités. Dans ce qui suit, nous présentons certains protocoles de cette classe:

a) Cluster-based Fault-tolerant Scheme (CFS)

CFS [33] est un protocole de routage tolérant aux pannes qui a comme but de tolérer les pannes des liens afin de garantir un routage fiable de données.

CFS est exécuté en trois phases :

- **La formation de clusters** : Chaque cluster possède deux clusterheads: le CH principal (CH_p) et son adjoint (CH_v). L'élection de ces clusterheads est basée sur le poids du nœud qui est une combinaison entre la 2-densité du nœud et son énergie résiduelle. Puisque le CH est responsable de plusieurs tâches, le protocole est exécuté en périodes (rounds) pendant lesquelles le nœud ayant le plus grand poids dans son 2-voisinage est élu comme CH_p et le deuxième comme CH_v. Après, le CH_p diffuse un message d'avertissement pour construire son cluster. Chaque nœud qui n'est pas un CH et n'est pas un membre d'un autre cluster, transmet un message REQ-JOIN pour joindre son CH_p qui a envoyé le message.
- **L'établissement des routes** : Tous les CHs établissent un chemin CH-à-CH envers la station de base pour la transmission de données et créent un temps pour la communication intra-cluster pour éviter les interférences. Lorsqu'un événement est détecté, le membre concerné envoie cette information aux CH_p et CH_v en même temps. Si CH_p ne relaye pas le message dans un timeout il sera considéré comme défaillant et le CH_v le transmettra.

- **Transmission de données**: Dans cette phase, les nœuds commencent à transmettre les données collectées à leurs CH_p correspondants. Après la radio de chaque membre est éteinte jusqu'à l'arrivée de son temps de transmission. Chaque CH agrège les données reçues de ses membres et les envoie vers la station de base.

b) Fault-Tolerant Energy-Efficient Distributed Clustering for WSN (FEED)

Dans le protocole FEED [34], le réseau est divisé en un ensemble de clusters. Chaque cluster contient un clusterhead (CH), un pivot de clusterhead (PCH) qui a des capacités supplémentaires par rapport à un clusterhead, et d'un nœud superviseur (SN) qui peut remplacer son CH ou son PCH correspondant lorsque l'un des deux tombe en panne. L'élection du CH est basée sur certaines métriques telles que la densité, la centralité, l'énergie restante et la distance entre les nœuds.

FEED se déroule en quatre phases :

- **La première phase** : Les nœuds échangent un message entre eux et chacun d'eux calcule la densité et la centralité. Dans ce protocole, une nouvelle méthode de calcul de centralité est introduite. Ensuite, chaque nœud calcule son premier score en fonction de son énergie, sa densité et sa centralité.
- **La deuxième phase** : Chaque nœud qui a le plus grand poids (énergie, densité et centralité) parmi ses voisins, se considère comme volontaire.
- **La troisième phase**: Dans cette phase, chaque nœud ajoute le facteur distance au premier score calculé dans la phase précédente, et le volontaire avec le deuxième meilleur score est élu adjoint.
- **La dernière phase** : Les nœuds volontaires calculent leur score final et selon ces scores calculés, ils s'affirment comme des nœuds CH, PCH ou SN. Après cela chaque nœud dont le statut est différent de CH, PCH et SN, rejoint le cluster le plus proche.

c) Energy-efficient fault-tolerant clustering and routing algorithms for WSN (DFCR)

DFCR [35] est un protocole qui prend en considération la conservation d'énergie et la tolérance aux pannes qui sont les deux enjeux majeurs dans le déploiement des RCSF. Au début de cet algorithme, la station de base diffuse un message "HELLO" pour que tous les CHs peuvent calculer la distance à la station de base en fonction de RSSI[3] du message reçu. Puis la

[3] RSSI: Received Signal Strength Indicator

station de base diffuse un message "HopPacket" qui indique le nombre de sauts du CH à la station de base. Après cela, chaque CH envoie le paquet à tous ses CHs voisins. En recevant ce paquet, chaque CH compare la valeur du compteur avec la valeur de son propre compteur. Si cette valeur reçue est inférieure à celle stockée le CH incrémente son compteur de 1 et le retransmet, sinon, il ignore le paquet. Ensuite, la phase de mise en place sera lancée où chaque CH diffuse un message "HELLO" dans sa portée de communication. Le nœud qui reçoit au moins un message, est considéré comme couvert. Par ailleurs, il diffuse un message "Help" et chaque nœud voisin qui reçoit ce message, envoie une réponse pour former son chemin alternatif.

En outre, il peut exister des nœuds qui ne sont pas couverts par un CH en raison de déploiement aléatoire ou bien la défaillance brutale de CHs. Dans ce cas, ces nœuds sont affectés à un autre CH via une communication multi-sauts utilisant les nœuds couverts comme nœuds relais. Dans la deuxième phase, les CHs agrègent les données reçues de leurs membres et sélectionnent le prochain nœud relai pour atteindre la station de base. Cette sélection dépend de la distance et le nombre de sauts calculés précédemment pour minimiser la consommation d’énergie.

II.5 Le protocole de routage LEACH

LEACH [4] est l'un des algorithmes de routage hiérarchique le plus populaire pour les réseaux de capteurs. L'idée est de former des clusters de nœuds de capteurs basés sur les zones où il y a un fort signal reçu, puis utiliser des clusterheads locaux comme passerelle pour atteindre la station de base. Cela permet d'économiser de l'énergie car les transmissions ne sont effectuées que par les clusterheads plutôt que par tous les nœuds ordinaires.

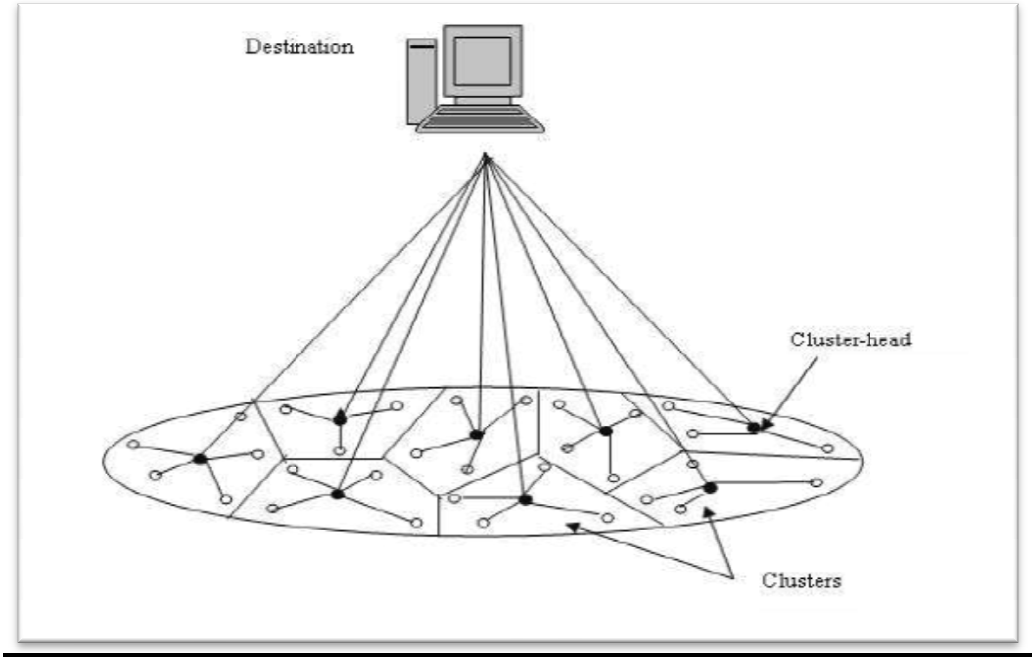

Figure 8: Architecture de LEACH [4]

L'architecture de communication de LEACH est similaire à celle des réseaux cellulaires. Elle consiste à former des cellules basées sur la puissance du signal reçu, et utiliser les têtes de cellules comme routeurs vers le nœud puits. Ces cellules sont appelées groupes (*clusters*), quant aux têtes : chefs de groupes (*clusterheads* CH). Les CHs sont choisis de façon aléatoire selon un algorithme spécifique d'élection basé sur une fonction de probabilité comme montre l'équation (1).

$$T(n) = \begin{cases} \dfrac{P}{1 - P \times \left(r \bmod \frac{1}{P}\right)} & Si\ (n \in G) \\ 0 & Sinon \end{cases} \qquad (1)$$

Dans LEACH, les nœuds sont chargés de collecter des données, les envoyer à leurs CHs correspondants qui les agrègent et les transmettent directement au nœud puits. En outre, les CHs ont pour mission d'assurer les fonctions les plus coûteuses en énergie, à savoir la communication avec le nœud puits qui est supposé éloigné, ainsi que tous les traitements de données (agrégation, fusion et transmission de données) afin de réduire la quantité des données transmises. Ce dispositif permet d'économiser l'énergie puisque les transmissions sont uniquement assurées par les CHs plutôt que par tous les nœuds du réseau. Par conséquent, LEACH réalise une réduction significative de la dissipation d'énergie. La figure 9 résume le fonctionnement de LEACH.

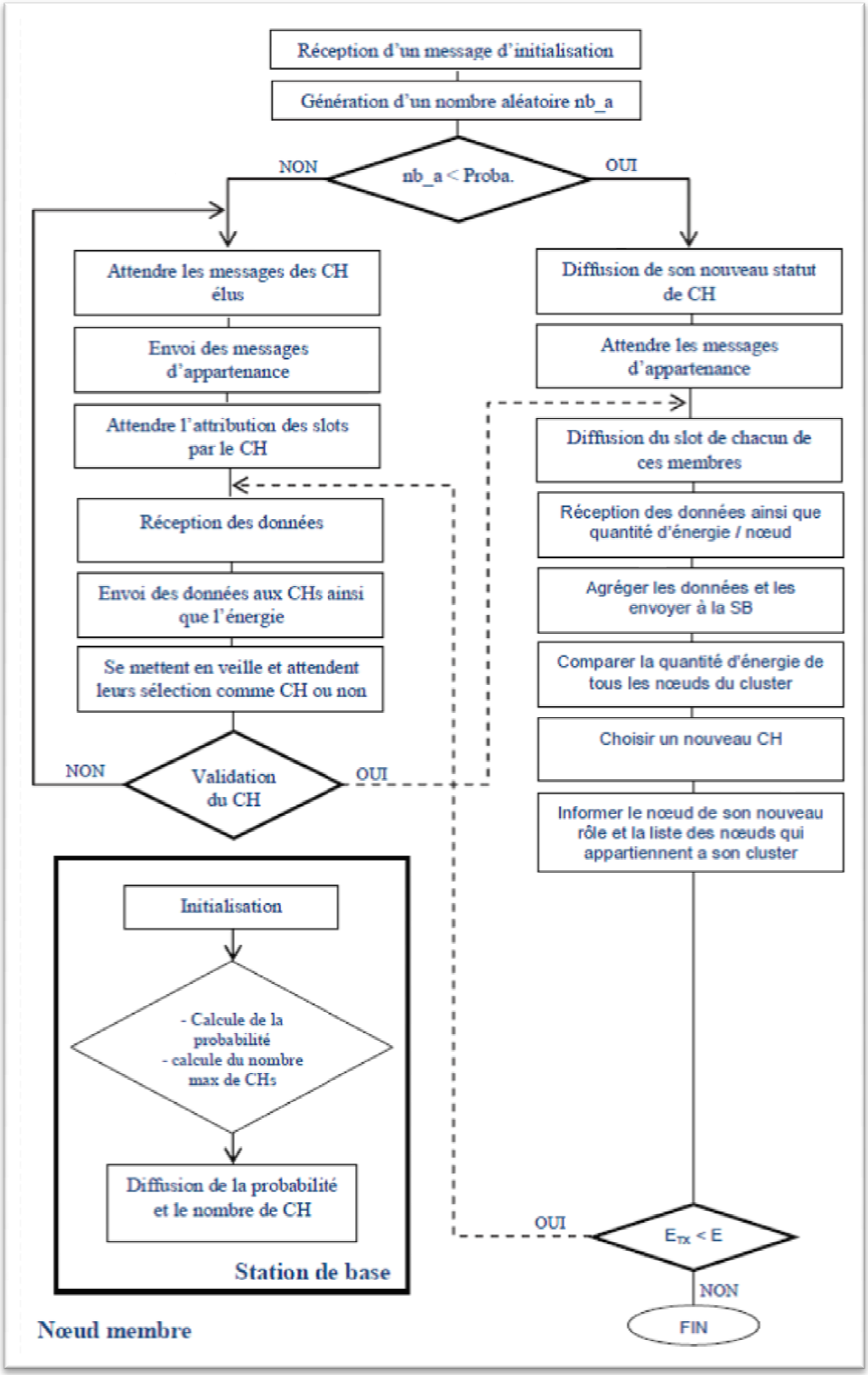

Figure 9: Fonctionnement de LEACH [4]

II.6 Conclusion

Dans ce chapitre, nous avons présenté la spécificité de routage dans les réseaux de capteurs, la tolérance aux pannes dans les RCSF, les protocoles de routage tolérants pannes les plus répandus, et le protocole LEACH qui fait l'objet de ce projet de fin d'études.

Dans le chapitre qui suit, nous allons évaluer les performances du protocole de routage LEACH dans un environnement non-idéal représenté par le modèle lognormal shadowing. Puis, nous avons proposé une version améliorée qui s'adapte à ce type d'environnement.

Chapitre II
Une version de LEACH adaptée au modèle LogNormal Shadowing

Chapitre III
Une version de LEACH adaptée au modèle LogNormal Shadowing

III.1 Introduction

La plupart des protocoles de routage conçus pour les RCSF ont été basés sur une hypothèse idéale concernant le canal sans fil c'est-à-dire le canal radio est considéré toujours fiable et que les messages arrivent toujours sans erreur tant que la distance séparant les nœuds communicants est inférieure à la portée de transmission. Cette hypothèse ne reflète pas une réalité dans les RCSF. De ce fait, il s'avère nécessaire d'étudier les performances des protocoles conçus pour un environnement idéal dans un environnement non idéal pour voir leur robustesse dans ce type d'environnement.

Dans un environnement réaliste, les fluctuations du signal radio peuvent provoquer la non-fiabilité des liens de communication entre les nœuds et par la suite provoquer la perte de paquets de données ou générer des paquets corrompus. Dans des travaux, les auteurs ont proposé des protocoles dans un environnement dont les liens de communication sont non fiables. Dans ces protocoles, les nœuds relais sont évalués et le nœud avec la plus grande probabilité de livraison fiable de données est considéré comme nœud fiable.

Dans ce chapitre, nous reproduisons le protocole de routage LEACH en Nesc [36] et nous l'analysons dans un environnement non-idéal représenté par le modèle LogNormal Shadowing [6] pour montrer la dégradation des performances de ce protocole dans ce type d'environnement. Puis, nous proposons une nouvelle version de LEACH qui s'adapte à ce type d'environnement. Pour ce faire, nous avons utilisé le simulateur TOSSIM [37].

Pour réaliser ce travail, nous présentons le modèle LogNormal Shadowing, les outils logiciels nécessaires à l'évaluation de LEACH tels que TinyOS, NesC et TOSSIM. Nous terminerons ce chapitre par une présentation des résultats relevés lors des tests des performances de LEACH dans les deux modèles en termes de nombre de paquets échangés, de taux de paquets perdus et de consommation d'énergie.

III.2 Le modèles de propagation utilisés

Dans cette section, on présente les deux modèles utilisés pour évaluer les performances de LEACH : le modèle de disque unitaire (UDG) et le modèle LogNormal Shadowing (LNS).

III.2.1 Le modèle du disque unitaire

Soit un réseau de capteurs sans fil contenant N nœuds connectés entre eux. On modélise ce réseau par un graphe non orienté G = (E, V) tels que E représente l'ensemble des arêtes représentant les liens bidirectionnels entre deux nœuds et V l'ensemble des nœuds. On suppose également que tous les nœuds ont la même portée de transmission désigné par R_c.

Le modèle de disque unitaire définit l'ensemble E des arêtes selon l'équation (2).

$$E = \{(u, v) \in V^2 : \quad dist(u, v) \leq R_c\} \qquad (2)$$

où $dist(u, v)$ est la distance euclidienne entre u et v.

Ce modèle bien qu'il soit couramment utilisé ne peut pas être considéré comme un modèle réaliste car il suppose que les messages sont toujours reçus sans erreur si la distance entre les nœuds communicants est inférieure ou égale au rayon de transmission R_c. Cette hypothèse ne tient pas compte des fluctuations aléatoires du signal radio, qui peut avoir un impact significatif sur les transmissions.

III.2.2 Le modèle LogNormal Shadowing

Dans ce modèle, la qualité du lien entre deux nœuds communicants est estimée en fonction de la distance qui les sépare et le coefficient d'atténuation de l'environnement (α). L'équation (3) explicite le calcul de la probabilité d'une réception sans erreur.

$$F(x) = \begin{cases} 1 - \dfrac{\left(\frac{x}{R_c}\right)^{2\alpha}}{2}, & si \;\; 0 < x \leq R_C \\ \dfrac{\left(\frac{2R_c - x}{R_c}\right)^{2\alpha}}{2}, & si \; R_C < x \leq 2R_c \\ 0 & Sinon \end{cases} \qquad (3)$$

où (α) représente le facteur d'atténuation qui dépend de l'environnement. Sa valeur varie entre 2 et 4. Par exemple dans un environnement fortement perturbé sa valeur est de l'ordre de 4.

x est la distance séparant les deux nœuds communicants et R_c est la portée de transmission d'un nœud dans un environnement idéal.

III.3 Outils logiciels

Dans cette section, on présente les outils logiciels nécessaires pour l'évaluation de LEACH dans un modèle de propagation probabiliste. Tout d'abord, on traduit le fonctionnement de LEACH en langage NesC. Puis, on le simule en utilisant TOSSIM [37].

III.3.1 Le système d'exploitation TinyOs

Les contraintes des RCSF ont motivé l'université de Berkeley et d'autres contributeurs à développer un système d'exploitation destiné aux RCSF afin de faciliter l'implémentation et l'exécution de protocole dédiés à ce type de réseaux.

L'objectif de TinyOs consiste à minimiser la taille du code afin de respecter les contraintes de ressources énergétiques et physiques des nœuds capteurs. Ce système a l'avantage de permettre une programmation simple et puissante tout en gardant la portabilité du code pour les nombreuses plateformes supportées. Il respecte une architecture basée sur une association de composants et utilise une programmation entièrement réalisée en langage NesC.

a) Propriétés de TinyOS

TinyOS reste néanmoins le plus répandu pour les RCSF car il répond aux exigences particulières des applications des RCSF. Il convient alors de mentionner les propriétés qui rendent TinyOS aussi populaire et réputé pour ce genre de réseaux [38] **:**

- Une taille de mémoire réduite.
- Une basse consommation d'énergie.
- Des opérations robustes.

- Applications orientées composants: TinyOS fournit une réserve de composants systèmes utilisables au besoin.
- Programmation orientée évènement: Généralement sur TinyOS, un programme s'exécute suivant le déclenchement des événements. Sinon, les capteurs restent en veille ce qui maximise la durée de vie du réseau.

b) Eléments de Tinyos

TinyOS est construit autour des différents concepts décrits ci-dessous.

- **Les composants :** constitués de:
  - **Frame :** est un espace mémoire de taille fixe permettant au composant de stocker les variables globales et les données qu'il utilise. Il n'en existe qu'un seul par composant.
  - **Tâches :** contiennent l'implémentation des fonctions. Elles sont décomposées en deux catégories : les commandes et les évènements.
- **Les interfaces :** représentent le descriptif des fonctions définies dans les tâches.

III.3.2 Le langage NesC

NesC [36] est un langage de programmation orienté composants syntaxiquement proche du langage C. Il est conçu pour la réalisation des systèmes embarqués distribués, en particulier, les RCSF.

NesC s'appuie sur des composants ce qui permet de décomposer une application en modules réutilisable. Il utilise trois abstractions de programmation: interface, module et configuration :

- **Interface**: contient les signatures des commandes (command) et des évènements (event) que le module devra implémenter. Comme TinyOs n'autorise pas les pointeurs de fonctions,NesC propose une alternative : Les interfaces paramétrées. interface SendMessage [uint8 id] permet de créer 256 interfaces de type SendMessage, il suffit alors d'en sélectionner une à l'aide de son identifiant.

- **Module**: permet d'implémenter les composants. Un module est composé de deux blocs, le premier contient les déclarations, le second l'implémentation. Dans le premier bloc, les mots clés <<use>> et <<provide>> permettent de savoir si le module fait appel à une fonction de l'interface (use) ou redéfini son code (provide), <<use nomInterface>> signifie que le module doit fournir l'implémentation des évènements de cette interface, <<providenomInterface>> signifie que le module doit fournir les commandes de cette interface. Le second bloc

commence par le mot clé implémentation et se compose des variables privées et de l'implémentation des différentes commandes et évènements. Un module est de la forme suivante :

```
Module nomModule{
Provides {
        //liste des interfaces fournies
        Interface nomInterface }
Uses {
        //liste des interfaces requises
        Interface nomInterface  }
Implémentation {
```

- Configuration: définit le ou les composants qui seront utilisés par l'application (permet de décrire les composants composites). Elle est constituée de modules et/ou d'interfaces et de la déclaration des liaisons entre composants. Elle est de la forme suivante :

```
Configuration nomConfig{}

Implémentation {
        //liste des modules et configurations utilisées
        Composants Main, Module1,……..ModuleN, Config1…….……Configm
        //description des liaisons
```

On distingue les modules et les configurations dans le but de permettre aux concepteurs d'un système de construire des applications rapidement et efficacement. Par exemple, un concepteur peut fournir uniquement une configuration qui relie un ensemble de modules qu'il ne développe pas lui même. De plus, un autre développeur peut fournir une librairie de modules qui peuvent être utilisés dans la construction d'autres applications.

La figure 10 illustre les interactions entre les différents éléments dans une application orientée composants.

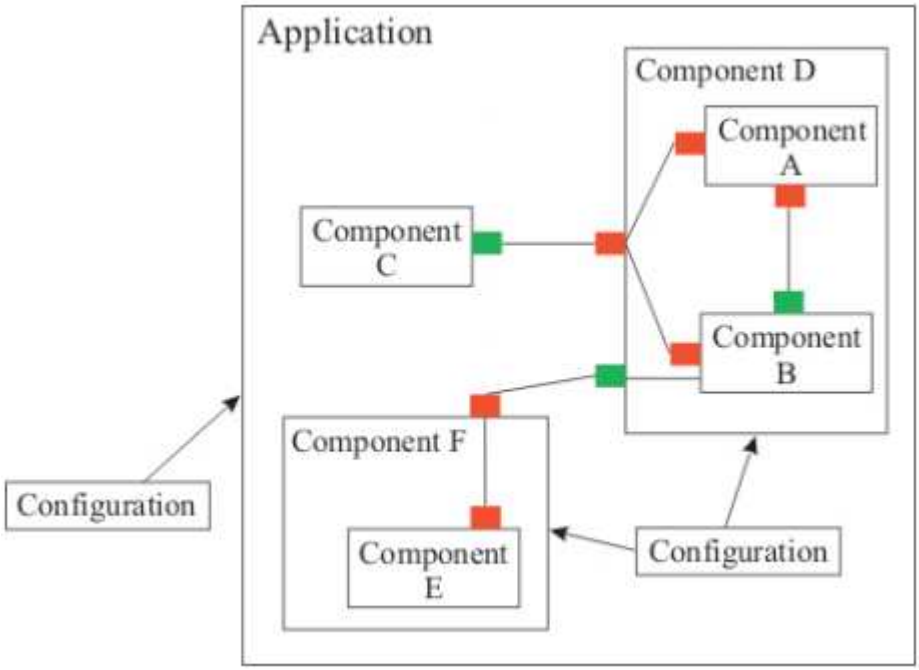

Figure 10: Architecture d'une application en NesC [36]

III.3.3 TOSSIM

Avant sa mise en place, le déploiement d'un RCSF nécessite une phase de simulation afin de s'assurer du bon fonctionnement de tous les protocoles de communication qu'il utilise.

En effet, pour de grands réseaux, le nombre de capteurs peut atteindre plusieurs milliers et entraine donc un coût financier relativement important. Ainsi, il faut réduire au maximum les erreurs de la conception. Malgré cela, il reste des facteurs réels qui ne peuvent être pris en compte par la simulation, tels que les contraintes physiques (perturbations électromagnétiques, inondations, etc.) ou les aléas (détériorations dues à un animal, etc.). Pour arriver à simuler le comportement des capteurs au sein d'un RCSF, un outil très puissant a été développé et proposé pour TinyOS sous le nom de TOSSIM. Le principal but de TOSSIM est de créer une simulation très proche de ce qui se passe dans les RCSF dans le monde réel. Une économie d'effort et une préservation du matériel sont possibles grâce à cet outil**.**

Pour une compréhension moins complexe de l'activité du réseau, TOSSIM peut être utilisé avec une interface graphique TinyViz. Cette dernière est équipée par plusieurs API plugins qui permettent d'ajouter plusieurs fonctions à notre simulateur comme par exemple suivre la dépense d'énergie en utilisant un autre simulateur qui s'appelle PowerTOSSIM [39].

III.3.4 TinyViz

TinyViz [40] est une interface graphique Java. Elle permet de donner un aperçu des capteurs à tout instant ainsi que des divers messages qu'ils émettent. Elle détermine un délai entre chaque itération des capteurs afin de permettre une analyse pas à pas du bon déroulement des actions en activant différents modes comme Radio, CPU, etc.

Nous allons détailler un peu ce que fait chaque bouton présent dans l'interface :

- **ON/OFF** : il met en marche ou éteint un capteur.
- **Delay** : il permet de sélectionner la durée au bout de la quelle se déclenche le timer.
- **Play** : il permet de lancer la simulation ou de la mettre en pause.
- **Grilles** : il permet d'avoir une grille pour situer les capteurs en espace.
- **Clear** : il efface tous les messages qui transitent entre les capteurs.
- **Arrêt** : il met fin à la simulation.

III.4 Implémentation et évaluation de LEACH

Notre travail s'est déroulé en deux phases :

1. Evaluation de LEACH dans un environnement idéal représenté par le modèle du disque unitaire.
2. Evaluation de LEACH dans un environnement réaliste représenté par le modèle lognormal shadowing.

Nous donnons un aperçu sur l'implémentation de LEACH en NesC.

Notre application s'articule autour des composants suivants :

- un module, appelé «MHLeachPSM.nc ».
- une configuration, appelée «MHLeachRouter.nc ».
- le fichier d'entête, appelé «MH.h ».

III.4.1 Implémentation de LEACH en NesC

Dans cette section, nous décrivons les structures de données ainsi que les principaux commandes et événements nécessaires pour l'implémentation du protocole LEACH.

Un paquet dans TinyOS est représenté par une structure appelée TOS_Msg, qui est contenue dans un champ « int8_t data[TOSH_DATA_LENGTH] ». Les structures de données du paquet diffèrent selon le statut du nœud (station de base, CH ou membre).

Le nœud puits

```
typedef struct PUITS
{uint16_t ID; //l'identificateur du puits qui correspond à tos_local_address=0
uint8_t round; //le round courant
float probability; //la probabilité que chaque nœud devienne CH
uint8_t Depth; //la puissance du signal d'un CH dans le réseau
}PUITS;
```

Le nœud CH

```
typedef struct CLUSTER_HEAD
{
uint16_t ID_CH; //l'identificateur de chaque CH qui correspond à tos_local_address
uint16_t ID_MEMBRE; //l'identificateur du membre qui appartiendra à ce CH
uint8_t data_agre; //la donnée agrégée à envoyer au nœud puits
uint16_t SLOT_ATT; //le slot attribué à chaque membre
uint16_t FREQ; //la fréquence avec laquelle un membre envoie sa donnée
}CLUSTER_HEAD;
```

Le nœud membre

```
typedef struct MEMBRE
{
uint16_t ID_MEMBRE;//l'identificateur de chaque membre qui correspond à tos_local_adress
uint16_t ID_CH; //l'identificateur du CH auquel appartiendra le nœud membre
uint8_t temp; //la température captée
}MEMBRE;
```

III.4.2 Version améliorée de LEACH : Improved_LEACH

Nous présentons la version améliorée de LEACH qui s'adapte au modèle LNS. Dans cette version, pour une communication intra-cluster si la probabilité de réception sans erreur entre un membre et le clusterhead correspondant est supérieure au seuil prédéfini alors le message est reçu par le CH correctement sinon un membre fiable est impliqué comme nœud relais pour atteindre le clusterhead. Le schéma algorithmique suivant illustre cette amélioration.

Algorithme : Communication intra-cluster

Début

- $d_h(M_i, CH)$ est la distance euclidienne entre le nœud membre M_i et CH
- $\begin{pmatrix} x_c \\ y_c \end{pmatrix}$: les coordonnées du CH,
- $\begin{pmatrix} x_i \\ y_i \end{pmatrix}$: les coordonnées du membre M_i,

$Mem(CH_i) = \emptyset$

While Actif(CH) **do**

- M_i calcule la distance qui le sépare de son CH

$$d = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}$$

- Mi calcule la probabilité de réception sans erreur,

$$pr(d) = 1 - \frac{\left(\frac{d}{R_c}\right)^{2\alpha}}{2}$$

if $(pr(d) \geq Seuil)$ **then**

- Mi envoie ses données directement à son CH

$Mem(CH_i) = Mem(CH_i) \cup \{M_i\}$

else

- M_i sélectionne un nœud relais $\left(M_j \in Mem(CH_i)\right)$ avec une distance minimale
- M_i transmet ses données collectées au M_j

endif

end while

Fin

III.4.3 Evaluation de LEACH

On a évalué LEACH dans deux modèles : le modèle UDG et le modèle LNS. Dans le modèle LNS, on a fixé le seuil de la probabilité d'une réception sans erreur à 0.6. De ce fait, si la probabilité de réception sans erreur entre deux nœuds est inférieure à 0.6 alors le message échangé entre eux est considéré comme perdu. Par analogie avec le modèle UDG, tout message dont la probabilité de réception est comprise entre 0.5 et 0.6 est considéré comme message perdu.

Nous proposons d'évaluer les métriques suivantes : le nombre de paquets reçus au niveau de la station de base, le taux de perte de paquets et l'énergie dissipée. Les paramètres de simulation sont représentés dans le tableau 1.

Tableau 1: Paramètres de simulation

Paramètre	Valeur
Temps d'une période	100 sec
Temps de formation des clusters	5 sec
Nombre de nœuds	100
Nombre de CHs	5
Temps de simulation	5, 10, 15, 20 périodes

a) Nombre de paquets reçus

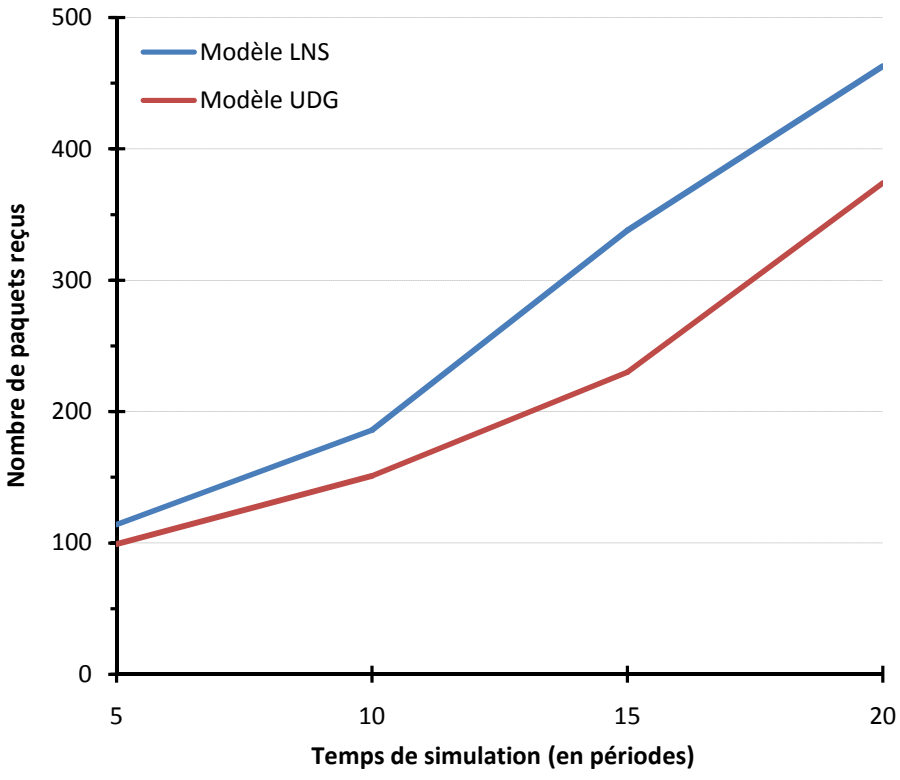

Figure 11 : Nb de paquets perdus vs. temps de simulation

La figure 11 montre que le nombre de paquets reçus dans le modèle LNS est supérieur que celui dans le modèle UDG puisque dans le modèle UDG tout paquet reçu dont la probabilité de réception sans erreur est comprise entre 0.5 et 0.6 est considéré comme paquet corrompu.

b) Taux de perte de paquets

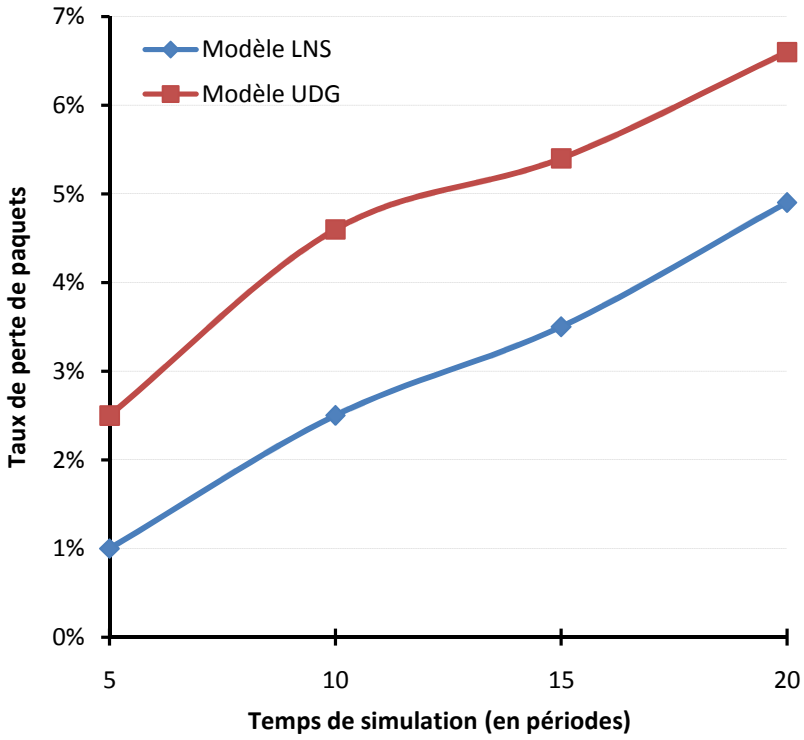

Figure 12: Taux de perte vs. temps de simulation

La figure 12 montre que la version améliorée fournit un taux de perte réduit comparativement avec la version originale de LEACH car dans la version améliorée si la probabilité est au dessous du seuil le nœud membre implique un autre qui est fiable comme relais.

c) Energie consommée

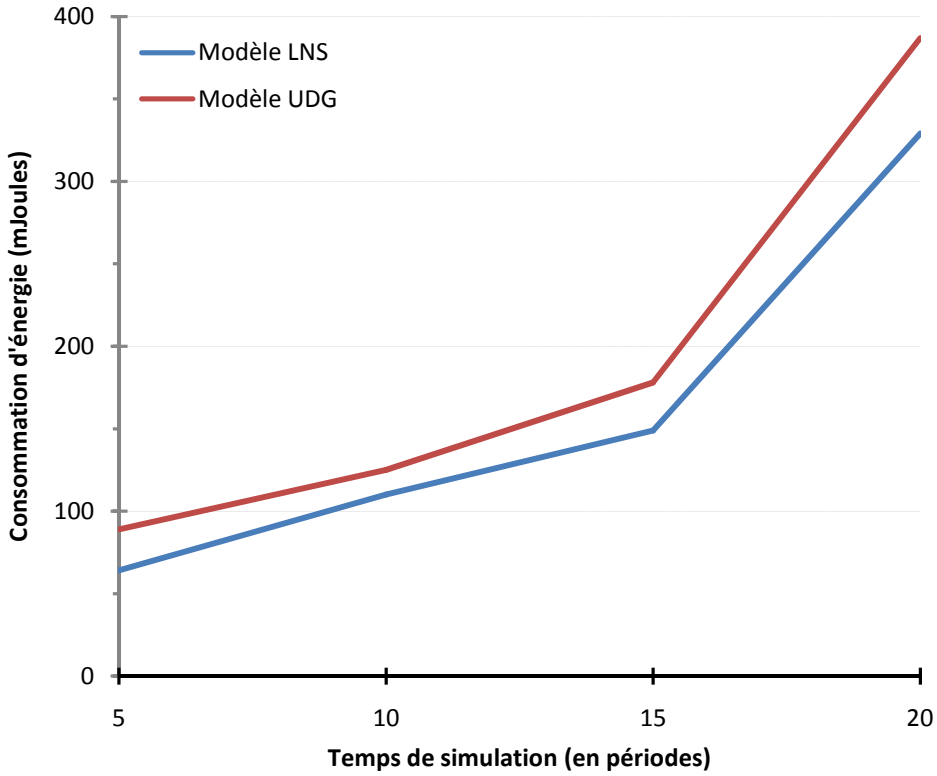

Figure 13: Consommation d'énergie

La figure 13 que la version améliorée de LEACH consomme moins d'énergie que la version originale car nous avons supposé que si le message est corrompu il y aurait une retransmission. Ceci augmente le nombre de paquets envoyés dans LEACH c'est-à-dire paquets reçus et paquets perdus.

III.5 Conclusion

Dans ce chapitre, nous avons proposé une version améliorée de LEACH qui s'adapte à un environnement non-idéal représenté par le modèle LogNormal Shadowing. Il prend en considération les fluctuations du signal radio en vue d'assurer une livraison fiable dans un environnement non-idéal où les liens peuvent être fiables ou bien non fiables tout dépend du degré d'atténuation dans l'environnement considéré. La version proposée a été évaluée par le modèle LogNormal Shadowing tel que la probabilité de réception sans erreur d'un paquet de données a été fixé à 0.6.

Les résultats de simulation ont montré que notre protocole peut s'adapter à un environnement réaliste puisqu'il assure la réception des paquets de données malgré la non fiabilité des liens et aussi, il consomme moins d'énergie par rapport aux protocoles LEACH et multihop-LEACH en minimisant le nombre de communications redondantes.

Conclusion générale

Conclusion générale

Les réseaux de capteurs sans fil ont connu un grand essor au cours de ces dernières années. Toutefois, la conception des applications liées aux RCSF doit faire face à certaines difficultés en raison de leurs capacités limitées et leurs ressources énergétiques très réduites qui constituent généralement un souci majeur au déploiement des réseaux de capteurs. L'objectif principal de projet de fin d'études était d'instaurer la tolérance aux pannes dans le protocole LEACH pour assurer la fiabilité du routage.

. Dans ce travail, nous avons évalué LEACH dans un environnement non-idéal représenté par le modèle lognormal shadowing. Puis on a proposé une version améliorée de LEACH qui s'adapte à ce type de modèle.

Références bibliographiques

Références bibliographiques

[1] Boukerche, R. W. N. Pazzi, and R. B. Araujo, “Fault-tolerant wireless sensor network routing protocols for the supervision of context-aware physical environments,” Journal of Parallel and Distributed Computing, vol. 66, no. 4, pp. 586–599, 2006.

[2] Y. Zeng, C. Sreenan, L. Sitanayah, N. Xiong, J. Park, and G. Zheng, “An emergency-adaptive routing scheme for wireless sensor networks for building fire hazard monitoring,” Journal of Sensors, vol. 11, no. 3, pp.2899–2919, 2011.

[3] O. Chipara, Z. He, G. Xing, Q. Chen, X. Wang, C. Lu, J. Stankovic, and T. Abdelzaher, “Real-time power-aware routing in wireless sensor networks,” in Proceedings of the 14th IEEE Workshop Quality of Service (IWQoS’06), pp. 83–92, 2006.

[4] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-efficient communication protocol for wireless microsensor networks,” in Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, pp. 1–10, January 2000.

[5] N. C. Brent, J. C. Charles, and S. J. David, “Unit disk graphs,” Discrete Mathematics, vol. 86, no. 1-3, pp. 165–177, 1990.

[6] T. S. Rappaport, “Wireless Communications: Principles and Practice”, Prentice Hall, 2002.

[7] C. Boano, J. Brown, Z. He, U. Roedig, and T. Voigt, “Low-Power radio communication in industrial outdoor deployments : The impact of weather conditions and ATEX-Compliance,” in Sensor Applications, Experimentation, and Logistics, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Ed. Springer Berlin Heidelberg, vol. 29, pp.159–176, 2010.

[8] C. Mettu. (2011), Telosb datasheet: https://fr.scribd.com/doc/68138250/Telosb-Datasheet-t

[9] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. “Wireless sensor networks: a survey”, Computer Networks (Elsevier), vol.38, no.4, pp.393-422, March 2002.

[10] K. Lin, J. Yu, J. Hsu, S. Zahedi, D. Lee, J. Friedman, A. Kansal, V. Raghunathan, and M. Srivastava, “Heliomote : Enabling long-lived sensor networks through solar energy harvesting,” in Proceedings of the 3rd ACM International Conference on Embedded Networked Sensor Systems (SenSys’05), pp.309–309, 2005.

[11] M. LEHSAINI, “Diffusion et Couverture basées sur le clustering dans les réseaux de capteurs sans fil”, Thèse de doctorat, Université de Tlemcen, Université de Franche-Comté, 2009.

[12] C. Jae-Hwan and L. Tassiulas, “Energy conserving routing in wireless ad-hoc networks,” in Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 1, pp.22–31, 2000.

[13] S. Soro and W. B. Heinzelman, “Prolonging the lifetime of wireless sensor networks via unequal clustering,” in Proceedings of the 19th IEEE International on Parallel and Distributed Processing Symposium., April 2005.

[14] D. Tian and N. D. Georganas, “A coverage-preserving node scheduling scheme for large wireless sensor networks,” in Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, pp.32–41, 2002.

[15] V. Raghunathan, C. Schurgers, P. Sung, and M. Srivastava, “Energy-aware wireless microsensor networks,” Signal Processing Magazine, IEEE, vol. 19, no. 2, pp. 40–50, March 2002.

[16] C.Y. Chong and S.P. Kumar. Sensor Networks: Evolution, Opportunities, and Challenges. In Proceedings of the IEEE, vol.91, no.8, pp. 1247-1256, 2003.

[17] T.B. Gosnell, J.M. Hall, C.L. Ham, D.A. Knapp, Z.M. Koenig, S.J. Luke, B.A. Pohl, A. Schach von Wittenau, and J.K. Wolford. Gamma-Ray Identification of Nuclear Weapon Materials. Technical Report, Lawrence Livermore National Lab., Livermore, CA (USA), February 1997.

[18] P. Johnson and D.C Andrews. “Remote continuous monitoring in the home”, Journal of Telemedicine and Telecare, vol.2, no.2, pp.107-113, June 1996.

[19] E.M. Petriu, N.D. Georganas, D.C. Petriu, D. Makrakis, and V.Z. Groza. “Sensor-based information appliances”, IEEE Instrumentation Measurement Magazine, vol.3, no.4, pp.31-35, December 2000.

[20] Eya Dhib. “Routage avec QoS temps réel dans les réseaux de capteurs”, Mémoire d’ingénieur, option : “ingénierie de réseaux”, Ecole Supérieure des Communications de Tunis, 2007.

[21] L. Paradis and Q. Han, “A survey of fault management in wireless sensor networks,” Journal of Netwoks Systems Management, vol. 15, no. 2, pp. 171–190, June 2007.

[22] J. Tateson, C. Roadknight, A. Gonzalez, T. Khan, S. Fitz, I. Henning, N. Boyd, C. J. Vincent, and I. Marshall, “Real world issues in deploying a wireless sensor network for oceanography,” in Proceedings of ACM Workshop on Real-World Wireless Sensor Networks, June 2005.

[23] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler, “An analysis of a large scale habitat monitoring application,” in Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, pp. 214–226, 2004.

[24] T. Schmid, H. Dubois-Ferrière, and M. Vetterli, “Sensorscope : Experiences with a wireless building monitoring sensor network,” in Proceedings of ACM Workshop on Real-World Wireless Sensor Networks, June 2005.

[25] K. Langendoen, A. Baggio, and O. Visser, “Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture,” in Proceedins of the 20th International on Parallel and Distributed Processing Symposium, April 2006.

[26] K. Martinez, P. Padhy, A. Riddoch, H. Ong, and J. Hart, “Glacial environment monitoring using sensor networks,” in Proceedings of ACM Workshop on Real-World Wireless Sensor Networks, June 2005.

[27] A. Ajay, N. Tarasia, S. Dash, S. Ray, and A. R. Swain, “Fault-tolerant multilevel routing protocol with sleep scheduling (FMS) for wireless sensor networks,” European Journal of Scientific Research, no. 1, pp. 97–108.

[28] A. Ajay, N. Tarasia, S. Dash, and S. S. A. Ray, “A dynamic fault tolerant routing protocol for prolonging the lifetime of wireless sensor networks,” International Journal of Computer Science and Information Technologies, vol. 2, no. 2, pp. 727–734, 2011.

[29] Z. Che-Aron, W. Al-Khateeb, and F. Anwar, “The enhanced fault-tolerance mechanism of AODV routing protocol for wireless sensor network,” International Journal of Computer Science and Network Security, vol. 10, no. 6, pp. 41–50, 2010.

[30] C. Perkins and E. Royer, “Ad-hoc on-demand distance vector routing,” in Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, pp.90–100, Feb 1999.

[31] F. Benhamida and Y. Challal, “FaT2D : fault-tolerant directed diffusion for wireless sensor networks,” in Proceedings of International Conference on Availability, Reliability, and Security, pp.112–118, February 2010.

[32] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed Diffusion: a scalable and robust communication paradigm for sensor networks,” in Proceedings of the 6th ACM Annual International Conference on Mobile Computing and Networking, pp.56–67, 2000.

[33] M. Lehsaini and C. Tabet Hellel, “A novel cluster-based fault-tolerant scheme for wireless sensor networks,” in Proceedings of the 24th IEEE International Conference on Microelectronics (ICM’2012), pp. 1–4, December 2012.

[34] M. Mehrani, J. Shanbehzadeh, A. Sarrafzadeh, S. Mirabedini, and C. Manford, “FEED : fault-tolerant, energy-efficient, distributed clustering for WSN,” in Proceedings of the 12th International Conference on Advanced Communication Technology, vol. 1, pp. 580–585, February 2010.

[35] M. Azharuddin, P. Kuila, and P. K. Jana, “Energy-efficient fault-tolerant clustering and routing algorithms for wireless sensor networks,” Journal of Computers and Electrical Engineering, vol. 41, no. 0, pp. 177–190, 2015.

[36] https://github.com/tinyos/nesc, Modified, June 2015

[37] P. Levis, N. Lee, M. Welsh, and D. Culler, “TOSSIM : accurate and scalable simulation of entire tinyos applications,” in Proceedings of the 1st ACM International Conference on Embedded Networked Sensor Systems, pp. 126–137, 2003.

[38] H. Alatrista, J. Mathieu, K. Gouaïch S. Aliaga, "Implémentation de protocoles sur une plateforme de réseaux de capteurs sans fil", Master 1 informatique, Université de Montpellier II, 29 Avril 2008.

[39] Wassim Znaidi, "Modélisation formelle de réseaux de capteurs à partir de TinyOS", Projet de fin d'études, Ecole Polytechnique de Tunisie, 2006.

[40] http://www.tinyos.net/dist-1.1.0/snapshot-1.1.1Nov2003cvs/doc/tutorial/lesson5.html, modified, October 2003.

Annexe

Annexe

A. Procédure d'installation sous Windows XP 

Ce guide propose l'installation du principal outil nécessaire au bon fonctionnement du système, notamment Cygwin (couche d'émulation de l'API Linux) qui permet d'avoir une interface Unix sous Windows. Cygwin est un environnement d'émulation Linux qui permet d'avoir un shell et de compiler et exécuter les programmes Linux (On dispose ainsi de gcc, apache, bash, etc.).

1- Télécharger le fichier tinyos-1.1.0-1is.exe de la source http://www.tinyos.net/dist-1.1.0/tinyos/windows/ .

2- Exécuter ce fichier pour installer la version 1.1.0 sous windows XP. L'installation se fait automatiquement. Un raccourci de Cygwin est sauvegardé sur le bureau.

3- Accéder à **C:\tinyos\cygwin\opt\tinyos-1.x\doc\tutorial\verifyhw.html** et suivre les étapes que contient cette page afin de vérifier si l'installation est bien réussie.

B. Installation de TinyViz

Les concepteurs développent au fur et à mesure l'outil TinyViz sans mettre à jour les fichiers sources déjà existants dans les anciennes versions. Cela ne permet pas de lancer TinyViz dans des conditions normales. Pour pouvoir le lancer, il est nécessaire de passe par **les étapes suivantes:**

1- Installer TinyOS-1.0

2- Accéder à: **cd /opt/tinyos-1.x/tools/java**

ET taper : **make**

3- Installer les mises à jour de NesC1.1.1 and TinyOS1.1.15.

Pour se faire, rechercher sur le net http://www.tinyos.net/dist-1.1.0/tinyos/windows/ ces mises à jour en téléchargeant le **rpm** et le mettant dans **C:\tinyos\cygwin\home\PLANETE PC**

Et taper dans le **shell**:

rpm -ivh --ignoreos nesc-1.1.2b-1.cygwin.i386.rpm

rpm -ivh --ignoreos --force tinyos-1.1.15Dec2005cvs-1.cygwin.noarch.rpm

4- Aller à **opt/tinyos-1.x/tools/java/net/tinyos/sim** et verifier si ces fichiers sont presents: **SimObjectGenerator.java** et **MoteSimObjectGenerator.java**

S'ils existent, alors les supprimer de ce répertoire.

5- Editer le **makefile** qui est dans **C:\tinyos\cygwin\opt\tinyos-1.x\tools\java\net\tinyos\sim** et écrire cette instruction : **net/tinyos/message/avrmote/*.class**

(Voir ** makefile pour vérifier là où il faut insérer cette instruction)

6- Aller à **shell** et taper:

cd /opt/tinyos-1.x/tools/java/net/tinyos/sim

make clean

make

7- Accéder à l'application qui va être simulée. On prend par exemple, l'application Blink.

Accéder au **shell** et faire:

cd opt/tinyos-1.x/apps/blink

make pc

 # tinyviz

export PATH="$TOSROOT/tools/java/net/tinyos/sim:$PATH"

TinyViz -run build/pc/main.exe 20 ///Insérer le nombre de noeuds. Par exemple 20

**** makefile:**

Voici le makefile à éditer. Dans le niveau indiqué (la ligne avec un fond coloré), insérer l'instruction net/tinyos/message/avrmote/*.class si elle n'existe pas:

```
  net/tinyos/sim/lossy/*.class            \
  net/tinyos/sim/msg/*.class              \
  net/tinyos/sim/packet/*.class           \
  net/tinyos/sim/plugins/*.class          \
  net/tinyos/sim/script/*.class           \
              net/tinyos/sim/script/reflect/*.class   \
              net/tinyos/sim/ui                       \
              net/tinyos/sim/plugins/plugins.list     \
              net/tinyos/sf/*.class                   \
              net/tinyos/util/*.class                 \
              net/tinyos/packet/*.class               \
              net/tinyos/message/*.class              \
    net/tinyos/message/avrmote/*.class      \
              org/apache/oro/text/regex/*.class       \
              org/python/compiler/*.class             \
              org/python/core/*.class                 \
              org/python/modules/*.class              \
              org/python/parser/*.class               \
              org/python/parser/ast/*.class           \
              org/python/rmi/*.class                  \
              org/python/util/*.class)
  rm -rf jarbuild-tmp
  mkdir jarbuild-tmp
  (cd jarbuild-tmp; jar xf ../simdriver-tmp.jar; jar xf ../$(ROOT)/jars/oalnf.jar; rm -rf META-
INF; jar cmf ../simdriver.manifest ../simdriver.jar .)
  rm -rf simdriver-tmp.jar jarbuild-tmp
jarclean:
  rm -f simdriver.jar
```

Résumé: Une version de LEACH adaptée à un environnement non-idéal représenté par le modèle lognormal shadowing

Les pannes sont inévitables dans les réseaux de capteurs sans fil, ceci peut influencer sur leurs performances. Pour illustrer l'impact de la non-fiabilité des liens radios, on a évalué LEACH dans le modèle lognormal shadowing. Les résultats de simulation ont montré que ce facteur a un impact négatif sur les performances de LEACH. Dans cette optique, nous avons proposé une version de LEACH qui prend en compte la qualité de lien dans le processus de routage. Les résultats obtenus ont montré l'apport de notre proposition en termes de taux de paquets perdus et de l'énergie consommée.

Mots clés : Le modèle du disque unitaire, LEACH, le modèle LogNormal Shadowing, TOSSIM, TinyOS.

Abstract: An adapted version of LEACH in non-ideal environment represented by Shadowing Lognormal model

Breakdowns are inevitable in wireless sensor networks, this can affect their performance. To illustrate the impact of unreliable radio links we evaluated LEACH in lognormal shadowing model. The simulation results showed that this factor has a negative impact on the performance of LEACH. In this context, we have proposed a version of LEACH that involves quality link in the routing process. The results obtained showed the contribution of our proposal in terms of packet loss rate and the energy consumed.

Keywords: Unit disk model, LEACH, Shadowing Lognormal model, TOSSIM, TinyOS.

ملخص:

الأعطاب لا مفر منها في شبكات الاستشعار اللاسلكية، وهذا يمكن أن يؤثر على أدائها .لتوضيح أثر الوصلات اللاسلكية الغير سليمة قمنا بتقييم في نموذج التظليل اللوغاريتمي .وأظهرت نتائج المحاكاة أن هذا العامل له تأثير سلبي على أداء LEACH.في هذا السياق، اقترحنا نسخة من LEACHالتي تعتبر صلة نوعية في عملية التوجيه .وأظهرت النتائج التي تم التوصل إليها مساهمة اقتراحنا من حيث الحزمة معدل الخسارة والطاقة المستهلكة.

الكلمات المفتاحية: نموذج القرص الوحدوي، LEACH, نموذج التظليل اللوغاريتمي ، TinyOS, TOSSIM