République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
UNIVERSITE DES SCIECES ET DE LA TECHNOLOGIE " HOUARI BOUMEDIENE"
FACULTE D'ELECTRONIQUE ET D'INFORMATIQUE

MEMOIRE
Présenté pour l'obtention du diplôme de MAGISTER
EN Informatique
Spécialité : Informatique Mobile

Par : LALOUANI WASSILA

SUJET

COORDINATION DANS LES RESEAUX DE CAPTEURS AVEC ACTIONNEURS

Soutenu le 07/07/2009, devant le jury composé de :

Mme- MOUSSAOUI	Maître De Conférence, USTHB, Présidente
Mr- BADACHE,	Professeur, USTHB, Directeur de Thèse
Mr- ABDELI,	Maître De Conférence, USTHB, Examinateur
Mme-GHARBI,	Maître De Conférence, USTHB, Examinatrice

RESUME

Les réseaux de capteurs WSN connaissent une large utilisation dans de nombreuses applications de surveillance de l'environnement. Pourtant, le besoin d'une interaction intelligente aux contraintes de l'environnement a conduit à l'émergence de réseaux de capteurs avec actionneurs (WSAN), qui sont non seulement capables de capter ces contraintes mais également de réagir.

Les WSAN ne sont toutefois pas eux-mêmes exempts de problèmes. Ainsi la cœxistence capteurs-actionneurs exclut tout recours aux protocoles déjà proposés dans les WSN. C'est pourquoi, de nombreux travaux ont été menés pour développer de nouvelles solutions adaptées aux nouvelles exigences en axant ses efforts sur la coordination, ou, autrement dit, la manière dont les nœuds gèrent leur propre réseau pour accomplir efficacement leur mission.

Deux types de coordination sont distinguées : capteur-actionneur et actionneur-actionneur et comme toutes les deux sont nécessaires, le développement d'un framework s'avère indispensable. Ce travail s'intéresse plus particulièrement à la première coordination appelée à répondre aux exigences et aux besoins applicatifs et environnementaux parmi lesquels le besoin d'énergie et de latence. En réponse aux problématiques posées dans ce cadre, plusieurs solutions [28][29] ont été proposées avec un intérêt particulier pour le regroupement des nœuds en fonction de leur déploiement initial et ce dans le but de former les clusters. A la réception des données, les actionneurs amorceront la coordination inter-actionneurs.

Dans les WSAN, le positionnement des capteurs dépend de l'application et des sphères à contrôler. Les actionneurs sont des nœuds qui doivent réagir rapidement en prenant les décisions les plus appropriées. Une solution consiste alors à les placer au bon endroit pour qu'ils puissent réceptionner l'information le plus tôt possible, tout en assurant la préservation des ressources énergétiques des capteurs. Le positionnement des actionneurs [56][60][65] constitue une préoccupation essentielle si l'on considère d'une part que leur emplacement affecte énormément la réalisation des exigences du système et se répercute sur les performances du réseau et que, d'autre part, il est souhaitable donc de disposer d'une couverture maximale d'actionneurs dans l'ensemble de la zone d'intérêt. En plus de la couverture, l'attention doit être concentrée sur le nombre de sauts (nœuds) qu'un paquet de données doit parcourir pour atteindre un actionneur.

La nature contraignante et les nouvelles exigences liées aux WSAN nous ont amenés à proposer un nouveau framework de coordination. La description complète de ce framework

s’appuie sur une taxonomie [71]. A la différence des frameworks développés dans les études antérieures, celui-ci s’inspire pour la réallocation des actionneurs des systèmes multi robots, ce qui rajoute un nouveau critère à la taxonomie de référence et permet de structurer convenablement le réseau, préalablement à toute opération de clustering. Ainsi, chaque actionneur dirige un cluster et est chargé de contrôler l’état du système en fonction du nombre de pannes, de celui des expirations et, enfin, du nombre de données perdues. Une présentation de notre framework renforcée par une simulation comparative nous permettra de confirmer l’efficacité de notre approche.

Table de matière :

Introduction Générale

Les réseaux de capteurs (WSNs) ont connu ces dernières années une large utilisation particulièrement dans le domaine de la surveillance de l'environnement grâce à leur efficacité dans la collecte d'informations provenant du monde physique et à leur capacité d'alerte. Pourtant, certains domaines d'applications ne peuvent se suffire de ces seules qualités et exigent une interaction intelligente aux contraintes de l'environnement. C'est dans ce contexte que l'on peut situer l'émergence d'une nouvelle classe de réseau, nommée Wireless Sensor and Actor Networks (WSANs), capable non seulement de capter les contraintes de l'environnement mais aussi de réagir [1]. L'architecture des WSANs comprend deux entités essentielles: les capteurs qui détectent les événements et les actionneurs qui prennent les décisions et exécutent les actions appropriées à travers l'environnement. Les WSAN présentent par rapport aux WSN, deux caractéristiques très importantes [1] : la condition de temps réel et la coordination. La condition de temps réel dépend de l'application car un besoin de réaction rapide peut s'exprimer suite à la capture. Par exemple, dans une application d'incendie, les actions sont à lancer sur la zone d'événement aussitôt que possible. Les données rassemblées et fournies par les capteurs doivent être encore valides au moment de l'action. En ce qui concerne la Coordination et à la différence des WSN où l'entité centrale (sink) assure la collecte des informations et la coordination, dans les WSANs, les phénomènes de gestion de réseau font appel à la coordination entre les capteurs et les actionneurs et entre les actionneurs eux mêmes. En particulier, la coordination capteurs-actionneurs assure la transmission des informations de l'événement des capteurs aux actionneurs. Apres réception de l'événement, un besoin de coordination inter-actionneurs se fait ressentir pour prendre une prise de décision et une réaction appropriées.

D'un autre côté, l'hetérogeniété du réseau c'est-à-dire la cœxistance des capteurs et des actionneurs déjà évoquée plus haut impose des contraintes plus strictes qui font que ces protocoles et algorithmes peuvent se révéler inadaptés pour les WSANs. Le problème le plus souvent abordé par la recherche est celui de la coordination ou comment les nœuds gèrent leur propre réseau pour accomplir leurs missions d'une manière efficace.

La problématique essentielle de la communication capteur-actionneur réside dans la réduction de la latence de communication eu égard à la proximité des capteurs et des actionneurs. Pour la coordination capteur-actionneur, il s'agira d'étudier les problèmes suivants [28]:

- A quelles conditions obéit cette communication ?
- Quels capteurs communiquent avec quels actionneurs ?
- Comment cette communication est elle réalisée ?

Plusieurs solutions ont été proposées en réponse à ces questions. La majeure partie d'entre elles tendent à regrouper les nœuds selon leurs déploiements initiaux. Les actionneurs entament un processus de découverte des voisins de façon à établir le contact avec l'ensemble des capteurs; cette phase sera utilisée pour la formation des clusters collaboratifs. Pour chaque cluster, il y a un actionneur désigné comme un cluster head qui rassemblera et traitera les données provenant des capteurs à l'intérieur de son cluster. Chaque capteur déterminera son prochain saut dans son propre cluster qui minimisera la latence et l'énergie. Les actionneurs ayant reçu les données d'événement entameront la phase de coordination inter-actionneurs pour répondre aux exigences des applications.

Dans les WSANs, le positionnement des capteurs dépend de l'application et des portions qu'ils doivent contrôler. Les actionneurs sont des nœuds qui doivent agir rapidement et prendre les décisions les plus appropriées. Une solution très intéressante consiste à placer ces actionneurs au bon endroit pour qu'ils puissent recevoir l'information le plus tôt possible, tout en préservant les ressources énergétiques des capteurs.

L'un des problèmes de conception dans les WSANs réside donc dans la répartition des actionneurs. L'intérêt que présentent les solutions de distribution des actionneurs est du au fait que leur emplacement affecte énormément la réalisation des exigences du système et se répercute sur les performances du réseau. Par exemple, dans de nombreuses configurations, il est souhaitable d'avoir une couverture maximale de l'ensemble de la zone d'intérêt. Par conséquent, une répartition uniforme des actionneurs dans toute la région peut être requise. En outre, dans de nombreuses applications, il est important de réagir rapidement aux nouveaux événements que ce soit par l'exécution des tâches spécifiques que par la réalisation d'expériences sophistiquées pour recueillir des données plus précises ou même de se

rapprocher de l'endroit où l'événement s'est produit. Ainsi, en plus de la couverture, l'attention doit être concentrée sur le nombre de sauts qu'un paquet de données doit parcourir jusqu'à atteindre un actionneur.

Dans ce travail, nous avons proposé un nouveau framework de coordination permettant de répondre aux nouvelles exigences des WSAN ainsi qu'à leur nature contraignante. Pour une description complète de ce framewok, nous nous référons à une taxonomie. A la différence des framework précédemment dans des études antérieures, Ce travail s'inspire des systèmes de réallocation des actionneurs pour minimiser la consommation d'énergie et de la latence, ce qui rajoute un nouveau critère à la taxonomie prise comme référence [71] et permet de bien organiser le réseau préalablement à toute opération de clustering. Chaque actionneur joue le rôle d'un cluster head et est chargé de contrôler l'état du système selon le nombre de pannes, le nombre des expirations et le nombre de données perdues. Les pannes sont dues essentiellement au manque d'énergie. Dans le but de prévenir les pannes, nous utilisons un schéma d'agrégation des données appropriées qui s'adapte à l'état du réseau.

Cinq chapitres structurent notre document. Le premier décrira l´état de l'art sur les WSAN. Le second et le troisième présenteront les deux framework de coordination qui constituent notre référence [28][29]. En chapitre 4, seront présenté les algorithmes de réallocation [56][60][65]. Enfin, le dernier chapitre nous permettra à travers une présentation de notre framework accompagnée d'une simulation comparative, de confirmer l'efficacité de notre approche.

Chapitre I : Les Réseaux de Capteurs avec actionneurs.

I.1 Introduction

Les réseaux de capteurs sans fil - Wireless Sensor Networks (WSN) - sont considérés comme un type spécial de réseaux ad hoc dont les nœuds consistent en un grand nombre de micro-capteurs capables de récolter et de transmettre des données environnementales d'une manière autonome. La position de ces nœuds n'est pas obligatoirement prédéterminée. Ils sont dispersés aléatoirement à travers une zone géographique, appelée champ de captage, qui définit le terrain d'intérêt pour le phénomène capté. Les données captées sont acheminées grâce à un routage multi-saut vers 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 en précisant le type de données requises ou récolter les données environnementales captées par le nœud puit.

Ces réseaux de capteurs sont largement utilisés dans de nombreuses applications pour la surveillance de l'environnement. Toutefois, le besoin d'une interaction intelligente aux contraintes de l'environnement a permis l'émergence d'une nouvelle classe de réseau, nommé Wireless Sensor and Actor Networks (WSANs), capable non seulement de capter les contraintes de l'environnement mais aussi de réagir. L'architecture des WSAN comprend deux entités essentielles: les capteurs qui détectent les événements, les actionneurs qui prennent les décisions et exécutent les actions appropriées à travers l'environnement. Trois nouvelles contraintes apparaissent :

- Les actionneurs doivent répondre rapidement à certains événements pour ne pas engendrer des dégâts et pour que l'événement soit encore valide au moment de la réaction.
- Le réseau est hétérogène.
- En l'absence de besoin d'intervention du sink (nœud de puits), la gestion du réseau revient aux capteurs et aux actionneurs.

Cependant, l'hetérogeniété du réseau c'est-à-dire la cœxistance des capteurs et des actionneurs déjà évoquée plus haut impose des contraintes plus strictes qui font que ces protocoles et algorithmes peuvent se révéler inadaptés pour les WSANs. Le problème le plus souvent abordé par la recherche est celui de la coordination où autrement dit comment les nœuds

gèrent leur propre réseau pour accomplir leurs missions d'une manière efficace. On distingue deux types de coordination : capteurs-actionneurs et actionneur-actionneur. La première permet d'envoyer l'information relative à l'événement à un ou plusieurs actionneurs qui doivent être le meilleur ensemble élu. La deuxième permet de déterminer les actionneurs qui vont interagir.

Le chapitre est structuré autour de trois grandes parties. La première comporte une introduction au WSAN, incluant une définition et une présentation de ses caractéristiques ainsi que les principaux axes de recherches. La deuxième partie introduit le problème de coordination et les différentes questions qui s'y rattachent. Les solutions et les protocoles déjà proposés sont introduits en 3eme partie.

I.2 Les réseaux de capteurs et d'actionneurs

Les réseaux WSAN se composent d'un ensemble de capteurs et d'actionneurs qui communiquent entre eux au moyen de liens sans fil pour exécuter la capture et réagir d'une manière distribuée. Ces dernières années ont marqué un intérêt croissant pour leur utilisation dans de nombreuses applications qui vont des soins de santé à la guerre. Le rôle des capteurs consiste à recueillir les informations relatives à l'environnement physique alors que celui des actionneurs concerne la prise de décision et le déclenchement d'actions appropriées.

Les WSAN sont hétérogènes ; en effet, les capteurs et les actionneurs présentent des caractéristiques très différentes : les capteurs sont de petite taille, peu onéreux, habituellement statiques avec des dispositifs à puissance de calcul limitée et une communication et des ressources énergétiques également de faible portée. Les actionneurs sont plus riches en ressources et ils sont généralement mobiles. Les capteurs peuvent être déployés par centaines voire par milliers. En revanche, le nombre d'actionneurs disponibles dans un même réseau WSAN est plus faible du fait de différences dans les conditions de déploiement et dans les missions à accomplir.

En règle générale, on s'attend à ce qu'un WSAN déployé fonctionne de façon autonome dans les environnements sans surveillance. Les exigences opérationnelles des WSANs peuvent évoluer en fonction de la mission du réseau définie dans un contexte multidimensionnel caractérisé par exemple par le champ de déploiement (hostile centre animal, par exemple), le type d'application (surveillance, cheminement, détection d'intrusion et réduction) et le mode de fonctionnement (normal, exceptionnel, événement).

Les réseaux WSAN présentent les caractéristiques uniques suivantes :

- Condition du temps réel : elle dépend de l'application car un besoin de réaction rapide peut s'exprimer suite à la capture. Par exemple, dans une application d'incendie, les actions sont à lancer sur la zone d'événement aussitôt que possible. Les données rassemblées et fournies par les capteurs doivent être encore valides au moment de l'action.

Ainsi, si les capteurs détectent un intrus malveillant dans un secteur, l'information doit être transmise aux actionneurs qui doivent intervenir pendant que l'intrus est toujours dans le même secteur. La question de la communication temps réel est vitale dans les WSANs.

- Coordination : à la différence des WSN où l'entité centrale (sink) assure la collecte des informations et la coordination, dans les WSANs, les phénomènes de gestion de réseau font appel à la coordination entre les capteurs et les actionneurs et entre les actionneurs eux mêmes. En particulier, la coordination capteurs-actionneurs assure la transmission des informations de l'événement des capteurs aux actionneurs. Apres réception de l'événement, le besoin des actionneurs de coordonner entre eux se fait ressentir pour prendre la décision la plus appropriée et réagir.

De nombreux protocoles et algorithmes pour les WSNs ont été proposés ces dernières années [1]. Toutefois, l'hetérogeniété du réseau c'est-à-dire la cœxistance des capteurs et des actionneurs déjà évoquée plus haut et les conditions énumérées ci-dessus imposent des contraintes plus strictes qui font que ces protocoles et algorithmes peuvent se révéler inadaptés pour les WSANs.

I.2.1 Caractéristiques physiques de WSAN :

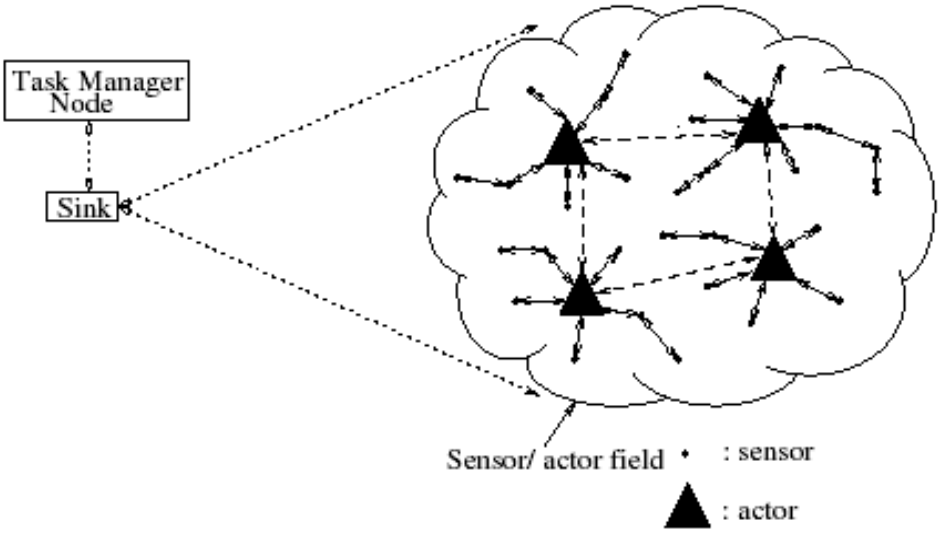


Figure 1.1 ARCHITECTURE PHYSIQUE DES WSANs.

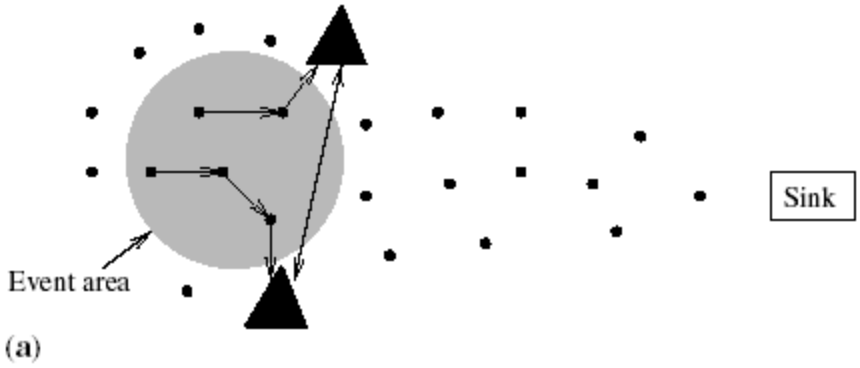


Figure 1.2. (a) architecture automatique.

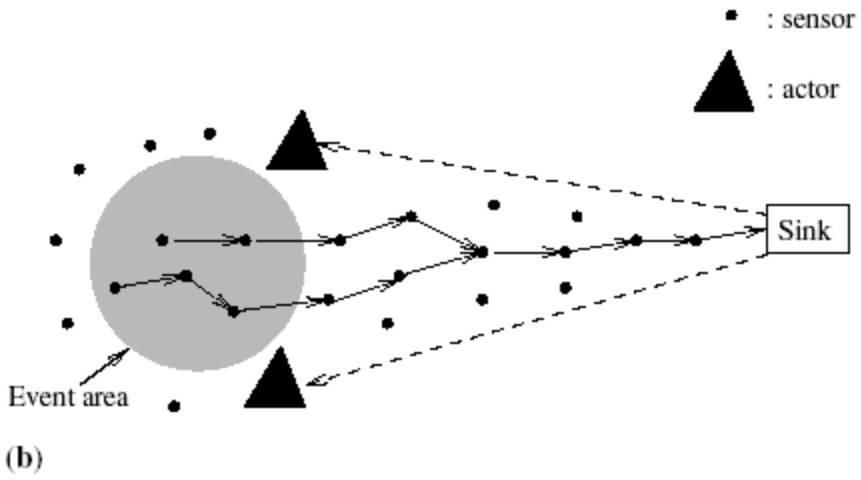


Figure 1.2. (b) architecture semi automatique.

Dans les WSANs, le rôle des capteurs consiste à rassembler les données de l'environnement, celui des actionneurs d'effectuer des actions appropriées sur la base des données rassemblées. Comme le montre la figure1.1, les nœuds sont dispersés dans la zone d'événement tandis que le sink surveille le réseau global et communique avec le nœud «task manager node » ainsi que les capteurs et les actionneurs. En détectant un phénomène, les capteurs peuvent soit transmettre leurs lectures aux actionneurs qui traitent les données entrantes et lancer les actions appropriées soit conduire de nouveau des données au sink qui peut émettre des commandes ou des requêtes aux actionneurs. Dans le premier cas, il s'agit d'architecture automatisée due à la non existence de l' unité centrale de traitement (interaction humaine par exemple). Dans le deuxième cas, c'est une architecture semi automatisée où le sink(unité centrale de traitement) rassemble des données et coordonne les actions nécessaires.

Ces architectures sont présentées dans la figure 1.2. L'une ou l'autre s'emploie selon les types d'application. L'avantage de l'architecture semi automatisée est qu'elle est semblable à celle déjà utilisée dans les applications sans fil de réseau de capteur [1]. Il n'y a ainsi nul besoin de développer de nouveaux algorithmes et protocoles pour exécuter la communication et la coordination.

La suite du chapitre est axée sur l'architecture automatisée, en raison d'une part de la **basse latence** et d'autre part de la **longévité** (durée de vie):

- Latence basse : l'information détectée est envoyée des capteurs aux actionneurs car ils sont proches l'un de l'autre (voir figure1.1). La latence est donc réduite au minimum dans l'architecture automatisée.

- Durée de vie : Comme indiqué dans la figure 1.2, dans l'architecture semi automatisée, partout où l'événement se produit, l'information d'événement traverse toujours les capteurs qui sont à moins d'un saut du sink. Ainsi, ces noeuds joueront le rôle de relais. Toutefois, la connectivité au sink peut se perdre et le réseau devient inutile en cas de panne ou de défaillance de la batterie des nœuds intermédiaires.

Bien que les techniques d'agrégation des données[1] diminuent la probabilité de ces occurrences, les capteurs autour du sink échouent (tombent en pane) toujours plus que les autres nœuds du réseau. De même, dans l'architecture automatisée, les nœuds à un saut des actionneurs peuvent être obligés de supporter une charge plus élevée de transmission des paquets par rapport aux autres noeuds. Cependant, il est quand même beaucoup plus probable dans ce cas que, pour chaque événement différent, les actionneurs de relais diffèrent. Ceci implique que les capteurs intermédiaires (ou de relais) seront également différents pour chaque événement. En d'autres termes, la charge de relais est plus au moins également distribuée entre tous les nœuds. En conséquence, l'architecture automatisée aura une durée de vie plus longue que l'architecture semi automatisée. D'ailleurs, dans l'architecture automatisée, l'information d'événement étant transmise localement par des capteurs autours de la zone d'événement, les capteurs éloignés de cette région ne fonctionnent pas comme des points de relais, ce qui induit une conservation des ressources du réseau (énergie et largeur de bande…).

I.2.2 Mode de fonctionnement dans les WSANs:

Le développement d'applications pour les WSANs est un challenge important au regard de la nécessité de prendre en compte la complexité de la distribution la décentralisation des opérations, les contraintes de ressources et la topologie dynamique. De plus, les applications doivent rester opérationnelles pour une longue période et elles doivent préserver leur intégrité autant que leur qualité de service.

Ces dernières années, le besoin d'investir les puissances de calcul s'est avéré indispensable pour abstraire la complexité du système. Pour cette raison, le traitement des requêtes (SNQP) [3]

[1] Fusionner les paquets de données.

connu un intérêt considérable et il est rapidement devenu un paradigme pour les réseaux de capteurs. Un autre type de technologie qui peut fournir un modèle pour les capteurs et les actionneurs suite à un événement détecté est représenté par ECA (event-condition-action).

Dans tous les cas, une application particulière opère dans l'un des modes :

- « Event driven application » (ECA) [4]: utilisé par exemple, pour la détection d'incendie, la gestion de la sécurité ou dans la détection de produit, le système restera inactif jusqu'à ce qu'un événement soit généré dans un nœud particulier pour se propager ensuite à tout le système qui, a son tour, initie les actions appropriées.

- « Demande driven application » (SNQP) [5]: utilisé dans le monitoring de l'environnement, par exemple, les activités y seront initiées en réponse à une requête externe.

Dans la première classe d'application, le système impose une réponse instantanée à l'événement. Ces applications sont possibles dans le second mode de fonctionnement sous réserve de vérifications régulières pour la détection de l'événement, ce qui conduira à une consommation de ressources plus élevée. ECA qui s'attache aux événements inattendus peut, en outre, fournir un paradigme naturel pour les événements réactifs. SNQP se concentre sur l'acquisition des données à partir d'un nombre limité de nœuds. Grâce au contrôle localisé et à sa concentration sur la réaction, et non sur le test d'occurrence de l'événement, ECA peut ainsi fournir un mécanisme effectif et efficace pour supporter les environnements réactifs.

Les deux systèmes présentent ainsi des profils différents d'exécution et, en conséquence, des besoins également différents. La différence entre les deux systèmes se caractérise ainsi:

- Le point de défaillance : SNQP assure l'initiation des requêtes, dans un ou plusieurs points de défaillance, et l'agrégation des données par des nœuds intermédiaires nommés « points de stockage ». Dans les ECA, l'événement peut être généré dans n'importe quel nœud du réseau qui peut être utilisé par n'importe quel actionneur, quel que soit son emplacement dans le réseau.

- Le modèle de communication : SNQP collecte les données sur la base d'un modèle régulier où un capteur assure la synchronisation et gère le cycle « wake up » ou « sleep ». ECA est réactif et imprévisible et son intervalle est irrégulier. Donc, le schéma « wake up » et « sleep » qui peut supporter le mode asynchrone d'opération s'avère nécessaire. De plus, ce modèle irrégulier implique que les nœuds consomment leur puissance durant

des périodes différentes et ainsi les nœuds défaillants sont irréguliers et plus difficilement prévisibles.

- Le routage : les SNQPs utilisent généralement un mécanisme de routage basé sur les arbres qui inonde le réseau au moins une fois pendant l'étape de construction de l'arbre (étape de découverte d'itinéraire). Ainsi, chaque collecte de données nécessite une découverte de routes qui engendre un coût de communication élevé. Dans ECA, il s'agit de petites mises à jour et non de collections de données. L'étape de découverte de route ne sera utilisée que pour le routage, ce qui induit des coûts très élevés de communication. L'étude [2] montre que les algorithmes basés sur la localisation s'avèrent plus adéquats pour ECA.

- Le modèle de données: si SNQP ne nécessite pas de distinction de données, ECA doit, au contraire, spécifier les types d'événements observés et générés.

- L'agrégation : elle est effectuée à la réception d'un signal pour ECA et à la suite d'une requête pour SNQP. Bien que les techniques mathématiques utilisées pour l'agrégation dans SNQP [3] puissent également être employées dans ECA, cette opération s'effectue dans une couche inférieure par le biais d'une approche qui s'apparente au traitement des signaux de collaboration dans les environnements distribués.

- Le stockage : même si les deux systèmes bénéficient des ressources de stockage, SNQP développe des mécanismes hiérarchiques et directionnels basés sur le protocole de routage alors que ECA bénéficie de la décentralisation.

- Le partitionnement du réseau : ECA s'exécute à travers une localisation spécifique et résiste à la segmentation qui peut résulter, par exemple, de la perte de connectivité des nœuds défaillants. ECA peut mieux résister à l'isolation par rapport au sink.

I.2.3 Les axes de recherche :

- La coordination capteur-actionneur : dans les WSANs, plusieurs actionneurs peuvent recevoir l'information à partir des capteurs, c'est ce qui est nommé « multi-acuator » ou MA. « Single actuator » ou SA est le fait qu'un seul actionneur reçoit l'information à la place de plusieurs. SA est donc un cas particulier de MA. Les pistes suivantes sont préconisées :

    - s'assurer qu'il n'y a aucun effet adverse dans l'environnement.

- Assurer la synchronisation entre les actionneurs au moment de l'envoi de l'événement capté.
- Envoyer l'information à un sous ensemble d'actionneurs couvrant la région d'événement dans les MA.
- Comparer SA et MA pour pouvoir choisir celui qui convient à une application particulière.

● Coordination des actionneurs : pour résoudre un problème particulier, les actionneurs communiquent entre eux de façon centralisée ou décentralisée. La coordination répond aux exigences suivantes :

- Dans le cas d'un seul actionneur, comment le sélectionner parmi plusieurs et comment trouver le nombre optimal d'actionneurs.
- La nécessité d'un modèle de communication entre les actionneurs.
- La nécessité d'exécution des différentes réactions suite aux événements détectés dans la région, de manière à s'assurer qu'il n'y a pas d'effet adverse dans l'environnement.
- La synchronisation entre les actionneurs pour une réaction simultanée à un même événement dans certaines applications.
- Le besoin de spécifier le contenu des messages et des algorithmes pour fournir une transmission efficace des données, quel que soit le type du message.
- La méthode de sélection de l'actionneur qui prend les décisions.
- Les protocoles de coordination et de communication qui doivent supporter les propriétés temps réel.

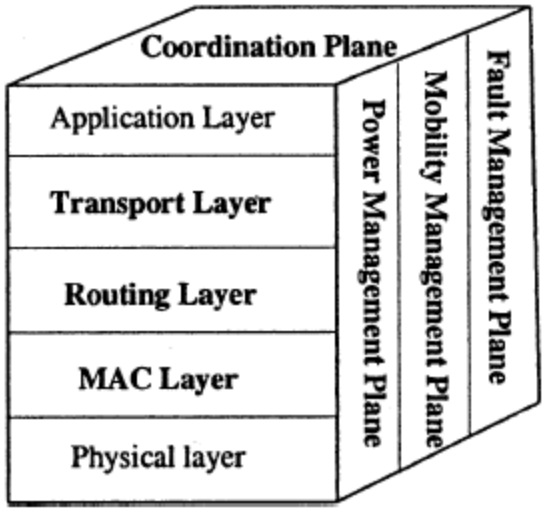


Figure 1.3 la pile des protocoles dans les WSAN.

- La couche transport : les nouveaux protocoles de transport doivent supporter la contrainte temps réel. Plusieurs protocoles de transport ont été développés pour les réseaux ad hoc et les réseaux de capteurs. Cependant, il n'existe pas un protocole qui englobe la fiabilité et la contrainte du temps réel. Comme la communication capteur-actionneur et actionneur-actionneur sont apparues successivement, un seul protocole commun est nécessaire.

- Le routage : Dans les WSANs, quand un événement apparaît, il n'y a pas un actionneur spécifique pour chaque message envoyé. Cette incertitude, due à l'existence de plusieurs actionneurs, pose un problème au niveau du routage. Sélectionner un actionneur est un challenge pour le nœud source. De plus, afin de déterminer le chemin élu et les données à envoyer, le protocole de routage doit supporter la communication en temps réel en considérant les différents délais (deadlines) dus aux différents intervalles de validité. Il doit en outre prendre en compte les priorités et mettre le moins de latence possible. Durant ces dernières années, le problème du routage a été traité dans de nombreux travaux :

    - An anycast [6] il ne supporte pas la coordination capteur-capteur à cause des informations corrélés de multiples capteurs qui détectent le même événement. Par ailleurs, ce mécanisme impose au capteur à un saut de l'actionneur de recevoir

les besoins a partir d'autres actionneurs situés de l'autre coté du réseau, ce qui peut provoquer un trafic inutile.

o SEAD[7] : inadéquat pour les WSANs car il ne prend pas en compte la minimisation de la latence qui leur est essentielle. De plus, il est développé dans le cas où tous les sink demandent l'information de la même source durant la période « refresh »[2]. Dans les WSAN, seuls les actionneurs, proches de la zone d'événement, sont intéressés par l'événement.

o SPEED[8] : il est adaptatif et basé sur la localisation. C'est un protocole temps réel utilisable dés lors que tous les capteurs connaissent leur position. Cependant, il est inadéquat pour les WSAN car il ne supporte pas le type de communication MA et la mobilité des actionneurs.

o Cluster : c'est un modèle où les membres du groupe ont des ressources limitées alors que les clusters head présentent une énergie élevée [9]. Il peut être utilisé pour les WSANs car les actionneurs peuvent être vus comme des clusters head et chaque capteur peut être considéré comme un membre du cluster. De nombreuses questions se posent :

-comment le cluster sera formé ? à base d'événement, par exemple.

-comment le cluster sera adapté à la mobilité ?

-comment le cluster satisfait la contrainte du temps réel ?

Pour la communication actionneur actionneur, les protocoles de routage DSR [10], AODV [11], OLSR [12] et DSDV [13] peuvent être utilisés si la contrainte temps réel est satisfaite.

- Couche de contrôle d'accès aux medias : pour transmettre l'information d'événement à travers un grand nombre de capteurs aux actionneurs, il est indispensable de passer par la couche MAC. Cependant, dans certaines applications, les actionneurs peuvent être mobiles. En fonction de leurs mouvements, ils peuvent partir d'une zone de transmission de certains capteurs pour rejoindre une autre ou se déconnecter totalement du réseau. Une fonction additionnelle consiste alors dans le maintien de la connectivité entre les capteurs et les actionneurs. Le temps de détection, de traitement et de délivrance

[2] Période de transmission

de l'information est primordial dans les applications WSANs. Les protocoles classiques basés sur la contention ne sont pas appropriés pour la communication en temps réel entre les capteurs et les actionneurs car la contention nécessite le handshaking qui augmente la latence des données. **Trace** [14] est un protocole TDMA qui souffre de l'overhead pour la réservation de la contention alors que **PBP 802.11 (Predictive Backoff Protocol for IEEE 802.11)** [15] nécessite beaucoup d'énergie du fait que tous le capteurs écoutent les autres transmissions. **Collision free** [16] est un autre protocole qui peut être adéquat pour les WSAN car il réduit le délai de délivrance des données et assure la propriété temps réel ainsi que la puissance d'énergie en éliminant les collisions. Le problème de ce protocole est qu'il utilise plusieurs canaux. Généralement, les protocoles existants n'investissent pas la mobilité. Pour la communication actionneur-actionneur, les protocoles MAC existants pour les WSAN et pour les réseaux ad hoc ne sont pas directement appliqués car ils doivent supporter la contrainte temps réel.

- Cross layering : les protocoles pour les WSN et les WSAN sont basés sur la notion de couche. Cependant, l'inflexibilité de ce paradigme induit des performances limitées pour WSANs à cause de la limitation d'énergie et de la latence, d'où le besoin pour les couches de s'investir entre elles.

- Développement : De nombreux travaux ont été consacrés aux développements des capteurs et des dispositifs de communication. Ces nouveaux dispositifs nécessitent la présence d'une plateforme de logiciel qui les impulse et les relie. TinyOS est conçu pour remplir ce rôle [17]. Les logiciels d'exploitation temps réel courants ne satisfont pas les besoins actuels des réseaux de capteurs. Une question architecturale importante dans la conception des réseaux de capteurs est de savoir si différents microcontrôleurs doivent être employés pour contrôler chaque unité d'E/S. Il est possible de maintenir plusieurs flots de données avec un microcontrôleur simple. Ceci prouve que c'est une option architecturale - pas une condition - d'utiliser différents microcontrôleurs par dispositif. D'ailleurs, l'interconnexion d'un tel système devra soutenir un modèle efficace de communication basé sur les événements. Il y a besoin d''équilibrer entre la puissance d'énergie, la vitesse de communication, la flexibilité et la fonctionnalité requises [17] [18].

Bien qu'il y ait eu beaucoup de travaux en développement et en déploiement des systèmes embarqués, les WSAN changent la nature du problème à cause des limitations d'énergie et de la bande passante. Les réseaux de capteurs ont constitué eux mêmes un domaine de

recherche très actif ces dernières années, mais la majorité des travaux se sont concentrés sur l'aspect de la sensation dans les WSNs et sur la mise en action dans le WSANs. Les réseaux de capteurs ont été déployés avec succès pour la seule surveillance de l'environnement et très peu pour sa transformation [19]. Finalement, les WSANs peuvent fournir la capacité à surveiller sans interruption l'intégrité des structures en temps réel, détecter les dommages à une partie, et assurer la robustesse dans le cas des échecs catastrophiques. Cependant, les WSANs exigent un nouveau paradigme de traitement qui prend en compte les ressources matérielles limitées, la corruption des liaisons de communication, la largeur de bande limitée et les contraintes d'énergie [20].

I.3 La coordination :

I.3.1 Capteur-actionneurs :

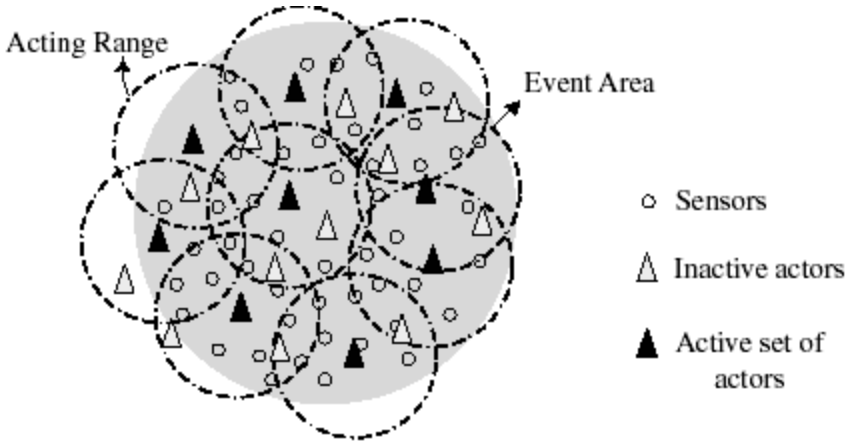


Figure 1.4 la construction des cluster

La problématique essentielle de la communication capteur actionneur réside dans la réduction de la latence de communication du fait de la proximité des capteurs et des actionneurs. Pour la coordination capteur-actionneur, il s'agira d'étudier les problèmes suivants :

- A quelles conditions obéit cette communication ?
- Quels capteurs communiquent avec quels actionneurs ?
- Comment cette communication est elle réalisée ?

En premier lieu, la communication capteur-actionneur doit consommer le minimum d'énergie comme dans WSN. Dans certaines applications comme les incendies, le trafic de communication est sensible aux retards et donc une condition principale de communication des capteurs et des actionneurs consiste à soutenir le trafic en temps réel.

Une autre condition pour cette communication dans WSAN repose sur la nécessité d'assurer l'ordre des événements fournis aux actionneurs. Par exemple, si deux capteurs rapportent deux événements différents à un ou plusieurs actionneurs, le rapport de ces événements doit suivre l'ordre de leur apparition, de manière à garantir une réaction correcte.

Une considération importante supplémentaire est la suivante : s'il y a multitude de capteurs qui rapportent un événement et que l'information des différents capteurs peut arriver aux actionneurs intéressés au même moment, il faudra nécessairement que l'action se fasse en une fois et dans toute la région d'événement.

S'il est souhaitable que cette synchronisation d'exécution se fasse par la coordination d'actionneur actionneur, il est également imaginable qu'elle soit permise par les capteurs. Dans quelques applications où l'événement a eu lieu dans des endroits différents, il peut s'avérer nécessaire que les événements soient communiqués à l'ensemble des actionneurs et pas spécialement à ceux qui sont les plus proches de la zone d'événement, lors de la détection de l'événement. Par contre, le rapport doit être envoyé aux actionneurs les plus proches pour qu'ils puissent réagir rapidement.

Dans ce cas, les capteurs doivent être en mesure de dépister l'événement et d'utiliser cette information pour déterminer l'ensemble des actionneurs.

En conséquence, de nouveaux protocoles doivent être développés pour WSAN avec comme objectif de :

- Fournir des services en temps réel en respectant les délais exigés par les applications.
- Assurer une communication entre les capteurs et les actionneurs avec un minimum d'énergie.
- Respecter l'ordre des différents événements quand ils sont rapportés aux actionneurs.

- Fournir la synchronisation des différents capteurs rapportant le même événement au multiple ou au même actionneur afin de ne délivrer qu'une seule réponse pour la région entière.

- Rapporter les phénomènes observés à un ensemble différent d'actionneurs pas nécessairement basés sur les limitations de proximité ou d'énergie, pour le cas où les événements ont lieu dans des endroits différents.

Ces protocoles doivent non seulement satisfaire les conditions précédemment évoquées de la communication capteur actionneur mais également traiter les sources et les destinations impliquées dans la transmission des données captées. Pour ces derniers, 4 solutions existent :

- Ensemble minimal d'actionneurs pour couvrir la région d'événement
- Nombre minimum de capteurs pour rapporter l'événement capté
- Les deux cas ci-dessus
- L'ensemble entier d'actionneurs et de capteurs à proximité de la région.

Les trois premières classifications serviraient à l'élimination de la redondance dans les WSAN. Il s'agit de réduire au minimum la consommation moyenne de puissance pour l'ensemble des capteurs et des actionneurs qui sont à proximité de l'événement. Par exemple, comme indiqué dans la figure 1.4, si le minimum d'actionneurs pour couvrir la zone d'événement est 9 et s'il y a 20 actionneurs dans la région, les 11 actionneurs restants ne réagissent pas à l'événement. Dans cet exemple, il est préférable qu'un ensemble minimal de capteur détecte l'événement et le rapporte. Dans ce cas, il convient d'utiliser le minimum de capteurs et d'actionneurs pour couvrir l'événement. D'un autre coté, dans d'autres applications, différentes de la classification précédente, apparaît le besoin de la redondance (c'est-à-dire que le nombre de capteurs et d'actionneurs se situe entre la valeur minimale et maximale du préférable).

Egalement, une condition plus stricte pour certaines applications pourrait être aussi bien constituée par la sélection d'un ensemble minimal d'actionneurs que par un ensemble d'actionneurs dont les zones ne se chevauchent pas.

Pour le type de transmission, il existe deux possibilités: un seul saut ou multi sauts. Cependant, la communication à un seul saut n'est pas efficace pour les WSN du fait la longue distance entre les capteurs et le sink. Dans les WSAN, cela peut ne pas être le cas car les

actionneurs sont très proches des capteurs comme le montre la figure1.4. Si la zone d'événement est petite et qu'il y a un actionneur au milieu de la zone, donc les nœuds situés loin de l'actionneur consomment peu d'énergie, alors que, dans une grande zone d'événement ou l'actionneur est en dehors de la zone, multi saut est plus efficace à cause de la distance entre le capteur et l'actionneur. Ainsi, le type de transmission dépend du déploiement et de la localisation des actionneurs auxquels les capteurs vont envoyer leurs données.

En plus du type de transmission et des conditions de communications entre les capteurs et les actionneurs, comme déjà indiqué ci-dessus dans la deuxième question, se pose le problème de la sélection où deux cas, Multi-Acteur (mA) et Simple-Acteur (SA) seront décrits en particulier pour montrer comment les capteurs peuvent choisir les actionneurs auxquels ils communiqueront leurs données.

1.3.1.1 Sélection des actionneurs :

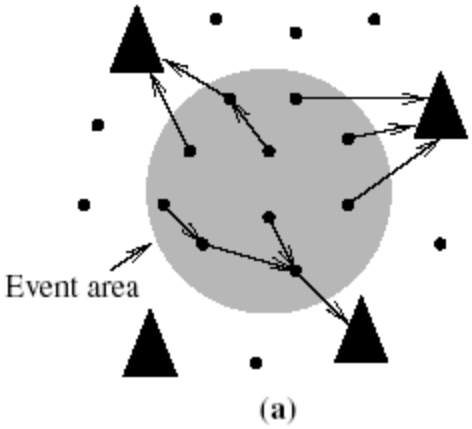


Figure 1.5 (a) multi actionneur (MA)

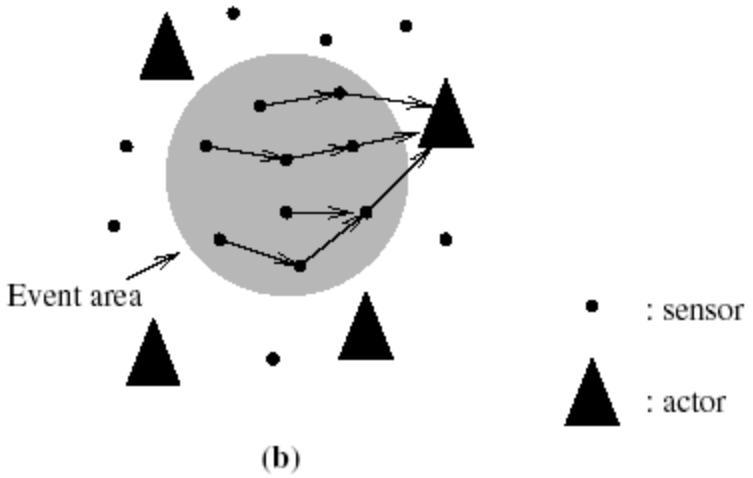


Figure 1.5 (b) simple actionneur (SA)

Suivant les indications de 1.5 (a), plusieurs actionneurs peuvent recevoir l'information captée. Ce cas est appelé Multi-Actionneur (MA) où chaque capteur peut décider en toute indépendance à quel actionneur il enverra ses lectures. Cependant, l'inconvénient du manque de coordination entre les capteurs est que trop d'actionneurs inutiles peuvent être activés et en conséquence la consommation totale d'énergie de tout le réseau peut devenir élevée. Pour éviter cette situation, les capteurs devraient se coordonner les uns avec les autres pour former des clusters. Pour chaque cluster, il y aura un actionneur pour rassembler les données.

Ces clusters peuvent être formés de la manière suivante:

- Le temps de transmission de l'événement capté aux actionneurs doit être réduit au minimum puisque une réaction rapide est exigée dans un tel système.
- les voies de communication entre les capteurs et les actionneurs seront choisies parmi celles qui consomment le moins d'énergie.
- L'ensemble des actionneurs doit couvrir la région entière d'événement.

À la différence de 1.5 (a) où les informations captées sont envoyées à plusieurs actionneurs, dans la figure 1.5 (b), seul un actionneur reçoit les informations. Ce cas est appelé Simple-Actionneur (SA). En fait, SA peut être considéré comme cas spécial de MA. Dans SA, un des défis principaux est de déterminer l'actionneur auquel les capteurs enverront leurs lectures. Le choix de l'actionneur peut être basé sur certains critères comme :

- la distance entre la zone d'événement et l'actionneur devrait être minime de sorte qu'il n'y ait que de faibles retards réalisés et que moins de puissance soit consommée,
- l'actionneur choisi sera celui dont le chemin de liaison avec le capteur consomme moins d'énergie,
- L'actionneur choisi doit être en mesure d'accomplir les actions attendues.

A noter que dans le dernier cas (SA), il n'y a aucune garantie que la réaction de l'actionneur choisi peut couvrir le domaine entier d'événement. Par conséquent, au lieu de considérer les questions de distance, d'énergie ou de synchronisation, les capteurs doivent s'efforcer de choisir le meilleur actionneur pour cet événement, c'est à dire, celui qui dispose de l'assurance, de l'énergie et des possibilités de mouvement pour effectuer l'action requise sur la zone d'événement.

Dans SA, l'actionneur peut immédiatement exécuter l'action pour peu qu'il en ait la capacité et l'énergie et que celle-ci puisse être assurée par un simple actionneur. La latence entre la sensation et l'action est dans ce cas basse.

Cependant, si un actionneur n'est pas compétent pour l'action demandée ou s'il n'est pas adapté aux contraintes d'assurance et d'énergie que nécessite cette action, il édite un message d'annonce à d'autres actionneurs. Cette rétroaction d'autres actionneurs permet ainsi de choisir un ou plusieurs actionneurs pour réaliser les actions appropriées.

L'intensité des événements peut ne pas être uniforme à l'intérieur de la zone d'événement. En conséquence, l'intensité du signal capté peut être différente d'un actionneur à un autre. Ces derniers peuvent comparer leurs signaux reçus, les valeurs des intensités les unes aux autres et déterminer où l'intensité d'événement est plus dense, ce qui peut avoir pour conséquence des actions plus efficaces du fait du déplacement des actionneurs mobiles vers le centre de l'événement.

L'inconvénient du MA est que la coordination entre les actionneurs est basée sur la négociation parmi des actionneurs multiples, à la différence du message d'annonce dans SA. Dans MA, chaque actionneur possède quelques informations partielles sur l'événement global et

ainsi, afin de prendre les décisions d'action appropriées, les actionneurs doivent se coordonner entre eux, ce qui peut engendrer des coûts de communication élevés et une haute latence.

I.3.1.2 les axes de recherche

Il est possible de résumer les problèmes de la recherche liés à la coordination dans MA ou SA comme suit :

- pour MA et SA, la livraison dans l'ordre des événements détectés dans une région peut être exigée de manière à s'assurer qu'il n'y a aucun effet nuisible sur l'environnement ciblé.

- Aussi bien en SA qu'en MA, il peut être exigé une synchronisation dans la période de communication des phénomènes sentis entre différents actionneurs responsables des réactions.

- Dans certaines applications où les événements se produisent dans différents endroits, il peut être nécessaire que l'information sentie soit envoyée à un actionneur ou à un ensemble d'actionneurs déterminés en se basant sur leur emplacement.

- En MA, et dans le but d'économiser l'énergie moyenne consommée par les actionneurs lorsqu'ils communiquent l'information captée, il peut parfois s'avérer nécessaire de transmettre par les capteurs l'information à plusieurs actionneurs. Dans ce cas, cette information n'est communiquée qu'à un sous ensemble d'actionneurs qui couvrent toute la région d'événement.

I.3.2 Actionneur-actionneur

Comme indiqué dans la section 1, les actionneurs communiquent entre eux, en plus de communiquer avec les capteurs. La communication d'Actionneur-actionneur se produit dans les situations suivantes :

- Du fait de son incapacité à agir et d'une énergie insuffisante, l'actionneur recevant des données captées peut ne pas réagir à l'événement.

- L'actionneur n'est pas prêt à effectuer l'action demandée et donc d'autres actionneurs voisins doivent être déclenchés.

- Des actionneurs multiples reçoivent la même information d'événement alors qu'un seul doit réagir. La communication entre eux permet de décider lequel interviendra.

- Lorsque des actionneurs multiples sont requis pour couvrir la région d'événement entière, il peut s'avérer nécessaire de s'assurer que ces régions ne se chevauchent pas, de manière à garantir un comportement uniforme au dessus de la région entière.
- Si des actionneurs multiples reçoivent l'information à partir de plusieurs capteurs pour un même événement, il peut être nécessaire de s'assurer que ces actionneurs agissent simultanément sur l'événement. Cette condition de synchronisation dans l'exécution de la réaction est exigée dans les applications où une exécution partielle des actions peut altérer l'état de l'événement dans la région où la réaction n'est pas exécutée.

- En cas d'événements multiples simultanés, la charge peut être répartie par l'intermédiaire d'une communication actionneur-actionneur. Elle peut, en outre, être exécutée de façon séquentielle. Cette contrainte est considérée comme un ordonnancement de l'exécution des charges.

- Apres qu'un actionneur reçoit l'information d'événement, et si l'événement s'élargit à d'autres zones, il peut communiquer les données captées ou les commandes d'action aux autres actionneurs situés dans ces zones. L'envoi de cette information sera ainsi assuré par l'ensemble des actionneurs initiaux, à l'exclusion de tout capteur.

Les situations décrites ci-dessus montrent la nécessité de la coordination actionneur-actionneur et convergent vers la question suivante : « quels actionneurs pour exécuter quelles missions ? ». La réponse peut reposer sur l'exploitation de la forte coordination entre les actionneurs afin de maximiser la performance [21]. Dans les WSANs, la charge constitue les actions nécessaires à effectuer lors de l'occurrence d'un événement.

La question peut donc être reformulée ainsi : « comment faire assurer la charge par de multiples actionneurs ».

Ce problème de répartition de charge dans les WSANs peut être examiné en utilisant les deux axes suivants :

- **Single actor task (SAT) vs. multi actor task (MAT):** SAT exige que chaque tâche soit effectuée par exactement un seul actionneur, alors que dans MAT chaque tâche nécessite plusieurs actionneurs. Donc, MAT pose le problème de la nécessité de combiner les efforts des actionneurs.

- **Centralised decision (CD) vs. Distributed decision (DD) :** dans les WSANs, il y a nécessité de prise de décision quant à l'action à exécuter en fonction de l'événement. Il peut s'agir d'une décision centralisée ou d'une décision distribuée (double densité). La DD permet aux actionneurs voisins de coordonner localement, ce qui fournit des réactions en temps réel et une coordination indépendante de la taille du réseau. Le CD aboutit à une prise de décisions d'action organisée car la décision est prise par un nœud qui peut être équipé d'équipements de communication plus puissants.

L'affectation des tâches aux actionneurs en exploitant les cas précédents constituera le propos de la section I.3.2.1. En section I.3.2.2, seront présentés les challenges de la recherche concernant MAT, SAT, CD et DD.

I.3.2.1 affectation des tâches

Dans le cas du MAT, si plusieurs actionneurs reçoivent des informations captées, ils négocient entre eux et coordonnent localement pour choisir l'actionneur le plus indiqué pour la tâche spécifiée. Dans le cas de CD, ils transmettent directement les caractéristiques de l'événement (endroit, intensité…) au noeud prédéterminé qui fonctionne comme centre de décision. Ce centre de décision, qui dispose déjà d'informations sur les actionneurs dans le réseau, choisit les meilleurs actionneurs pour la charge et les déclenche pour lancer l'action. Ces actionneurs choisis (tous deux DD ou CD) peuvent ne pas être ceux qui ont reçu des données captées par l'intermédiaire de la coordination capteur-actionneur, car les actionneurs qui reçoivent l'information d'événement peuvent ne pas être les mieux indiqués pour exécuter les

actions, soit par exemple parce qu'ils sont pas assez proches du secteur d'événement soit parce qu'ils sont incapables de prendre en charge l'événement.

Dans la MAT., si un seul actionneur (SA) reçoit l'information d'événement à la fin de la phase de coordination capteur-actionneur, il y a toujours un besoin de coordination parmi des acteurs dans le but de déterminer ceux, qui agiront et dans quelle région s'exécuteront les actions. Toutefois, dans ce cas, toutes les données captées sont rassemblées autour d'un actionneur qui peut fonctionner comme unité centrale de décision. Il y a alors un message d'annonce à d'autres actionneurs qui contient les détails relatifs à l'événement et aux actions à exécuter. Sur la base de la rétroaction d'autres actionneurs, la charge est assignée aux meilleurs actionneurs. Une fois l'action de charge assignée, chaque actionneur est investi d'une mission à l'intérieur de sa zone d'action. Cependant, pour réagir à chaque donnée représentant un phénomène qui s'est produit à l'intérieur d'une zone, l'union des réactions reparties doit couvrir l'ensemble du domaine d'événement.

De plus, quelques zones d'événement requièrent plus d'un actionneur pour exécuter la réaction (cela dépendra de l'intensité de l'événement et des capacités des actionneurs).

En outre, si le nombre d'actionneurs exécutant la réaction est très grand, les actions peuvent être effectuées en dehors de la zone d'événement, ce qui, selon l'application, peut causer des résultats désastreux tels que la consommation inutile des ressources de l'actionneur.

Egalement, l'exécution des mêmes tâches par différents actionneurs et au même moment peut être à l'origine de catastrophes (si l'on dispose, par exemple, d'actions tranquillisant un gaz).

Il importe donc, lors de l'affectation des charges aux actionneurs, de prendre en considération les risques qui se rattachent à l'assurance de la tâche.

Aussi, indépendamment du nombre d'actionneurs recevant les données captées, l'objectif de MAT consiste à choisir les meilleurs actionneurs tout en répondant aux exigences de l'application et de l'événement. Le type de sélection (MA ou SA), en phase de coordination capteur-actionneur, affecte seulement le mécanisme de coordination nécessaire au choix des meilleurs actionneurs.

L'objectif principal de SAT réside également dans le choix du meilleur actionneur (celui par exemple qui peut couvrir toute la zone d'événement) pour réagir.

Si le choix se porte sur MA, la coordination pour SAT peut être considérée comme un cas spécial de la MAT (les actionneurs qui reçoivent les données captées coordonnent dans la DD ou dans CD et choisissent le meilleur actionneur).

Toutefois, si un actionneur isolé est informé d'un événement donné (SA), il peut soit lancer l'action par lui-même soit communiquer d'abord avec les autres actionneurs/centre de décision. Si la coordination capteur-actionneur prend beaucoup de temps et l'application est incompatible avec la latence, l'actionneur doit alors fournir les conditions minimales pour réagir (il doit, par exemple, couvrir toute la région et disposer d'assez d'énergie). Le lancement d'une action doit s'effectuer à temps.

Ainsi, même si l'action n'est pas réalisée par le meilleur actionneur, elle est accomplie de façon opportune.

Si l'application tolère le retard ou si l'actionneur ne dispose pas des conditions minimales pour commencer la réaction par lui même, ce qui lui permettrait de choisir l'actionneur adéquat dans le cas de DD, il faudra effectuer un message d'annonce et choisir alors le meilleur actionneur en fonction des réponses transmises.

I.3.2.2 axes de recherche

Comme déjà vu, dans WSAN, le problème de répartition de charge est résolu par les actionneurs qui coordonnent explicitement, de manière centralisée ou distribuée. Cette coordination s'inscrit dans la problématique suivante:

- Elle se heurte à la question des algorithmes qui sont nécessaires, pour indiquer aux actionneurs recevant l'information captée s'il s'agit de SA ou MA. Dans le cas de SAT, le problème est de choisir l'actionneur qui effectuera l'action, parmi tous les actionneurs qui en sont capables. Pour le cas de MAT, il s'agit de savoir comment décider du nombre optimal d'actionneurs effectuant les actions.

- Le modèle de communication est nécessaire entre les actionneurs aussi bien pour SAT que pour MAT. La communication entre les actionneurs voisins peut s'avérer impossible si la distance qui les sépare est supérieure à leur gamme de transmission.

Dans ce cas, les actionneurs utilisent les capteurs comme intermédiaires pour la coordination actionneur-actionneur.

• Dans DD, pour les deux cas SAT et MAT, l'exécution instantanée de différents événements détectés dans une région peut être exigée de manière à s'assurer qu'il n'y a aucun effet nuisible sur l'environnement de cible.

• Dans la DD ou CD, pour la MAT, quelques applications peuvent nécessiter une intervention simultanée et synchronisée des actionneurs sur l'événement. Dans ce cas, les actionneurs doivent coordonner en mode distribué ou centralisé pour déterminer les périodes d'exécution des actions.

• Dans la CD et DD, cas SAT et MAT, lorsque des événements se produisent en différents endroits, il peut s'avérer nécessaire que les actions soient exécutées par un ensemble d'actionneurs qui ne sont pas nécessairement proches de l'endroit d'événement initial. Dans ce cas, les actionneurs, proche de l'événement, reçoivent l'information et la redirigent vers les actionneurs les plus appropriés.

• Dans le CD et DD, pour MAT, il peut être nécessaire de passer par une redondance des actions déclenchées par un sous ensemble d'actionneurs dans le but de conserver l'énergie consommée dans la région.

• Si la couverture d'action est plus grande que la région d'événement, pour des cas de SAT et MAT, il importe que la charge soit partiellement exécutée par un ensemble d'actionneurs. Cette exécution partielle des actions impose de nouvelles idées couvrant en premier lieu la taille de l'événement.

• Dans la DD, pour les deux cas MAT et SAT, en fonction du nombre d'actionneurs recouvrant l'information d'événement, se produit un message d'annonce (cas de SA) ou une négociation (cas de MA). En conséquence, est il nécessaire d'indiquer le contenu des messages ? Des algorithmes qui fournissent des transmissions de données efficaces pour les deux types de messages doivent être développés.

- Dans le cas du CD, le défi consiste à choisir l'actionneur qui fonctionnera comme unité de décision. Il y a, d'ailleurs, un besoin de mécanisme efficace qui fournit à l'unité de décision les caractéristiques courantes (endroits, capacité,..) d'autres actionneurs dans le réseau, de sorte qu'il puisse déclencher les actions les plus appropriées.

- L'une des conditions les plus essentielles de la coordination des actionneurs consiste à réduire au minimum le temps d'accomplissement de la réaction. Ainsi, la coordination et les protocoles de communication doivent soutenir la communication en temps réel dans les WSANs.

I.4 Les protocoles existants

La communication temps réel dans les réseaux de capteurs n'est pas nouvelle. Hu et. al. [8] ont proposé un protocole de communication temps réel nommé SPEED, qui fournit l'unicast temps réel et le multicast temps réel pour les WSNs. Il les réalise en employant une combinaison de contrôle, de rétroaction et d'expédition géographique non déterministe qui prend en compte et la QoS[3] et le nombre de sauts.

Lu et. al. [22] présente une architecture de communication temps réel pour les réseaux de capteurs à grande échelle. Il décrit une politique de gestion des paquets nommée *velocity monotonic scheduling* qui prend en compte et la latence et les limitations de distance.

Felemban et. al. [23] proposent un nouveau mécanisme de délivrance de paquets nommé Multipath and Multi-Speed Routing Protocol (MMSPEED) pour une garantie probabiliste de la QoS dans les réseaux de capteurs. Plusieurs niveaux de QoS sont fournis en garantissant la livraison des paquets avec différentes vitesses, tandis que plusieurs exigences probabilistes sont soutenus pour acheminer de façon fiable les paquets.

Même si un certain nombre de protocoles sont proposés pour WSN, ils peuvent ne pas être adaptés en s'appliquant directement aux WSAN. Des considérations liées notamment à l'heterogeinité des caractéristiques, à la structure de réseau et aux différentes opérations entre les

[3] Qualité de service.

capteurs et les actionneurs sont à prendre en considération. En particulier, et en comparaison avec WSN, la coordination capteur actionneur et la coordination actionneur-actionneur sont les axes les plus importants dans les WSANs. Plusieurs recherches ont été menées pour explorer les réseaux de capteurs hétérogènes [24, 25], sans répondre aux exigences des réseaux de capteurs et d'actionneurs.

Pour les WSAN, Hu et al [26] propose un paradigme de communication anycast. Pour chaque événement, il construit un arbre qui sera modifié en fonction des mouvements des sinks. La diffusion d'intérêt à partir des sinks permet la découverte de routes.

E. Cayirci et. al [27] proposent un protocole de routage many-to-many avec conservation d'énergie. Les actionneurs enregistrent les types de données qui les intéressent en diffusant des messages d'enregistrement. Les capteurs construisent ensuite leurs tables de routage. Dans ce schéma, chaque donnée captée sera envoyée à tous les actionneurs intéressés, ce qui engendrera un trafic de communication élevé. De plus, les deux approches traitent la coordination capteur-actionneur et actionneur-actionneur, ce qui augmente l'efficacité de la capture et de la réaction.

Melodia et. al. [28] propose un framework de coordination basé sur le clustering dirigé par événement. Tous les capteurs dans la zone d'événement acheminent leurs lectures aux actionneurs appropriés à l'aide d'un arbre d'agrégation. Leur travail suppose l'immobilité des actionneurs qui peuvent réagir dans une zone limitée. Ce framework assure la coordination actionneur-actionneur en divisant la zone d'événement entre les différents actionneurs.

Un autre travail [29] utilise la même hypothèse event-driven, mais propose un algorithme event-reporting qui divise la zone d'événement en pièces de maps et transmet les données captées et classées par priorité. De plus, l'algorithme de coordination peut supporter la mobilité des actionneurs dans la zone de déploiement.

Dinh et. al. [30] reprend les progrès et les conclusions de la recherche concernant le problème de coordination pour élaborer une évaluation de performance de trois protocoles de routage utilisés dans les réseaux ad hoc. Cette évaluation est très utile pour la recherche de communication actionneur-actionneur

Durresi ET. al. [31] présente un protocole de diffusion pour les WSAN (GBSA). C'est un algorithme distribué permettant d'indiquer, selon la localisation géographique, s'il est souhaitable de transmettre les messages ou non.
Quelques protocoles sont détaillés ci après :

I.4.1Paradigme de communication

Les protocoles de routage précédemment proposés pour les WSN n'ont pas été conçus pour des réseaux hybrides, où ils peuvent exploiter les ressources des dispositifs riches dans le but de réduire l'utilisation du flux de communication, de l'énergie, de la largeur de bande, de la mémoire et des dispositifs de calcul-contraints des capteurs. Le présent protocole [26] exploite le paradigme de communication d'anycast pour fournir une exécution sensiblement améliorée pour toute application impliquant des noeuds à ressource hybride. Les nœuds à grande capacité peuvent fournir les services suivants : un stockage persistant ou une communication persistante ou encore une action appropriée. Le but de ce protocole est d'assurer une longue durée de vie au système, de réduire la latence peer to peer et d'assurer la scalabilité. Il se caractérise par :

- Sa Simplicité : pour s'adapter aux capacités des capteurs, l'anycast doit être efficace en temps de calcul et en mémoire.
- Son efficacité en consommation d'énergie: l'anycast doit minimiser l'overhead de communication et de contrôle.
- Son Autonomie d'organisation et son adaptativité : pour s'adapter aux mouvements des sinks et garder sa robustesse même en cas de défaillance des nœuds, l'anycast doit être auto organisé et adaptatif.
- Sa distribution : pour qu'il soit scalable, il doit être complètement distribué.

Ce protocole suppose que plusieurs microservers (sinks) qui peuvent être mobiles s'intéressent aux mêmes données. La donnée doit arriver à l'un d'entre eux seulement (anycast). Il suppose aussi que les applications peuvent tolérer les pertes de données car l'anycast n'assure pas la fiabilité. D'un autre coté, la mobilité du sink implique une latence et une charge remarquables car il faut découvrir la route vers un autre sink plus proche. De plus, cette mobilité du sink nécessite une synchronisation entre les sinks avant d'entamer l'envoi des données avec la source d'événement.

A cette fin, le protocole utilise un arbre partagé correspondant à chaque source d'événement. Les sinks forment les feuilles de l'arbre et peuvent dynamiquement joindre ou quitter l'arbre d'anycast.

Bien que l'utilisation de l'arbre pour implémenter l'anycast supporte la mobilité, elle nécessite une sauvegarde des états. Elle maintient aussi un chemin vers chaque sink, ce qui élimine la découverte de route pour chaque mouvement du sink.

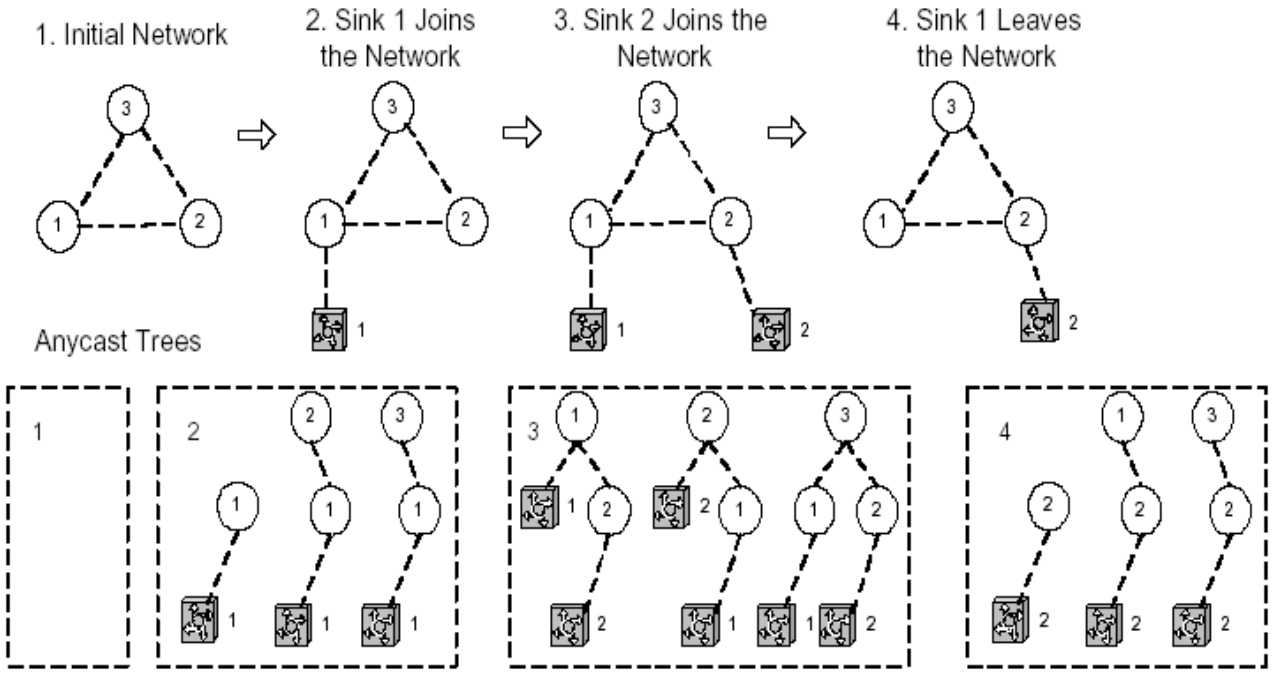


Figure 1.6 Illustration du mécanisme d'anycast.

Cette figure illustre la façon dont l'arbre évolue selon que deux sinks rejoignent ou quittent le réseau. Chaque capteur est potentiellement considéré comme source d'événement. Donc, pour chaque capteur I, il existe un arbre anycast Ti. Chaque arbre est construit des feuilles à la racine. Lorsque sink S rejoint le réseau, une nouvelle branche doit être rajoutée à chacun des arbres. Dans le but de minimiser la consommation d'énergie, l'information qui permet d'ajouter une branche ou d'associer un coût au chemin de i vers S est incluse dans le paquet de découverte de route. La taille de la table est limitée et chaque entrée dans la table expirera avec l'expiration du timer. Lors de l'envoi des paquets, les capteurs sélectionnent parmi les entrées de la table celle qui a le moindre coût. Donc, le paquet est envoyé à un seul sink. Les sinks diffusent périodiquement des paquets pour le rafraîchissement des entrées des différentes tables. Lorsque

un Sink change de position, quelques paquets peuvent être envoyés sur cette route ou à un sink alternatif. Ce protocole n'assure pas la fiabilité et suppose que c'est à l'application de gérer la perte de paquets. La taille de l'arbre dépend du nombre de sinks dans le réseau. Cependant, la taille des tables est limitée de manière à assurer la scalabilité.

I.4.2 Framework distribué de coordination pour les réseaux de capteurs et d'actionneurs

Dans les WSAN, il est nécessaire de délivrer les données des événements avec un minimum d'énergie, tout en respectant la contrainte temps réel. Comme indiqué dans [32] les protocole de routage qui ne considèrent pas la localisation géographique ne sont pas scalables. Ce framework [28] étudie la coordination capteur actionneur en se basant sur le routage géographique. Pour garantir la scalabilité et la consommation efficace d'énergie, plusieurs solutions se basent sur les clusters [33][34][35][36]. La majeure partie d'entre elles dépendent de la topologie (prédéterminé) des capteurs et peuvent être reconfigurées en fonction de la mobilité ou de la défaillance des capteurs.

Dans ce travail, la composition des clusters est a base d'événement selon le protocole *Integer Linear Programming (ILP)*. De ce fait, seulement la zone d'évènement est groupéeclustred. Dans chaque groupe, les capteurs envoient leurs données au même actionneur. L'information d'événement est collectée à l'actionneur approprié (selon la localisation) pour une meilleure utilisation d'énergie. La construction du cluster à base d'événement élimine aussi la charge de communication avant l'occurrence de l'événement, ce qui est préférable dans des applications où les événements sont rares. Ce framework traite la coordination entre les capteurs et les actionneurs ainsi que la coordination inter actionneurs. Le protocole ILP est un algorithme multi états qui garantit une utilisation efficace d'énergie avec un délai de délivrance optimal. Concernant la coordination entre les actionneurs, le protocole *Integer Non-Linear Program (MINLP)* divise les actions entre les actionneurs.

I.4.3 algorithme de routage pour la coordination actionneur-actionneur dans les WSAN

Ce travail [30] se concentre sur la comparaison des algorithmes de routages des réseaux ad hoc : AODV [11], DSR [10] et DSDV [13] pour assurer la communication actionneur-actionneur.

I.4.4 Protocole de diffusion géométrique dans les WSANs

La diffusion est un processus par lequel un noeud envoie un paquet à tout le réseau. Plusieurs protocoles utilisent l'inondation qui implique une redondance coûteuse en énergie et en bande passante. Ce protocole [31] diminue la consommation d'énergie ainsi que la latence et il est scalable.

Il réduit le nombre de retransmissions par l'utilisation de l'approche géométrique et suppose que chaque nœud connaît sa localisation. Plusieurs méthodes existent pour la localisation : le GPS, la différence des temps d'arrivée, l'angle de réception et la puissance du signal. GBS exploite les ressources d'énergie des actionneurs et il est le premier protocole dédié aux WSAN.

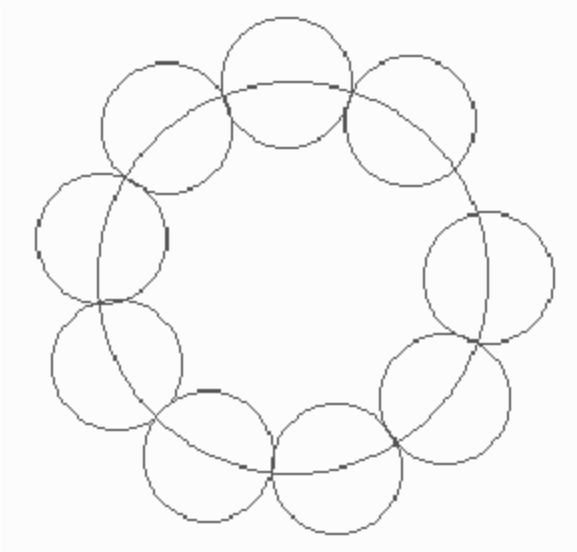


Figure 1.7 la diffusion à partir des actionneurs.

Les actionneurs couvrent une zone plus large par rapport aux capteurs. C'est pourquoi ils peuvent transmettre avant que les capteurs ne le fassent. GBS suppose que chaque capteur connaît la location des 'actionneurs les plus proches et leur nombre. Chaque actionneur connaît aussi la localisation des autres actionneurs. S est le nœud qui génère le message de broadcast m. Il envoie m à l'un des actionneurs qui le transmet à tous les actionneurs sélectionnés, selon la distance qui les sépare. Les actionneurs diffusent alors le message à leurs voisins. Ensuite, un ensemble réduit de capteurs transmettent m pour couvrir la zone restante. Les capteurs voisins de plus d'un capteur ne transmettent pas car ils peuvent faire partie d'une zone déjà couverte. Afin de connaître cette information, chaque capteur doit attendre un délai après la première réception de m. Le message ne sera diffusé que si le capteur le reçoit une seconde fois.

I.4.5 Framework de communication temps réel pour les WSAN

Dans ce framework [29], les actionneurs sont mobiles. Lors de la détection d'un événement, l'information est envoyée par le capteur à l'un des actionneurs qui la communique, à son tour, à l'actionneur approprié. Grâce au déplacement rapide des actionneurs vers la zone d'événement, le système permet d'assurer une communication temps réel, une latence moindre pour l'information des actionneurs et une meilleure coordination de l'organisation.

Lors de la détection des événements, les capteurs concernés commencent la construction des clusters. Le clustering permet de construire l'arborescence nécessaire à l'étape d'agrégation. Le schéma d'agrégation de données multi niveau est tolérant aux erreurs dues aux pannes des capteurs. Les nœuds concernés doivent effectuer l'agrégation des données de leurs voisins immédiats avant de les rediriger vers le nœud de rapport. Le capteur qui représente le cluster-head rapporte enfin toute donnée collectée de sa propre région à l'actionneur approprié. Cet actionneur n'est pas nécessairement celui qui va réagir, ce qui fait appel à la coordination actionneur-actionneur pour l'élection de l'actionneur. La zone d'événement est subdivisée en grids de taille égale.

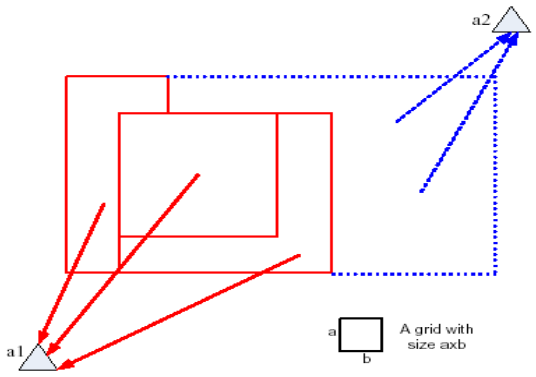


Figure 1.8 la combinaison des grids

Chaque actionneur qui reçoit les données déjà envoyées par les capteurs, aura une vue limitée du clustering. Il devra combiner plusieurs grid dans le cas où il reçoit plusieurs rapports concernant le même événement à une même période de temps. Il doit, en outre, communiquer avec d'autres actionneurs pour échanger les grids et évaluer la taille de l'événement comme le montre la figure I.8, ce qui permet aux actionneurs d'avoir une idée des coordonnées de

l'événement et de réorganiser ainsi la zone d'événement en divisant la combinaison du grid en rectangles. Les actionneurs pourront dès lors, localiser l'événement et se répartir les charges.

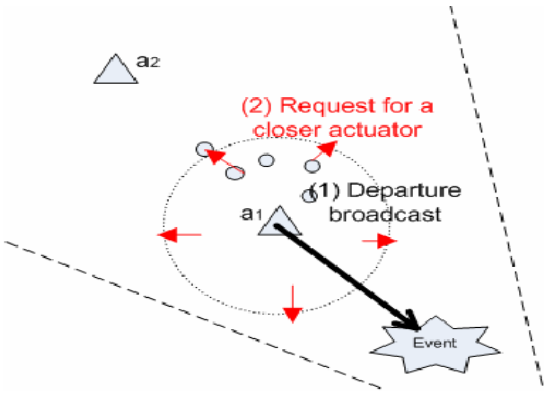


Figure 1.9 déplacement des actionneurs.

Les nouvelles positions des actionneurs doivent être indiquées aux capteurs. Les actionneurs diffusent des messages pour leurs départs et leurs arrivés à d'autres capteurs. Chaque capteur détermine son actionneur potentiel. Ce processus est indiqué dans la figure 1.9.

I.4.6 Comparaison entre les protocoles

Les critères de comparaison ainsi évoqués constituent l'objet de ce paragraphe :

- Types de réseaux : Les premiers travaux ont été consacrés à la coordination dans les réseaux de capteurs hybrides pour converger vers les réseaux WSANs. C'est pourquoi, on distingue les protocoles dédiés aux WSN et aux WSAN.

- Coordination : Dans les WSAN, il existe deux types de coordination, la coordination capteurs-actionneurs et la coordination inter actionneurs. Bien qu'elles soient inséparables, certains travaux n'en tolèrent qu'une seule pour bien cerner le problème.

- Temps réel : cette caractéristique qui est la plus importante pour la coordination dans les WSAN n'est pas prise en considération dans les WSN.

- Localisation : La communication entre les composants des WSAN se base sur les protocoles de routage. Ceux-ci, pour être efficaces, doivent se baser sur la localisation et sur le principe d'organisation du réseau ou clustering.

- Mobilité : Dans les WSAN, les actionneurs peuvent être spécialisés dans des tâches bien spécifiques alors que l'événement peut être déclenché de n'importe quel emplacement. Les actionneurs peuvent se déplacer pour réagir dans la zone d'événement et donc la mobilité constitue un paramètre essentiel dans ces réseaux.

	WSAN/ WSN	Mobilité	Localisation	Clustering	Coordination	Temps réel
Paradigme de communication [26]	WSN	Sink	Oui	Pas de clustering.	Capteur-sink	N'est pas prise en compte.
Framework distribué de coordination pour les réseaux de capteurs et d'actionneurs [28]	WSAN	Pas de mobilité des noeuds.	Oui	Le clustering à base d'événements.	Capteur-actionneur. Actionneur-actionneur.	prise en compte.
Algorithme de routage pour la coordination actionneur-actionneur dans les WSAN [30]	WSAN	Pas de mobilité des noeuds.	Non	Pas de clustering.	Actionneur-actionneur	prise en compte
Protocole de diffusion géométrique dans les WSANs [31]	WSAN	Pas de mobilité des noeuds.	Oui	Pas de clustering.	Capteur-actionneur	N'est pas prise en compte.
Framework de communication temps réel pour les WSAN [29]	WSAN	Actionneur	Oui	Le clustering à base d'événements.	Capteur-actionneur. Actionneur-actionneur.	prise en compte

Tableau 1 : comparaison entre les protocoles.

I.5 Conclusion:

Ce chapitre constitue une introduction à l'étude des réseaux de capteurs et d'actionneurs qui sont très utiles aux environnements réactifs. Différents axes de recherches ont été exposés pour ces catégories de réseaux, la question centrale étant celle de la coordination autrement dit comment les composants du WSAN communiquent entre eux.

Les protocoles déjà présentés pour la coordination entre les capteurs et les sinks dans les WSN ne peuvent pas s'appliquer aux WSANs pour les raisons suivantes :

- Dans WSN, on ne distingue pas les capteurs, le plus important étant la donnée ou l'intérêt.
- Les capteurs jouent tous le même rôle et il n'y a pas de rôle spécifique pour chaque capteur.

- Les capteurs ne prennent aucune décision pour réagir aux événements.
- La propriété du temps réel en WSN n'a pas autant de poids que dans les WSAN.

En dernière partie, et même si peu de travaux ont été consacrés à la coordination dans les WSANs, quelques protocoles sont détaillés.

1. **« Le paradigme de communication »** [26] qui est un protocole utilisé dans les réseaux WSN à réseaux hybrides. Il prend en charge la coordination et supporte la mobilité des sinks. Pourtant, il ne prend pas en considération la contrainte temps réel, ce qui constitue un handicap qui le rend inutilisable dans les réseaux WSAN.

2. « **Le framework distribué de coordination pour les réseaux de capteurs et d'actionneurs »** [28] est dédié au WSANs. Il intègre la coordination capteur-actionneur et la coordination actionneur-actionneur et utilise un routage géographique. Il prend en compte la latence et la consommation d'énergie mais pas la mobilité des actionneurs.

3. Afin d'exploiter les études déjà effectuées au sujet du routage dans les réseaux ad hoc, **une étude à base d'analyse des performances des algorithmes de routage** [30] compare l'efficacité de trois algorithmes de routage pour la communication actionneur-actionneur. La communication capteur actionneur n'est pas prise en considération.

4. **Le protocole de diffusion géographique dans les WSAN** [31] montre comment il est possible d'exploiter la localisation géographique pour communiquer un message quelconque, tout en consommant le minimum d'énergie et de latence.

5. **Le framework de communication temps réel pour les WSAN** [29] est un framework qui s'apparente au 2eme protocole tout en tenant compte de la mobilité des actionneurs. Ce framework n'a pas été évalué par simulation et ses paramètres de performance ne sont pas définis.

Notre travail se fonde sur le problème de coordination dans les réseaux de capteurs avec actionneurs, en respectant les différentes contraintes exigées par l'environnement et l'application : énergie, latence, priorité des événements. De ce point de vue, nous nous concentrerons sur le 2eme [28] et le 5eme [29] travail, cités ci-dessus, de manière à en apprécier les avantages et les inconvénients et de proposer une solution appropriée.

Chapitre II : le framework distribué pour la coordination dans les réseaux de capteurs et d'actionneurs

II.1 Introduction

Les WSANs sont des systèmes distribués hétérogènes, constitués de capteurs et actionneurs dont la fonction consiste à capter, contrôler et réagir à l'environnement physique. Le présent chapitre traite le problème de coordination et de communication dans les WSAN. Pour la résolution du problème capteur-actionneur et actionneur-actionneur, un framework [28] est présenté. Ce framework, qui suppose que les actionneurs sont statiques, assure, par ailleurs, un compromis entre la consommation d'énergie et la fiabilité, celle-ci étant le délai maximal de réaction. Il intègre également le temps d'arrivée des données à l'actionneur. Ce délai doit être minimisé au maximum de façon à garantir la fraîcheur et la faisabilité des actions déclenchées par les actionneurs.

Le modèle de coordination capteur-actionneur repose sur le partitionnement du réseau en clusters. Lors de la détection d'un événement, l'ensemble des capteurs est fragmenté et chaque sous-ensemble de capteurs sera sous la responsabilité d'un actionneur. Cette solution de partitionnement est d'abord décrite par des formules mathématiques qui ne convergent pas si la taille du réseau augmente. Un protocole distribué a été proposé dans ce cadre.

Egalement, pour la répartition des charges entre les actionneurs, deux solutions ont été proposées, la première, optimale est basée sur un système d'équation mathématique alors que la seconde, approximative, assure la scalabilité. Le chapitre est donc scindé en deux grands modules : la coordination capteur-actionneur et la coordination actionneur-actionneur, chaque partie incluant deux solutions, l'une optimale et l'autre approximative.

II.2 Coordination capteur-actionneur

II.2.1 Formulation du problème

Comme déjà évoqué précédemment, la communication capteur-actionneur doit intégrer la contrainte temps réel. C'est pourquoi une nouvelle notion, celle de la fiabilité, est introduite pour représenter le pourcentage de paquets générés par les capteurs et reçus dans un délai

prédéterminé. Cette notion qui porte, contrairement à d'autres définitions, sur la livraison en temps réel de paquets de données aux actionneurs, peut être abordée sous deux angles principaux : par rapport à l'événement et par rapport au paquet.

La fiabilité des paquets :

La latence limite B représente le délai maximal autorisé entre le moment où l'événement est détecté et celui où l'information d'événement arrive à l'actionneur. Pour être fiables, les paquets de données doivent arriver aux actionneurs au bout de la limite B.

La fiabilité de l'événement :

La fiabilité de l'événement **r** est le ratio des paquets de données fiables parmi l'ensemble des paquets générés durant l'intervalle[4] de décision. Le coût de la fiabilité de l'événement **rth** est la fiabilité minimale tolérée par l'application. La différence (rth-r) entre la fiabilité exigée par l'application et celle observée durant une période donnée, représente le manque de fiabilité. Une substration négative indique que la fiabilité est au dessus du coût exigé, ce qui signifie un excès de fiabilité.

Remarque: la latence limite B et le rth dépendent des exigences de l'application. Le problème de la coordination capteur-actionneur consiste à établir un chemin de données entre chaque capteur résidant dans la zone d'événement et les actionneurs avec :

i) L'assurance que la fiabilité observée r est supérieure à celle exigée par l'application rth (i.e., $r \geq rth$);

ii) Une minimisation de la consommation d'énergie associée aux chemins de données.

Le problème de la coordination est résolu grâce à un partitionnement dirigé par événement et un modèle nommé ILP. Dans la prochaine section, sera décrit le modèle du réseau et de l'énergie. ILP sera décrit en section II.2.1.2.

[4] A Chaque fois qu'un ou plusieurs paquets sont perdus par l'intermédiaire des capteurs, l'actionneur est avisé de la perte dans l'en-tête du prochain paquet de données, de sorte que la perte de paquets puisse être prise en compte dans le Calcul de la fiabilité.

II.2.1.1 modélisation du réseau

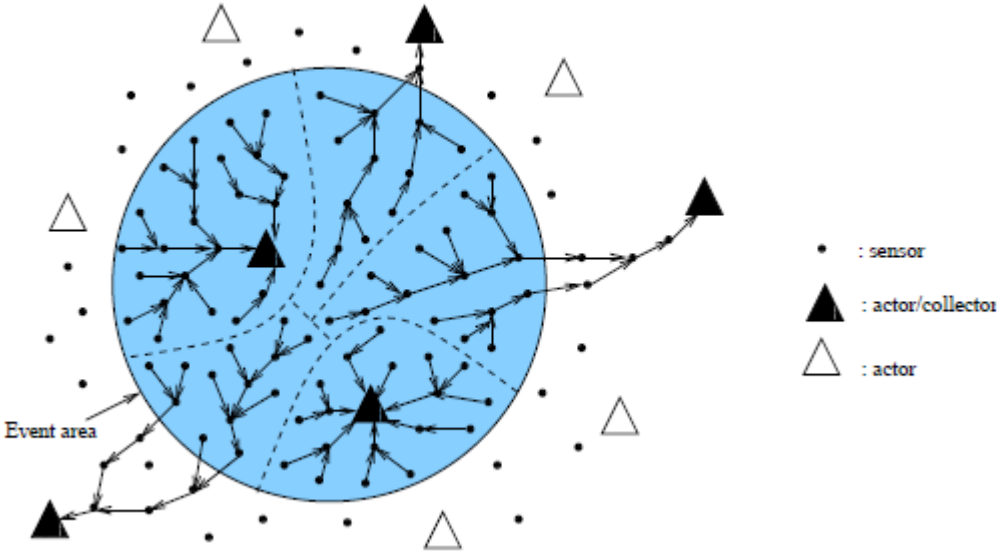


Figure II.1 : le clustering à base d'événement avec multiple actionneurs.

Le réseau de capteurs et d'actionneurs est représenté par un graphe $G(S^V, S^E)$

-$\mathbf{S^V}$ est un ensemble de noeuds (vertexes) /$\mathbf{S^V = \{v1, v2, \ldots, vN\}}$.

-$\mathbf{S^E}$ est l'ensemble des liens (arcs) tel que : $\mathbf{eij \in S^E}$ Ssi **vi** et **vj** sont dans la même portée de transmission.

-Soit $\mathbf{S^A}$ l'ensemble des actionneurs, avec $\mathbf{N_A = |S^A|}$. L'actionneur qui collecte le trafic à partir d'un ou plusieurs sources est nommé *collecteur*.

-Soit SS l'ensemble des capteurs qui détectent l'événement, avec **NS = |SS|,** c'est-à-dire les capteurs qui résident dans la zone d'événement.

On a donc, $\mathbf{S^A \in S^V}$, $\mathbf{S^S \in S^V}$ et $\mathbf{S^A \cap S^S = \emptyset}$.

On définit **P = {(s, a) : s€S, a€A}** comme l'ensemble source-destination.

Cependant, les informations propres à chaque connexion peuvent être décrites par une matrice

P = [Psa], où **Psa** représente le taux moyen des informations transitant (bits/s) entre un nœud source $\mathbf{s \in S^A}$ et un actionneur **a € A**. Il est supposé que toutes les sources génèrent un bit d'information. Par conséquent, la matrice du trafic dans le modèle peut être omise car lorsque toutes les sources d'information génèrent le même taux de transfert des données, comme c'est le cas dans de nombreuses applications d'intérêt, la solution du problème ne dépend pas d'elle.

La consommation d'énergie au niveau de la couche physique peut être définie par la formule suivante : **E = Etranselec + βdα+Erecelec**, où :

• Etranselec est indépendant de la distance et dépendant de la charge électronique de la transmission (PLLs, VCOs, bias currents, etc.) et de la puissance de traitement ;
• Erecelec dépend de la charge électronique de réception,
• βdα dépend de la distance et représente la puissance pour transmettre un bit entre une source et une destination distante de d.

Comme indiqué dans [37], **Etranselec = Erecelec = Eelec.** L'expression peut être simplifiée à
$E = 2Eelec + \beta d^{\alpha}$, où :
-α est l'exposant de la perte de chemin ($2 \leq \alpha \leq 5$),
-β est une constante [J/(bits · mα)],
-Eelec est l'énergie nécessaire pour transmettre ou recevoir un bit [J/bits].

Dans ce système, un capteur entame l'agrégation des données dés lors qu'il reçoit celles-ci à partir d'au moins deux nœuds. Un seul paquet est créé au lieu de plusieurs d'où une réduction des données transmises. Pour ce faire, il faut implémenter un algorithme da*ta fusion* [38], au niveau de chaque capteur. En outre, le coût de traitement n'est pas connu car le coût de communication est nettement plus élevé, ce qui est prouvé par expérimentation puisque l'énergie nécessaire à la transmission d'un kbit équivaut à celle nécessaire à l'exécution de 300,000 instructions.

II.2.1.2 les détails de « integer linear program »

Le but ici est de trouver un arbre d'agrégation de données (da-trees) qui relie l'ensemble des capteurs résidant dans la zone d'événement (nommé sources) aux actionneurs appropriés. Les da-trees sont composés par agrégation de flux individuels. Le flux est une connexion entre un capteur et un actionneur. Les feuilles du da-tree sont les sources (mais toute les sources ne sont pas nécessairement des feuilles) et chaque actionneur peut être la racine d'un da-trees comme il peut ne pas participer à la communication.

Les da-trees sont construits de telle manière que chaque source appartient à un arbre seulement et chaque arbre possède un actionneur comme racine. Par conséquent, chaque source est associée à un actionneur pour assurer une partition (groupement) optimale. En fait, event-driven partitioning peut être vu sous deux aspects :

1) la sélection de l'ensemble optimal des actionneurs auxquels les capteurs transmettront leurs lectures;
2) la construction des da-trees avec un minimum d'énergie et le respect de la contrainte de fiabilité liée à l'événement.

La stratégie optimale pour le partitionnement à base d'événement est formulée comme *Integer Linear Program* (ILP) [39]. La topologie du réseau doit être *1-connected*, c'est-à-dire qu'il existe au moins un chemin entre chaque capteur et actionneur. Les variables suivantes sont utiles à la formulation du problème de coordination capteur-actionneur:

a) $\mathbf{e_{ij}}$ est une variable binaire représentant un lien, égale à 1 Si et Seulement si les nœuds i et j sont dans la même portée;

b) $\mathbf{c_{ij}}$ est le coût associé au lien entre les noeuds **i** et **j**, i.e., $\mathbf{2Eelec + \beta d^{\alpha}_{ij}}$, où $\mathbf{d_{ij}}$ est la distance entre **I** et **j**

c) $\mathbf{x^{k}_{ij}}$ est une variable binaire qui est égale à 1 Si et Seulement si le lien (i, j) fait partie du da-tree associé à l'actionneur k;

d) $\mathbf{f^{k,s}_{ij}}$ est une variable binaire égale à 1 Si et Seulement si la source s envoie des données à l'actionneur **k** et le lien **(i, j)** appartient au chemin du **s** vers **k**;

e) $\mathbf{l^{k,s}}$ est une variable binaire égale à 1 Si et Seulement si le capteur s envoie des données à l'actionneur k;

f) $\mathbf{p_{ij}}$ le délai de propagation associé au lien **(i, j)** et défini comme **dij/v**, où **v** est la vitesse de propagation du signal ;

g) **đ** est un paramètre qui comptabilise le délai de traitement et d'accès au canal pour chaque capteur ;

h) **B** est la latence limite associée à chaque flux;

i) **r** et **rth** sont respectivement la fiabilité de l'événement et la fiabilité exigée par l'application;

j) $\mathbf{b^{k,s}}$ est une variable binaire égale à 1 Si et Seulement si la connexion entre la source **s** et l'actionneur **k** ne respecte pas la latence limite **B**, i.e., le délai point à point est plus grand que B;

k) **Q** le nombre de sources non-compatibles.

Le problème se pose donc de la manière qui suit **(formulation du problème** ComMin**)**:

$$
\begin{aligned}
&Given: && e_{ij},\ c_{ij},\ p_{ij},\ v,\ \bar{d},\ B,\ r_{th} \\
&Find: && x_{ij}^{k},\ f_{ij}^{k,s},\ l^{k,s},\ b^{k,s},\ r \\
&Minimize: && C^{TOT} = \textstyle\sum_{k\in\mathcal{S}^{\mathcal{A}}} \sum_{(i,j)\in\mathcal{S}^{\mathcal{E}}} x_{ij}^{k} \cdot c_{ij} + \gamma \cdot Q && (1) \\
&Subject\ to: &&
\end{aligned}
$$

$$\sum_{j\in\mathcal{S}^{\mathcal{V}}} (f_{sj}^{k,s} - f_{js}^{k,s}) = l^{k,s}, \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}; \quad (2)$$

$$\sum_{j\in\mathcal{S}^{\mathcal{V}}} (f_{kj}^{k,s} - f_{jk}^{k,s}) = -l^{k,s}, \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}; \quad (3)$$

$$\sum_{j\in\mathcal{S}^{\mathcal{V}}} (f_{ij}^{k,s} - f_{ji}^{k,s}) = 0,$$

$$\forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}, \forall i \in \mathcal{S}^{\mathcal{V}}\ s.t.\ i \neq s, i \neq k; \quad (4)$$

$$f_{ij}^{k,s} \leq e_{ij}, \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}, \forall i \in \mathcal{S}^{\mathcal{V}}, \forall j \in \mathcal{S}^{\mathcal{V}}; \quad (5)$$

$$f_{ij}^{k,s} \leq x_{ij}^{k}, \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}, \forall i \in \mathcal{S}^{\mathcal{V}}, \forall j \in \mathcal{S}^{\mathcal{V}}; \quad (6)$$

$$\sum_{k\in\mathcal{S}^{\mathcal{A}}} l^{k,s} = 1, \forall s \in \mathcal{S}^{\mathcal{S}}; \quad (7)$$

$$f_{ij}^{k,s} \leq l^{k,s}, \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}, \forall i \in \mathcal{S}^{\mathcal{V}}, \forall j \in \mathcal{S}^{\mathcal{V}}; \quad (8)$$

$$\varepsilon \cdot [B - \sum_{(i,j)\in\mathcal{S}^{\mathcal{E}}} f_{ij}^{k,s}(p_{ij} + \bar{d})] \leq b^{k,s},\ \ \forall s \in \mathcal{S}^{\mathcal{S}}, \forall k \in \mathcal{S}^{\mathcal{A}}; \quad (9)$$

$$Q = \sum_{k\in\mathcal{S}^{\mathcal{A}}} \sum_{s\in\mathcal{S}^{\mathcal{S}}} b^{k,s}; \qquad r = \frac{|\mathcal{S}^{\mathcal{S}}| - Q}{|\mathcal{S}^{\mathcal{S}}|} \geq r_{th}. \quad (10)$$

L'objectif de la fonction (1) est de minimiser la consommation d'énergie. Une pénalité est imposée par la multiplication de Q par γ(coefficient de pénalité). La valeur de γ doit être suffisamment élevée pour garantir l'unicité de la solution. Dans le même esprit, cela peut permettre la minimisation de Q dans l'équation (10).

Un flux est une connexion entre une source et une destination. Un flux associé à un même actionneur sera rassemblé dans un même da-tree. Différentes contraintes existent. Les contraintes (2), (3) et (4) expriment la conservation des flux [39] :

• la contrainte (2) garantit que chaque noeud source génère uniquement un flux associé a exactement un seul actionneur. Elle assure que les nœuds non source ne génèrent aucun flux.

• La contrainte (3) exige que chaque flux généré par une source ne soit collecté que par un seul et unique actionneur.

• La contrainte (4) garantit que le flux est nul pour les nœuds non sources et non actionneurs.

• La contrainte (5) assure que le flux est créé par les liens existants entre les nœuds adjacents (les nœuds dans la même portée de transmission).

• La contrainte (6) force l'agrégation des données des différentes sources associées au même actionneur.

• La contrainte (7) impose que chaque source ne communique ses données qu'à un seul actionneur.

• La contrainte (8) Veille à ce que toutes les variables d'un flux allant d'une source à un actionneur soient égales à zéro sauf pour l'actionneur sélectionné par la source.

• La contrainte (9) requiert que la variable binaire $\mathbf{b^{k,s}}$ soit égale à 1 si et seulement si le flux entre la source **s** et l'actionneur **k** viole la latence limite **B**. le coefficient négatif **ε** rend la somme entre parenthèses inférieure a 1. En effet, si la latence limite est violée, le coté gauche de (9) sera une petite valeur positive, ce qui force la variable binaire $\mathbf{b^{k,s}}$ à 1. Dans le cas contraire, la valeur du coté gauche sera négative et $\mathbf{b^{k,s}}$ doit être égale à o pour minimiser la fonction objective dans (1).

• Finalement, dans (10), **Q** est le nombre de sources non compatibles et r peut être calculé en divisant le nombre de sources incompatibles par l'ensemble des sources. Il est évident que **r** doit être supérieur à **rth** (fiabilité exigée par l'application).

En tant qu'ILP, plus NP-complet que « Geometric Connected Dominating Set", ComMin est NP-complet et sa solution ne concernera qu'un nombre limité de nœuds, inférieurs à 100. Ce système a été développé dans le seul but d'avoir un aperçu sur les propriétés de la solution optimale, et l'élaboration de solutions distribuées qui tentent de reproduire les caractéristiques de la configuration optimale du réseau.

La conception du protocole distribué présenté dans la section suivante est basée sur l'analyse et l'étude des performances du système précédant. Apres l'observation, les nœuds tentent d'envoyer des données vers l'actionneur le plus proche afin de réduire la consommation d'énergie, à moins qu'un autre noeud n'exécute l'agrégation, ce qui peut réduire le coût de communication. La formulation mathématique constitue, en outre, un repère fondamental pour la consommation optimale d'énergie d'une éventuelle solution distribuée. Dans cet esprit, dans la section II.2.2 sera détaillé l'algorithme distribué qui est moins optimal mais évolutif.

II.2.2 Le protocole distribué (la coordination capteur-Actionneur):

L'objectif de ce protocole est la construction des da-trees reliant les différentes sources aux actionneurs résidant dans la zone d'événement. La construction doit minimiser la fonction objective (1), c-à-d fournir la fiabilité nécessaire **rth** à moindre consommation d'énergie.

Comme on le verra dans la section II.2.2.1, le protocole proposé pour la construction du da- arbres entre les sources et les actionneurs peut être considéré comme une solution approximative au problème de partitionnement à base d'événement. Ce protocole est conçu pour produire des performances proches de la solution optimale. Il est nommé ***Distributed Event-driven Partitioning and Routing*** **(DEPR)**. Ses performances seront comparées avec la solution optimale dans la dernière section.

En utilisant les algorithmes de routage géographique, les décisions du routage seront basées sur la localisation, ce qui réduit l'utilisation de l'énergie d'une manière presque optimale [32]. Ceci revient à dire que dans un réseau de capteur dense, l'information de la topologie d'une zone éloignée de la zone de décision (décision du routage) n'est pas essentielle. Pour cette raison, l'objectif du protocole proposé est de minimiser la consommation d'énergie en ne s'intéressant qu'à l'information locale. De cette manière, il garantit un compromis entre l'efficacité d'énergie pour établir les da-trees et la quantité d'informations nécessaires à chaque capteur afin de prendre les décisions du routage.

Chaque capteur surveille la valeur de la fiabilité. La fiabilité est contrôlée en se basant sur la minimisation du délai de transmission des informations par réduction du nombre moyen de sauts dans le chemin, tandis que la méthode la plus utilisée pour la minimisation d'énergie est celle du contrôle de la puissance de transmission. Une autre méthode, utilisée pour réduire la charge dans le réseau est celle de l'agrégation des données.

Le protocole **DEPR** impose la prise en compte des exigences suivantes :

- Chaque capteur connaît sa position et peut être équipé d'un récepteur GPS [49] ou d'autres techniques de localisation [40].
- Chaque capteur connaît la position de ces voisins car il diffuse localement sa position.
- Chaque actionneur balise périodiquement sa position aux capteurs.
- Le réseau est synchronisé en utilisant les protocoles existants [41].

Les algorithmes de routage basés sur la localisation doivent régler le problème des boucles qui peuvent bloquer la transmission des informations. Les deux notions suivantes garantissent l'inexistance de boucles :

• Soient les noeuds v et x, l'avance absolue du nœud x par rapport à v ($\mathbf{A_{xv}}$), est la distance entre v et l'actionneur qui lui est le plus proche cv moins la distance entre x et son actionneur voisins cx[5].

• Soit les noeuds v et x, l'avance du collecteur c de x par rapport à v ($\mathbf{A_{xcv}}$), est la distance entre v et c moins la distance entre x et c.

Logiquement, si $\mathbf{A_{xv}}$ est positif, alors x est plus proche d'un actionneur que v et si $\mathbf{A_{xcv}}$ est positif alors x est plus proche de c que de v.

Pour n'importe quel chemin multi sauts, une substraction positive $\mathbf{A_{xv}}$ *au niveau de n'importe quel saut* garantit l'inexistence de boucle dans le chemin car au niveau de chaque saut le paquet est plus proche des collecteurs qu'au saut précédant. Une $\mathbf{A_{xcv}}$ positive permet de s'assurer que le chemin de la source vers la destination ne contient aucune boucle.

II.2.2.1 descriptions générales de DEPR :

L'objectif du protocole proposé est de créer des da-trees reliant les sources (capteurs) à un sous ensemble d'actionneurs nommés collecteurs. Un da-tree associe à chaque collecteur un sous ensemble de source. De cette manière, l'ensemble des sources sera implicitement partitionné avec chaque partie composée par les sources associées à un seul collecteur. Chaque capteur altère entre 4 états différents: **idle**, **start-up**, **speed-up**, **aggregation** et **recovery** dont le diagramme de transition d'un état à un autre est représenté dans la figure II.2.

L'objectif principal de cette solution est de minimiser la latence par la réduction du nombre de sauts entre la source et la destination lorsque la fiabilité exigée par l'application est violée et de conserver l'énergie lorsque la fiabilité nécessaire est assurée. Cette opération peut être réalisée sur la base des résultats des probabilités calculées au niveau des capteurs. Cela revient à inciter les capteurs à choisir leur prochain saut de façon à augmenter la latence et à réduire la consommation d'énergie lorsque la fiabilité est élevée, et inversement à réduire le délai au détriment de la consommation d'énergie lorsque la fiabilité est faible. Ce résultat est obtenu en ajustant en même temps et dynamiquement la puissance de transmission.

Dans les réseaux sans fil multi sauts, le contrôle de la puissance de transmission est réalisé à l'aide du crosslayer [42]. Le niveau de puissance de transmission :

• affecte la qualité du signal reçu et induit donc des erreurs importantes et une consommation d'énergie au niveau de la couche physique ;

[5] Cv et cx peuvent être différents actionneurs.

- détermine aussi la zone de communication couverte par le signal, ce qui limitera le nombre des nœuds qui seront considérés comme sauts prochains.
- affecte les interférences directement liées à la capacité du canal et donc les possibilités de congestion qui font appel à la couche transport.

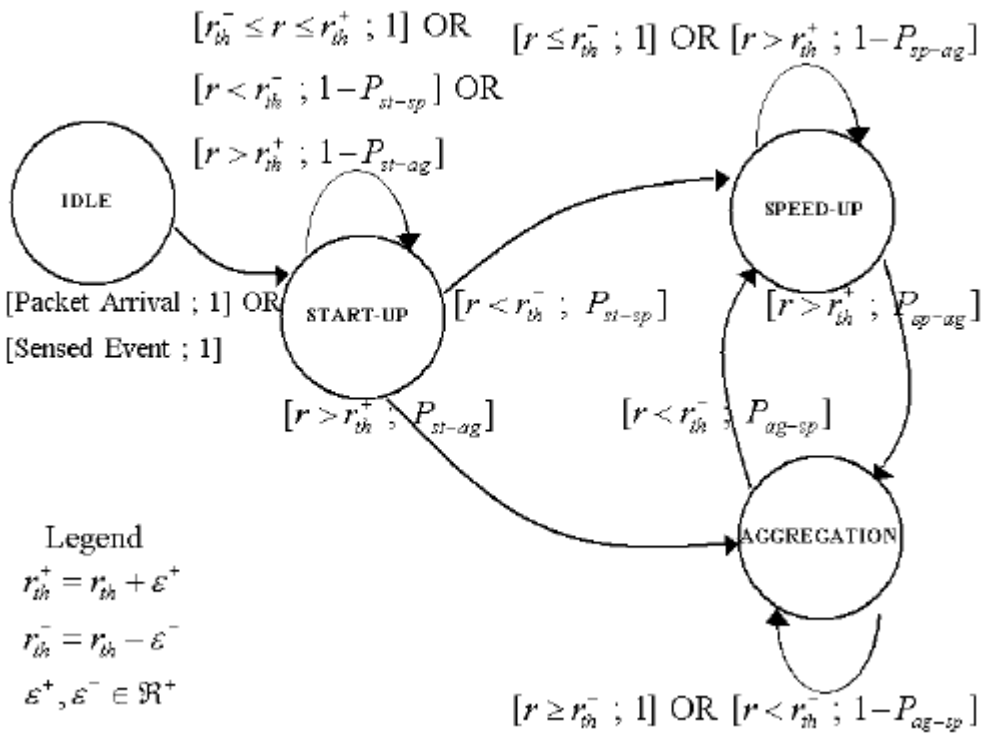
Par conséquent, le contrôle de puissance produit des impacts sur les paramètres de performance du réseau tels que le débit, la latence et la consommation d'énergie. Plus précisément, le choix d'une haute puissance de transmission avec une politique appropriée au niveau de la couche réseau, peut réduire le nombre de sauts nécessaires pour arriver à la destination, tout en augmentant les interférences et les terminaux exposés dans les canaux sans fils partagés. Il permet aussi d'améliorer la connectivité du réseau en augmentant le nombre de liens directs perçus par chaque nœud, mais au détriment de la réduction de la réutilisation spatiale des puissances de transmission. Comme nous le verrons plus loin dans les simulations, en cas de faible trafic, la réduction de la longueur du chemin conduit à une réduction de la latence et à une consommation d'énergie.

Par ailleurs, le choix d'une plus faible puissance de transmission permet de réduire les interférences potentielles perçues par les émetteurs et, dans une certaine mesure, de diminuer la consommation d'énergie et d'augmenter la durée de vie du réseau [43]. Toutefois, une faible puissance de transmission exige plus de nœuds reliant la source et la destination résultant de plus de latence et à une augmentation du nombre de terminaux cachés.

Par conséquent, un régime efficace de contrôle de la puissance devrait déterminer la façon d'équilibrer entre de multiples objectifs : la capacité du réseau, l'autonomie de la batterie, la latence, etc. C'est aussi le but de ce protocole adaptatif distribué.

Les nœuds source ajoutent un horodatage aux paquets de données associées a l'événement qu'ils transmettent aux actionneurs, pour leurs permettre de calculer le délai de chaque paquet. Durant chaque intervalle de décision, chaque actionneur calcule la fiabilité de l'événement r comme étant le ratio des paquets expirés, sur tous les paquets générés dans cette même période. La valeur calculée sera ensuite diffusée. Les Capteurs associés à ce collecteur fondent leurs états sur la fiabilité observée par ce collecteur.

Lorsque la valeur de r est inférieure à rth1 (le minimum de fiabilité exigé par l'application) où rth1=rth-o1, il est nécessaire d'augmenter la vitesse de délivrance en réduisant le nombre de sauts. Inversement, si r est supérieur à rth2 (rth2=rth+ o2), la fiabilité est dans ce cas en excès et il est préférable de conserver l'énergie.



FigureII.2 : transition des états des capteurs.

Les coefficients o1 et o2 sont nécessaires pour définir la zone de tolérance. De meilleures valeurs garantissent un bon compromis entre la stabilité et la tolérance et elles seront déterminées par les simulations. Chaque capteur commence à l'état **idle** où il détecte les événements et surveille le canal pour d'éventuels paquets.

Il transite à l'état start-up lorsqu'il détecte un événement ou s'il reçoit le premier paquet de données à partir des capteurs voisins. Les opérations collaboratives des capteurs dans cet état permettent d'établir des voies de communications vers l'actionneur associé à chaque source résidant dans la zone d'événement. Ces chemins constituent un bon compromis entre la latence et la consommation d'énergie. Afin de réaliser ces objectifs, chaque capteur dans la zone d'événement entre à l'état start-up et entame l'envoi des paquets vers un actionneur.

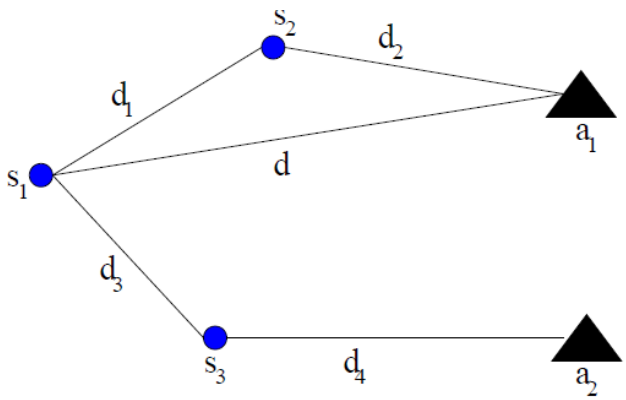
Les capteurs écoutent les messages reçus à partir du collecteur auquel ils sont associés. Si r est inférieur à rth1, il est nécessaire de réduire le délai de délivrance des paquets aux actionneurs, ce qui est assuré en diminuant la longueur du chemin. A cet effet, le capteur à l'état start-up avec (rth1 supérieur à r) transite à l'état speed-up avec une probabilité de **Pst.sp**. Cette probabilité peut être une fonction monotone décroissante exprimant le manque de fiabilité. La notation [cond; P] dans la Figure II.2 indique que la transition sera franchie avec une probabilité P si la condition cond est vérifiée.

Si la fiabilité calculée r est supérieure à rth2, il sera possible de conserver l'énergie. Dans ce cas, le capteur à l'état start-up entame l'état d'agrégation avec une probabilité Pst.ag, qui pourrait être une fonction monotone croissante exprimant l'excès de fiabilité. L'énergie sera, en effet, minimisée en reliant les données des différents voisins appartenant au même da-tree.

Ensuite, le capteur peut altérer entre speed-up et agrégation state afin de répondre aux exigences du collecteur. Le capteur à l'état speed-up transite à l'état d'agrégation avec une probabilité de Psp.ag lorsque r > rth2. La transition inverse est franchie avec une probabilité Pag.sp lorsque r< rth1. Pag.sp peut augmenter avec la croissance du manque de fiabilité alors que Psp.ag croit avec l'excès de fiabilité. Ce protocole répond ainsi à l'objectif de converger vers une solution à fiabilité proche de celle exigée par l'application, avec une minimisation de la consommation d'énergie. Le capteur retourne à l'état **idle** s'il ne génère et ne reçoit pas des paquets durant **idleTimeout** secondes.

Notons que la politique probabiliste doit préserver le système lorsque tous les capteurs changent leurs états en même temps. Plus clairement, l'algorithme proposé est particulièrement efficace lorsque le temps nécessaire pour mettre en place et configurer l'arbre est plus petit par rapport à la longueur de la phase de surveillance et de réaction. Dans ce qui suit, seront détaillées les opérations exécutées dans chaque État.

A. Etat Start-up



FigureII.3 : la règle à deux sauts.

Comme le montre la Figure II.3, un nœud entre dans l'état start-up quand il détecte un événement, ou quand il reçoit un paquet envoyé à l'actionneur. Le capteur i dans cet état, qu'il soit source ou relais pour un paquet de données, sélectionne son prochain saut fondé sur la règle dite des deux sauts. Le noeud i choisit j comme prochain saut parmi ses voisins. Ce

nœud doit minimiser la somme de la consommation d'énergie des deux chemins : de i à j et de j à son actionneur associé cj , qui est calculée comme suit :

$Ej = 2Eelec + \beta(d_{ij})^{\alpha} + 2Eelec + \beta(d_{jcj})^{\alpha}$, (11)

- d_{ij} représente la distance entre i et j,
- d_{jcj} représente la distance entre j et cj (l'actionneur le plus proche de j).

Ce dernier lien (j, cj) peut ne pas exister. La métrique d'énergie sur ce lien est considérée pour tenir compte de l'avancement vers la destination à partir d'un voisin. La règle de deux sauts sélectionne j comme prochain saut et est appliquée alternativement. Il convient de noter que cette position est uniquement basée sur les informations locales de localisation. Il n'est exigé des nœuds que de connaître leurs voisins et les positions des actionneurs, à l'exclusion de tout échange d'informations.

Visiblement, puisque chaque étape du routage est indépendante des décisions antérieures, un paquet destiné à un actionneur donné généré par la source peut être transmis vers un autre actionneur de destination par un nœud intermédiaire appartenant au chemin de bout en bout. Pour cette raison, le collecteur transmet son identifiant dans le da-tree dans le but d'informer les capteurs sur sa propre identité. Les opérations exécutées par un capteur à l'état start-up sont décrites en détail dans Algorithme 1. La règle à 2 sauts assure l'inexistance des boucles dans les chemins.

Lors de la détection d'un événement ou la réception d'un paquet, le nœud vi exécute le code suivant.

Algorithme 1: Etat Start-up

---

mincost = ∞
Pour chaque voisin v_j **faire**
Début
  Pour chaque acteur s_k **faire**
  Début
    Si ($2E_{elect} + \beta d^{\alpha}_{ij} + 2E_{elect} + \beta d^{\alpha}_{jsk}$**)** < mincost **alors**
      mincost = $2Eelect + \beta d^{\alpha}_{ij} + 2E_{elect} + \beta d^{\alpha}_{jsk}$
      nexthop = v_i;
    FinSi
  End
End
Informer **nexthop.**

B. Etat Speed-up

L'objectif de l'état speed-up est de minimiser le nombre de sauts entre les sources et les actionneurs grâce à l'utilisation de Greedy Routing Scheme (GRS) [44]. Suivant GRS, chaque noeud envoie ses paquets au nœud voisin le plus proche de la destination. Intuitivement, cette règle minimise le nombre de saut dans le chemin, la distance parcourue par le paquet et le nombre de transmission du même paquet de données. Le pseudo code des opérations exécuté par le capteur à l'état speed-up est représenté par l'algorithme 2. L'ensemble pi représente l'ensemble des voisins de vi ayant une valeur Aicij positive.

Algorithme 2: Etat Speed-up

Pour chaque nœud $v_j \in P_i$ **faire**
Début
  Si (distance(v_i, v_j) > distance(v_i,nexthop)) **alors**
    nexthop = v_j;
  FinSi
End

C. Etat Aggregation

L'objectif de cet état est de réduire la consommation globale d'énergie. A cette fin, les capteurs prennent les décisions de routage qui permettent de réduire la consommation globale d'énergie, en s'appuyant sur l'algorithme **«data fusion»**. Cet algorithme est supposé être exécuté sur chaque capteur. Comme les paquets de données peuvent être regroupés par n'importe quel noeud dans le réseau, l'objectif donc de router les données au voisin le plus proche dans le da-tree.

Après la construction des da-trees, chaque capteur connaît son collecteur. En écoutant le canal partagé, chaque capteur connaît à quel da-tree appartiennent ces voisins. Donc, le nœud vi dans cet état doit d'abord évaluer le coût de la transmission des données à ces voisins qui appartiennent au même da-tree. De cette manière, il identifie le voisin à un coût minimum **vmin**. Nous insistons sur le fait que cela ne va pas engager des frais supplémentaires, autres que ceux qui sont causés par l'écoute des autres transmissions. Deux situations différentes peuvent se produire : Vmin peut être soit sur le même da-tree que vi, et donc ils sont associés au même collecteur, soit dans un autre da-tree.

Si v_{min} est dans le même da-tree que v_i, v_{min} ne peut être sélectionné comme prochain saut que si le chemin ne contient pas de boucle. Le choix de v_{min} garantit la minimisation du coût de transmission de tout l'arbre. Si un tel voisin n'existe pas, v_i en choisira un autre qui n'appartient pas forcement au même da-tree.

Dans ce cas, vi est autorisé à sélectionner vmin comme son prochain saut si et seulement si vi est une feuille dans son da-tree et vmin est plus proche de son actionneur que vi. Il apparaît clairement que si vi n'est pas une feuille, une boucle sera créée à cause des nœuds qui seront autorisés à commuter entre deux arbres. Les détails de l'algorithmes sont décrits comme suit :

Algorithme 3: Etat Agrégation

Pour chaque voisins v_j **faire**
Début
  Si (distance(v_i,v_j) < distance(v_i,nexthop)) **alors**
    v_{min} = v_j;
  FinSi
End
s = actionneur(v_{min})
Si (s = monActionneur) **alors**
  Si (distance(v_{min},s) < distance(v_i,s) alors
    Nexthop = v_{min}
  Sinon
    Supprimer v_{min} de la liste et réexécuté l'algorithme
  FinSi
Sinon
  Si vi Є Feuilles **alors**
    Nexthop = v_{min}
  Sinon
    Supprimer v_{min} de la liste et réexécuté l'algorithme
  FinSi
FinSi

D. les transitions d'état :

Les transitions des capteurs sont dirigées par les messages envoyés par les actionneurs. Les messages sont périodiquement envoyés à chaque actionneur. Cette période est égale à Δf secondes.

À chaque instant de décision k, le contenu du message sera déterminé en se basant sur trois mesures de fiabilité: la fiabilité r[k], la fiabilité à court terme $r_{sh}[k]$, et la fiabilité anticipée ŕ[k+1]. r[k] est observé durant le dernier intervalle de décision Δd. De la même manière, rsh[k] est la fiabilité observée Durant Δd_{sh}, avec $\Delta d_{sh} < \Delta d$.

En se basant sur r[k] et sur l'historique des mesures r[k.1], r[k.2],..., l'actionneur calcule ŕ[k +1] = f(r[k], r[k.1], r[k.2], ...). Le paquet contient la fiabilité (advertised) radv[k], est calculée à la base de ces trois mesures et elle sera effectivement envoyée si sa valeur dépasse rth2 ou est inférieure à rth1. Si le capteur ne reçoit pas de paquets, il déduit que la fiabilité est entre rth1 et rth2.

Ces calculs sont résumés dans l'algorithme 4. The advertised reliability radv[k] est calculé comme suit: en règle générale, advertised value radv[k] à n'importe quel instant k est égal à r̂[k + 1]. De cette manière, l'actionneur identifie les tendances de la fiabilité et réagit selon sa valeur.

Algorithme 4: transition d'état

$r_{adv}[k] = r[_{k+1}]$
Si ((r[k+1] > rth2) ou (r[k+1] <rth1)) **alors**
feedbackNeeded=true ;
FinSi
Si ((r[k+1] > rth2) et (rth1<r[k] <rth2) et (rth1<r_{sh}[k] <rth2)) **alors**
feedbackNeeded=false ;
FinSi
Si ((r[k] < rth1) et (rth1<r_{sh} [k] <rth2) et !(r[k+1] <rth1)) **alors**
feedbackNeeded=true ;
$r_{adv}[k] = r[k]$
FinSi
Si (r_{sh}[k] <rth1) **alors**
feedbackNeeded=true ;
$r_{adv}[k] = r_{sh}[k]$
FinSi
Si (feedbackNeeded=true) **alors**
Envoyer feedback($r_{adv}[k]$)
FinSi

E. le mode recovery

Les noeuds peuvent opérer dans deux modes différents : *greedy mode ou recovery mode*. Dans le premier mode, le noeud qui détient le message tente de l'envoyer à la destination et s'il échoue, il passe au mode recovery. C'est-à-dire que la sélection de n'importe quel voisin induira des boucles dans le chemin menant à la destination. Généralement, ceci apparaît lorsque le nœud observe une zone vide qui le sépare de la destination. Ces nœuds sont nommés *concave*. Les paquets reçus par de tel nœud entrent dans recovery routing mode et ne retournent au mode greedy que lorsqu'il sont reçus par un noeud plus proche de la destination que le noeud concave. Plusieurs schémas [45][46] basés sur *face routing dans un graphe plat* sont proposés pour résoudre ce problème mais l'inconvénient de telles solutions est qu'elles peuvent sélectionner le chemin le plus long [47].

Lorsqu'un paquet est reçu par un nœud concave, l'algorithme du mode recovery sélectionne un détour à droite ou à gauche suivant certaines lois. Le nombre de sauts du nouveau chemin construit par FACE-2 [46] est en moyenne trois fois plus grand que le chemin le plus court [47].

Le protocole proposé combinera FACE avec l'algorithme distribué. L'objectif est alors de:

- détourner le chemin en se basant sur FACE-2, ce qui garantit la délivrance;
- ajuster le chemin long en se basant sur la fiabilité observée par l'actionneur.

Les opérations en mode recovery sont décrites dans ce qui suit : Soit le capteur v qui génère ou reçoit un paquet et v est un nœud concave et appartient au chemin vers l'actionneur cv.

Tous les voisins w dont la distance qui les sépare de l'actionneur le plus proche cw est plus petite que la distance qui sépare v de son actionneur voisin cv, peuvent être considérés comme prochain saut. Si aucun nœud ne satisfait cette condition, v transmet le paquet vers cv à travers le chemin détourné (selon face-2) et il est donc en recovery mode.

Dans le mode recovery, les nœuds transmettent à puissance maximale pour permettre à leurs voisins d'entendre les transmissions et à chaque saut, le paquet portera un numéro. Ce numéro identifie la position du nœud dans le chemin détourné et représente la proximité virtuelle de l'actionneur de destination. En effet, le paquet transmis par le premier nœud concave aura le numero1.

A l'état recovery, le nœud v ignore les messages de l'actionneur. Cependant, le nœud v appartenant au chemin détourné écoute les paquets transmis par les nœuds voisins.

Dans le cas où v détecte le paquet de w destiné au même actionneur et que le paquet possède un numéro de proximité supérieur à celui de v, v annonce w comme étant le prochain saut et w fera partie du chemin détourné vers cv. En écoutant les autres transmissions, v construira la liste de ces voisins qui feront partie du chemin détourné ainsi que leur numéro virtuel. L'algorithme utilisé pour l'état Speed-up sélectionne le voisin le plus indiqué (celui dont le numéro est le plus grand) à la place du nœud le plus proche de l'actionneur tandis que l'algorithme exécuté à l'état d'agrégation sélectionne le voisin le plus proche.

II.3 Coordination Actionneur-Actionneur

II.3.1 La formulation du problème

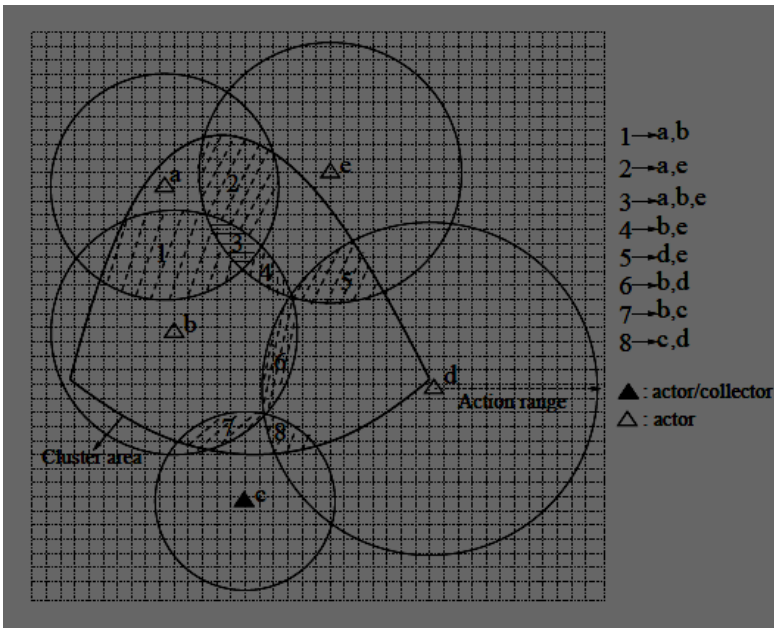


Figure 1.4 : zone de chevauchement pour le collecteur c.

L'objectif de la coordination actionneur-actionneur est de sélectionner le(s) meilleurs actionneur(s) pour un événement donné. Les actionneurs élus, ou les collecteurs, reçoivent les données à partir des capteurs sources qui définissent la zone d'événement. Chaque collecteur reçoit des données à partir d'un sous ensemble de sources (cluster). Sous la responsabilité d'un seul collecteur, chaque cluster prend en charge une partie des actions et de l'événement. Toutefois, un collecteur peut ne pas pouvoir réagir sur toute sa zone car sa gamme de transmission ne le permet pas. De plus, un actionneur peut ne pas être le plus approprié pour cette action en terme de temps d'accomplissement et/ou de consommation d'énergie.

Pour ces raisons, la coordination actionneur-actionneur est nécessaire avant d'engager l'action.

Définition : la latence limite pour la réaction est le délai maximal autorisé entre l'instant où l'événement est capté et celui où s'achève l'action.

Dans le cas où plusieurs actionneurs peuvent réagir dans une zone donnée, on parlera d'une zone overlapping, comme indiqué dans la figure 1.4, de 1 à 8. Le problème dans ce cas est celui du choix du meilleur ensemble d'actionneurs et de leurs tâches pour optimiser la consommation d'énergie et respecter les délais d'accomplissement des tâches. Dans le cas où seul un actionneur peut réagir, on parlera d'une zone de non-overlapping (les régions non barrées dans le schéma). Pour ce genre de zone, il s'agira de choisir le niveau de puissance de

l'actionneur qui minimise la consommation d'énergie et respecte le délai d'accomplissement des tâches. La question de la coordination est très importante dans le cas d'ovelapping.

Dans la section II.3.1.1, deux notions seront présentées: la zone de réaction et les modèles d'actionneurs, ce qui permettra de formuler en section II.3.1.2 le problème de coordination.

II.3.1.1 modèle d'actionneur

Les collecteurs reçoivent les données à partir des capteurs (sources). Les positions des nœuds sources permettent aux collecteurs d'identifier la portion de la zone d'événement dont ils sont responsables. Par référence à la figure 4, les notations suivantes sont présentées :

- Soit $\mathbf{S^A}$ l'ensemble des actionneurs avec $\mathbf{N_A = |S^A|}$.
- Soit $\mathbf{S^C}$ l'ensemble de collecteurs. On a donc: $\mathbf{(S^C \in S^A)}$.
- $\mathbf{A^h_{c,nov}}$ est la zone *non-overlapping* numéro $\mathbf{h^{th}}$ dirigée par c.
- $\mathbf{A^m_{c,ov}}$ est la zone *overlapping* numéro $\mathbf{m^{th}}$ dirigée par c. th
- **Hc** le nombre de zones non-overlapping.
- **Mc** le nombre de zones overlapping associées au collecteur c;
- $\mathbf{S^{A,m}_{c,ov}}$ est l'ensemble d'actionneurs qui peuvent réagir dans la zone overlapping $\mathbf{m^{th}}$ associée à c.

Chaque actionneur **a** possède les propriétés suivantes:

- **Ra [m]** représente les actions que peut exécuter **a**;
- $\mathbf{P^{Max}a}$ **[W]** est la puissance maximale que l'actionneur **a** peut utiliser pour exécuter l'action. Les actionneurs peuvent choisir leurs puissances parmi différents niveaux :

$$\mathbf{Pa,p = P^{Max}_a * P / L}, \qquad p = 1, 2, \ldots, L$$

- $\mathbf{Pa,p}$ est le niveau $\mathbf{p^{th}}$ de puissance propre à l'actionneur **a**. Un niveau de puissance élevée minimise le temps d'accomplissement des actions;
- η_a est l'efficacité de l'actionneur a;
- $\mathbf{E^{Av}_a}$ **[J]** est le niveau d'énergie de l'actionneur a, avec laquelle il peut réagir seul dans une zone non overlapping.

- **$P^{(m)}a,p$ [W]** est le niveau **p^{th}** de puissance de l'actionneur a pour la zone d'overlapping **m^{th}** dont les mesures sont égales à **$A^{mc}_{,ov}$ [m2];**
- **X(m)** est une matrice binaire et l'élément [x(m)a,p] est égal à un si et seulement si l'actionneur a réagit dans la zone overlapping Amc,ov en utilisant le niveau de puissance **P(m)a,p** ;
- **-T(m)a,p [s]** est le temps d'accomplissement de l'action par l'actionneur a, avec la puissance pth, seul et indépendamment de la zone **mth**.

$$T(m)a,p = K \cdot Amc,ov / [\eta a \cdot (P(m)a,p)\gamma a]$$

- **K[Wγa ·s/m2]** est une constante,
- **γa** est un paramètre qui varie entre (0, 1], et il définit la relation entre la puissance et le temps par rapport à l'actionneur a ;
- **ηa** est l'efficacité de l'actionneur;
- **δ [s]** est la bande limite d'achèvement de l'action qui dépend de l'événement et de l'application;
- **I(m)a** est égale à 1 si et seulement si mth est dans la porté de transmission de l'actionneur a, 0 sinon.
- **ha** est une variable binaire égale à 1 si et seulement si l'actionneur a est sollicité pour une action.

Le problème de la coordination actionneur-actionneur est formulé sous la forme d'un MINLP (Mixed Integer Non-Linear Program). Son objectif est de trouver, pour chaque partie de la zone d'événement, l'ensemble d'actionneurs qui minimise la consommation d'énergie globale et qui prend en compte la latence limite.

Il est supposé que :

- l'énergie nécessaire pour l'exécution des actions est beaucoup plus petite que l'énergie nécessaire à la communication.

- Les actionneurs sont capables de réagir sélectivement dans leur zone d'action, c'est-à-dire que si un actionneur est choisi pour réagir, que ce soit dans une zone d'overlaping ou non, ceci n'implique pas qu'il doit réagir dans la zone entière.

Le problème d'optimisation peut être ainsi formulé:

$\mathbf{P}_{Max}^{Res}$: **Residual Energy Maximization Problem**

$$Given: \quad N_A,\ L, M_c, E_a^{Av},\ T_{a,p}^{(m)}, I_a^{(m)}$$

$$Find: \qquad \underline{X}^{(m)} = [x_{a,p}^{(m)}],\ h_a \tag{18}$$

$$Maximize: \qquad E_{Avg}^{Res} = \frac{\sum_{a=1}^{N_A} h_a E_a^{Res}}{\sum_{a=1}^{N_A} h_a} \tag{19}$$

$$Subject\ to:$$

$$E_a^{Res} = E_a^{Av} - E_a^{Req} \geq 0,\ \forall a; \tag{20}$$

$$E_a^{Req} = \sum_{m=1}^{M_c} \left(\frac{\sum_{p=1}^{L} x_{a,p}^{(m)} P_{a,p}^{(m)}}{\sum_{a=1}^{N_A} \sum_{p=1}^{L} \frac{x_{a,p}^{(m)}}{T_{a,p}^{(m)}}} \right),\ \forall a; \tag{21}$$

$$\sum_{p=1}^{L} x_{a,p}^{(m)} \leq 1,\ \forall a,\ \forall m; \qquad \sum_{a=1}^{N_A} \sum_{p=1}^{L} x_{a,p}^{(m)} \geq 1,\ \forall m; \tag{22}$$

$$\frac{1}{\sum_{a=1}^{N_A} \sum_{p=1}^{L} \frac{x_{a,p}^{(m)}}{T_{a,p}^{(m)}}} \leq \delta,\ \forall m; \tag{23}$$

$$h_a \leq \sum_{p=1}^{L} \sum_{m=1}^{M_c} x_{a,p}^{(m)},\ \forall a; \qquad h_a \geq x_{a,p}^{(m)},\ \forall a,\ \forall p,\ \forall m; \tag{24}$$

$$x_{a,p}^{(m)} \leq I_a^{(m)},\ \forall a,\ \forall p,\ \forall m. \tag{25}$$

- La contrainte (20) garantit un résidu d'énergie non négative pour chaque actionneur.
- La contrainte (21) définit l'énergie nécessaire pour l'actionneur afin d'accomplir l'action dans la zone overlapping où il réside.
- La contrainte (22) assure que chaque actionneur utilise seulement un de ses niveaux de puissance et qu'au moins un actionneur réagit dans chaque zone d'overlaping. Notons que si plusieurs actionneurs réagissent en même temps, le temps d'accomplissement est réduit.
- La contrainte (23) limite le temps d'accomplissement pour qu'il respecte la latence limite pour chaque zone d'overlaping. Ce temps est exprimé comme suit:

$$\left(\sum_{\alpha=1}^{N} \sum_{p=1}^{L} \frac{X_{\alpha,p}^{(m)}}{T_{\alpha,p}^{(m)}} \right)^{-1}$$

- La contrainte (24) définit la relation entre x(m)a,p et ha,

- La contrainte (25) impose que chaque actionneur ne réagit qu'à l'intérieur de sa zone.

II.3.2 Le protocole de localisation à base d'intermédiaire

La solution est inspirée des agents dans un supermarché où l'allocation de ressource est le résultat d'une interaction entre le vendeur et le client. L'objectif de l'interaction est la sélection du meilleur ensemble d'actionneurs pour exécuter les actions appropriées dans la zone d'événement. Les actionneurs peuvent effectuer les tâches suivantes :

- *Le vendeur*. C'est l'actionneur responsable d'une zone d'événement c'est à dire le collecteur.

-*Auctioneer*. C'est l'actionneur chargé de sélectionner un actionneur approprié en qualité d'intermédiaire dans la zone d'overlapping. Un actionneur est choisi pour chaque zone d'overlapping par le collecteur responsable de cette zone.

-client. L'actionneur qui peut agir en particulier dans la zone d'overlapping peut être considéré comme un acheteur de cette région.

L'intermédiaire est présent dans chaque zone d'overlapping. Chaque actionneur communique au collecteur son niveau d'énergie et de puissance ainsi que le temps qu'il lui faut pour sélectionner les clients (c'est-à-dire le temps nécessaire à l'actionneur pour achever l'action dans toute la zone). L'objectif est de maximiser le total des revenus de l'équipe construite par les actionneurs participant à la réaction et dont le revenu dépend de l'énergie (Eresavg). Plusieurs sélections peuvent être déclenchées, en parallèle, sous la responsabilité de différents intermédiaires. Ce choix d'un intermédiaire par sélection se justifie par les raisons suivantes :

- Il provoque peu de charge de signalisation car des messages sont échangés entre les intermédiaires et les acheteurs pour une même zone d'overlapping, proche de l'intermédiaire. Les messages échangés seront localisés car ils ne sont pas dirigés vers le collecteur, qui peut être éloigné de la zone d'overlapping.

- La charge de sélection sera partagée parmi le maximum d'actionneurs, car le nombre d'intermédiaire est en général plus grand que le nombre de collecteurs.

Lorsque le vendeur c (le collecteur) reçoit les informations interceptées par les capteurs, il décide si une réaction est nécessaire dans sa zone. Il gère toutes les zones que ce soit en

overlapping ou en non overlapping, car il connaît la position et les actions des autres actionneurs.

Le problème de coordination réside dans les zones d'overlapping où plus d'un actionneur peut agir, tandis que dans les zones non overlapping, le collecteur assigne directement les charges à l'actionneur approprié. Le collecteur c sélectionne Mc intermédiaires, un dans chaque zone d'overlapping.

Soit s(m)ϵ S^A, l'intermédiaire élu, par le vendeur c, pour la zone mth. Cet intermédiaire est l'actionneur le plus proche du centre de la zone. Comme il est proche de chaque actionneur dans la région, l'énergie utilisée pour la communication ainsi que la latence seront réduites. Apres la sélection de l'intermédiaire, le collecteur lui communique les zone d'action et le temps d'accomplissement de l'action ainsi que le temps limite pour la répartition des charges. Ce message assistera l'intermédiaire dans la répartition des tâches. Au début de la réaction, l'intermédiaire envoie le message JOIN AUCTION à tous les clients dans cette zone. Les clients recevant ces messages transmettent leur énergie disponible ainsi que leur position. L'intermédiaire élu sélectionne ainsi les clients appropriés pour exécuter les actions attendues.

II.4 Analyse des performances

Au niveau de la coordination capteur-actionneur qui assure un compromis entre la consommation d'énergie et la latence, l'analyse de performance présentée dans [28] distingue deux sortes de configuration existent:

Configuration statique : A un moment donné, tous les noeuds sont dans le même état: start-up, speed-up ou agrégation.

Configuration dynamique : Les transitions seront considérées. Chaque nœud est dans un état indépendamment des autres.

Consommation d'énergie dans une configuration statique :

Elle signifie l'énergie nécessaire pour transmettre un bit de la source à un actionneur. La solution optimale est indépendante de la taille de la zone d'événement car :

- Le nombre de sources augmente lorsque la taille de la zone augmente, ce qui accroît l'énergie consommée.
- Comme plusieurs nœuds seront impliqués, l'agrégation optimisera la quantité d'information.

Dans la configuration start up et speed up, la consommation d'énergie augmente considérablement avec la taille de la zone d'événement. Ceci peut être compensé par l'état d'agrégation.

-Le passage à l'état d'agrégation à partir de l'état start-up optimise bien plus l'énergie qu'à partir de l'état speed up ;
-L'énergie consommée à l'état d'agrégation est deux fois inférieure à celle consommée à l'état start-up.
-L'énergie consommée augmente avec le nombre de capteurs.
-La consommation d'énergie à l'état speed up est plus grande que celle de l'état start-up.
Si le système passe de l'agrégation vers le speed up, on aura une réduction du nombre de sauts égale à 5 pour chaque paire capteur-actionneur. Cette réduction est de deux, si le système transite de l'agrégation à speed up.

Configuration dynamique :

Immédiatement après le passage à l'état start-up, la fiabilité observée par l'actionneur augmente car suite à l'avertissement de l'actionneur, un grand nombre de capteurs passent à l'état speed up. Par contre peu de capteurs passent à l'agrégation. Apres cette oscillation, la fiabilité se stabilise à une valeur donnée.
-Le nombre de capteurs impliqués dans cette phase de coordination se stabilise avec le temps.
-La simulation montre aussi que 95% des paquets reçus sont fiables.
-Le délai de délivrance dépasse la latence limite durant les périodes de manque de fiabilité.
Concernant la coordination actionneur-actionneur, on distingue trois approches : optimale, à un seul actionneur et à base d'intermédiaire.

- Approche optimale : en résolvant le problème mathématique.
- Un seul actionneur : l'action est exécutée par un seul actionneur dans une zone overlaping.
- Localisation à base d'intermédiaire : c'est le protocole proposé dans ce travail.

Localisation à base d'intermédiaire consomme une énergie proche de celle consommée par la solution optimale car chaque auctionneur calcule sa solution optimale séparément pour sa zone d'overlaping. De plus, ce protocole exploite l'efficacité de certains actionneurs pour disperser l'énergie nécessaire à l'achèvement des actions.

II.5 Conclusion :

Le framework de coordination capteur-Actionneur et actionneur-actionneur a été ainsi présenté avec une solution optimale pour chaque type de coordination. Compte tenu du problème de convergence qui se rattache à ces solutions, deux protocoles avec des performances proches de la solution optimale ont donc été proposés. Egalement, a été abordée la manière dont ces protocoles assurent un compromis entre la consommation d'énergie et le respect de la latence pré-déterminée.

Chapitre III : Le framework de communication temps réel pour les WSAN

III.1 Introduction:

Contrairement aux WSNs, les WSANs sont conçus pour répondre rapidement aux exigences des environnements réactifs. Dans ces environnements, l'automatisation des mécanismes de réaction est indispensable pour éviter les interventions externes, coûteuses en terme de latence. A cet effet, les WSAN déploient des actionneurs qui réagissent aux changements de l'environnement. Les fonctionnalités des actionneurs sont spécifiques à une application mais il est important de noter que le processus de réaction est le même. Les tâches exécutées par les actionneurs ne sont déclenchées qu'après la détection des événements par les capteurs.

Sachant que le déclenchement des actions dépend de l'instant où l'actionneur détecte l'événement, une contrainte importante pour la communication capteur-actionneur est ainsi constituée par la fiabilité ou la latence. Or, du fait principalement de fréquentes pannes des capteurs et de la latence consécutive à des congestions ou à l'agrégation des données, la communication dans les WSAN est peu fiable.

Dans ce chapitre, sera étudié un framework générique [29] [48] qui vise à maximiser la fiabilité du réseau WSAN. Cette fiabilité est étroitement liée à la latence, ou en d'autres termes à la fraîcheur des paquets reçus par les actionneurs. Cependant, les pannes et les congestions entraînent des pertes incontournables de paquets. Ce framework favorise les paquets les plus importants. Chaque capteur utilise des files de priorité permettant de classer les paquets de données par priorités. Ces files constituent un critère important pour le routage. Le routage des paquets ne doit en aucun cas influer sur la latence des paquets les moins prioritaires. Outre le routage qui est l'élément le plus important dans un framework de coordination, on distingue l'agrégation qui permet d'éliminer, le plus tôt possible, toute donnée inconsistante provenant de nœuds défaillants ou d'une congestion dans le réseau. Les données éliminées n'influent en aucun cas sur le routage des paquets consistants. De plus,

l'agrégation se base sur un schéma hiérarchique permettant d'envoyer plusieurs données consistantes dans un même paquet, évitant ainsi une consommation d'énergie inutile tout en assurant la minimisation de la latence globale du système. Par ailleurs, pour une meilleure répartition de charge, ce framework repartit équitablement les actionneurs en fonction des fréquences pre-estimées des occurrences des événements. Pour maximiser l'indice de fiabilité, le framwork intègre quatre modules clés:

- un module reporting qui subdivise le réseau en clusters afin de construire l'arborescence nécessaire à l'étape d'agrégation.
- un module d'agrégation de données multi niveau tolérant aux erreurs induites par les pannes des capteurs ;
- un protocole de transmission basé sur la priorité des événements.
- un algorithme d'allocation d'actionneur qui distribue intelligemment les actionneurs en répondre aux demandes provenant des capteurs.

Ces modules et leurs interactions seront détaillés en première partie avant que ne soit évaluation le framework.

III.2 Principe général du framework:

Les WSAN sont constitués d'une collection de capteurs **s** et d'actionneurs **a**. La zone d'événement est subdivisée en **grid** de tailles égales pour la gestion d'événement. Il est supposé que les nœuds connaissent leurs positions et donc leur **grid.** L'information de localisation peut être obtenue par un système GPS [49] ou par d'autres techniques [50] [51] [52]. Chaque capteur est responsable de la collecte des données d'événement de sa propre **grid**. Comme le mauvais fonctionnement des capteurs peut produire des lectures inconsistantes, les données d'une même **grid** seront regroupées (agrégation) de manière à s'assurer de leur cohérence. Apres détection d'un nouvel événement, les capteurs entament la construction d'arborescences d'agrégations à base de clusters. L'agrégation se fait en mode distribué et chaque flux de données sera dirigé vers le noeud de rapport **c** (cluster head). Un sous-ensemble de capteurs v communique l'information d'événement aux actionneurs. Ces nœuds communiquent directement avec les actionneurs et le rapport aboutira dès sa réception par l'un des actionneurs.

Les actions dévolues aux actionneurs sont spécifiques à l'application. Il est, toutefois, intéressant de noter que dans la majeure partie de ces applications, la fiabilité parfaite, comme dans TCP, n'est pas nécessaire. Elle peut même s'avérer impossible eu égard aux erreurs

produites en phase d'agrégation et de transmission. En outre, lorsque les informations parviennent rapidement aux actionneurs, le temps de réponse sera réduit et les décisions seront plus précises étant donné la fraîcheur des données.

L'indice de fiabilité mesure la probabilité que les données d'événement seront agrégées et reçues précisément dans un délai de latence pré-défini. La variabilité et l'ordre de transmission des paquets répondent à des paramètres tels que : leur type, leur urgence, leur gravité de l'événement. Aussi, chaque capteur maintient une file d'attente de priorité qui constitue également un critère de sélection du prochain saut. Lors de la transmission, les données des événements les plus importants sont prioritaires.

III.3 Objectif du framework

L'objectif principal de ce framework est de maximiser la fiabilité laquelle dépend du nombre de paquets bien reçus dans les délais prévus. La description formelle des paramètres du système nous permettra de définir la fonction objective à maximiser l'index de fiabilité **R** pour l'ensemble des événements. On distingue les variables suivantes :

- **e**: est un événement.
- **qe:** un paquet de données de l'événement **e.**
- **Qe**: un ensemble des paquets de l'événement e qui satisfait la contrainte temps réel.
- **Imp(e)**: l'importance de **e**.
- **Be**: le délai maximal pour que les informations de l'événement **e** arriveront aux actionneurs.
- **Dqe** le temps d'arrivée de **qe**
- **Ne**: le nombre de paquets de données propres à l'événement **e**
- **f**: la probabilité d'échec de l'agrégation des données.

Fonction objective: cette fonction doit maximiser **R.**

$$\mathbf{R = \Sigma_e\ Imp(e) * r_e\ ,}$$

$$\mathbf{Où :\ r_e =} \frac{Qe*(1-f)}{Ne} \mathbf{\ et\ Dqe < Be}$$

Plus précisément, la fiabilité du système **R**, est fonction de l'importance des événements et de leur fiabilité respective. Elle dépend du nombre de paquets reçus par un actionneur dans les délais et sans défaillance d'agrégation. L'échec de l'agrégation ne se produit que si un mauvais fonctionnement des capteurs domine une grid.

III.4 FRAMEWORK

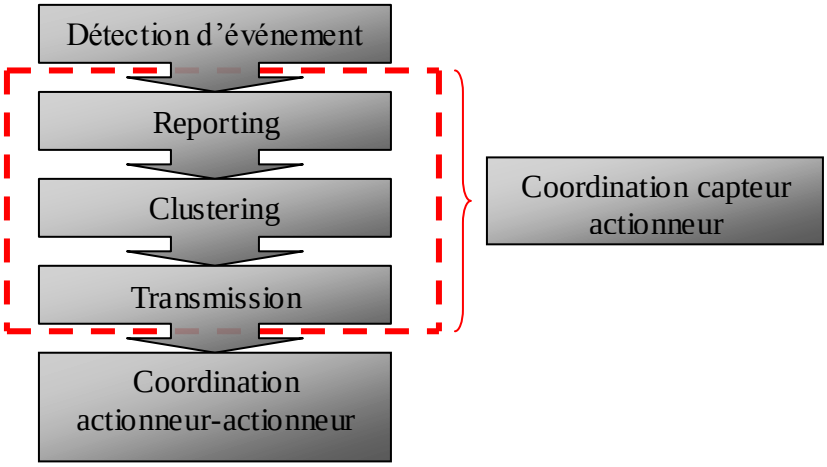


Figure III.1 : les modules du framework.

Ce framework traite le processus de coordination et intègre quatre modules afin de prendre en compte la contrainte temps réel. Lors de l'occurrence d'un événement, les capteurs les plus proches le détectent et entament le clustering. La zone d'événement sera divisée selon le cluster formé en **phase reporting**. A la fin de cette phase, chaque capteur doit appartenir à un et un seul cluster. Les capteurs qui détectent plus tôt l'événement sont les clusters head. Les clusters sont construits à base d'événement pour éviter d'inclure, dans le routage, des nœuds qui n'appartiennent pas à la zone d'événement. Chaque cluster head est responsable de la collecte des données de son propre arbre. Les branches de ces arbres constituent des nœuds d'agrégation. L'agrégation élimine les redondances et l'inconsistance des lectures. Après la phase d'agrégation, un sous-ensemble de capteurs communique l'information aux actionneurs. Les actionneurs n'attendent pas toute l'information pour commencer leurs interventions, ce qui minimise la latence.

Ce framework utilise un protocole de routage géographique. Le protocole de routage peut contrôler certains paramètres du système pour s'adapter aux conditions du réseau comme le niveau énergie. A titre d'exemple, le délai de transmission peut être considéré lorsqu'un nœud sélectionne son voisin. Dans ce framework et pour la transmission des paquets, chaque capteur doit prendre en compte le délai de transmission et l'importance des données. Les paquets sont classés dans une file de priorité qui sera utilisée par le mécanisme de routage et les données seront transmises par ordre de priorité.

En outre, le framework impose une structure de grid virtuelle dans le réseau et donc la représentation simplifiée à base de coordonnées peut être appliquée pour faciliter la formation et la combinaison des clusters. La zone d'événement est ainsi subdivisée en grid de taille égale. Ce framework fournit aussi un module d'allocation qui détermine la localisation des actionneurs. La répartition des tâches entre les actionneurs doit équilibrer les charges et minimiser la latence. Dans ce qui suit, les descriptions détaillées des quatre modules sont présentées.

III.4.1 Construction des « grid »

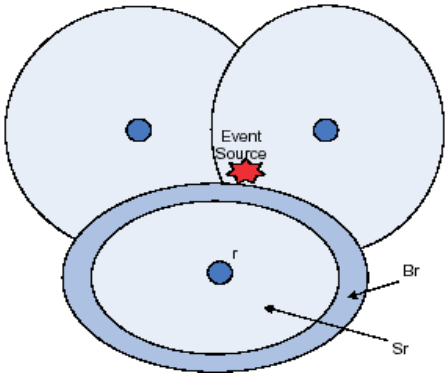


Figure III.2 formation des grid (map).

Le capteur qui détecte en premier l'événement commence la construction de son propre cluster. Les clusters divisent la zone d'événement en pièce «map», comme le montre la figureIII.2. Soit r le capteur qui détecte le premier l'événement. Il commence la construction de son cluster selon l'algorithme indiqué ci dessous. L'ensemble des noeuds v voisin de r diffuse ce même message de détection aux noeuds voisins de r d'au moins d'un saut maximal. Seulement, les nœuds qui détectent l'événement et n'ont pas encore informé les actionneurs rediffusent ce message à leurs voisins. On représente le noeud appartenant au cluster formé par le noeud r par Sr. Plusieurs r peuvent exister pour un même événement. Le nœud r sera donc le cluster head et rapportera à l'actionneur. Br est la limite du cluster de r où Br est incluse dans Sr. Les nœuds dans Br sont soit à un saut maximal du nœud r ou localisé dans une zone de chevauchement de deux clusters. Les noeuds appartenant à Br ne transmettent pas le message de détection d'événement et répondent aux noeuds qui les précèdent. Cette réponse contient les coordonnées, les valeurs de données et l'identité de l'événement, cette dernière pouvant inclure le type d'événement et le temps de découverte déjà déterminé par r.

Algorithme 1: construction des grid.

Pour chaque nœud r détectant un événement **faire**
Si r n'a pas entamé l'agrégation **alors**
Diffuser EvtDetect(r, 0, e) aux voisins de r ;
FinSi
Fin

Pour chaque nœud v recevant le message EvtDetect(r, 0, e) a partir de v` **faire**
Si (h < max_{hop} et (v.event et !v.reported)) **alors**
Envoyer EvtDetect(v, h+1, e) aux voisins de v
Sinon
Répondre EvtRep(meets boundaty) au v`
FinSi
Fin

Pour chaque nœud v recevant le message EvtRep(meets boundaty) **faire**
Si (message = meets boundaty) **alors**
Répondre EvtRep(x_v,y_v,$data_v$,e) au parent
Sinon
Regrouper les paquets de données et répondre EvtRep au parent
FinSi
Fin

III.4.2 Agrégation des données à base de Grid

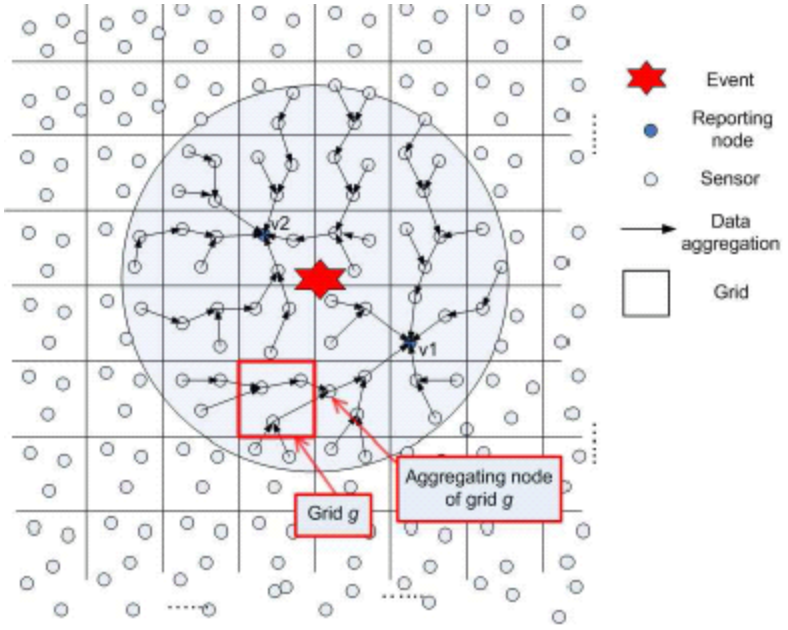


Figure III.3 : schéma d'agrégation.

Chaque partie du cluster peut être représentée par un arbre dont r est la racine comme le montre la figure III.3. Lorsqu'un noeud reçoit une réponse à partir de ses descendants dans l'arbre, il regroupe les réponses et les dirige vers son père. Les nœuds situés à une profondeur à nombre impair doivent effectuer l'agrégation des données de leurs voisins immédiats avant de les rediriger. Par contre, les noeuds dont la profondeur est un nombre pair assemblent les données de la coordination et les informations captées. Le capteur r rapporte, enfin, toute donnée collectée dans sa propre région à l'actionneur approprié. Cet actionneur n'est pas nécessairement celui qui va réagir, l'élection de l'actionneur approprié étant le fait de la coordination actionneur-actionneur.

Dans un réseau de capteurs denses, plusieurs capteurs peuvent ressentir le même événement avec les mêmes lectures. Il est donc préférable d'agréger les données avant de les envoyer. L'algorithme d'agrégation fonctionne de la façon suivante (voir figure 2):

Pour chaque grid, un noeud d'agrégation collecte les données, **<XI, X2, ..., Xn>,** afin de trouver une médiane **med**. Il compare chaque xi avec **med** et filtre celles qui ont une différence significative (exemple, plus grande que le coût prédéfini **Δd**). Ce nœud vh qui se situe à une profondeur impaire de h, calcule **med** de la manière suivante :

$$mean_{v_h} = (\sum_{j=1}^{c_{v_h}} data_{v_{hj}} + data_{v_h})/(c_{v_h} + 1), \qquad (3.1)$$

Où :

- **h** : est le nombre de saut à partir de *r*,
- v_h : le noeud à une profondeur h,
- $data_{vh}$: les données collectées par *vh*,
- $data_{vhj}$: *les* données collectées par les j^{eme} descendant de *vh*.
- c_{vh} est le nombre des descendants immédiat de *vh*.

Les données éliminées proviendront des capteurs inconsistants et seront donc ignorées. Ensuite, le nœud d'agrégateur calcule une moyenne **Xg** des données appartenant à g. Les données agrégées sont fiables si plus de la moitié des capteurs de g ne sont pas défaillants. La fiabilité des données agrégées de g peut donc être évaluée comme suit :

$$1 - fg = 1 - \sum_{i=Nx/2}^{Nx} \binom{Nx}{i} (fs)^i (1 - fs)^{Nx-i}$$

- ***fg*** est la probabilité d'échec de l'agrégation des données dans **g ;**
- **N** est le nombre de nœuds dans **g** ;
- ***fs*** est le taux des capteurs en pannes et
- **Ng** est le nombre des rapports à transmettre.

Le nœud d'agrégation peut servir comme nœud de rapport pour réorienter les données d'événements vers l'actionneur. Cependant, l'agrégation peut être étendue à plusieurs niveaux où un nœud de rapport est responsable de la collecte et de l'agrégation des données des blocs voisins déjà agrégés comme le montre la (Figure III.2). Dans le cas de deux niveaux, chaque capteur décide seul en fonction de la probabilité **Pv** d'être ou non un nœud de rapport**.**

$$Pv = \frac{1}{Nx * Ng} \ ;$$

Notons que chaque **grid** possède une seule moyenne, donc **Ng** est égale au nombre de **grids** à rapporter par le capteur. D'autres algorithmes pour la sélection des nœuds capteur/collecteur pourraient être utilisés ceux de [53], par exemple.

Algorithme 2: agrégation des données.

Pour chaque donnée recu par le capteur s à partir de x_i **faire**

Si r n'a pas entamé l'agrégation **alors**

Si plusieurs $x_i \in g$ et s est un nœud d'agrégation **alors**

Calculer med parmi < x_1 , x_2 , x_3 ,…., x_n >

Pour chaque x_i Є g **faire**

Si x_i – med > Δd **alors**

Blacklisté le nœud i ;

FinSii

Fin

X_g = la moyenne des données non blacklisté x_i Є g

FinSi

Finsi

Fin

III.4.3 Rapport des événement selon les priorités

Le cœur de ce framework est constitué par les protocoles de routage et de transmission des rapports aux actionneurs, dans le but d'une part de maximiser le nombre de rapports reçus dans les délais prédéterminés et, d'autre part, de prioriser les événements les plus importants. En conséquence, les files d'attentes des priorités de chaque capteur revêtent une importance double :

1. accélération de la transmission des données des événements les plus importants;
2. sélection des routes afin d'observer les délais et satisfaire les latences.

Dans cette file d'attente préemptive, les paquets de données sont placés en fonction de leur importance en utilisant la politique FIFO. Comme un léger réseau de capteurs avec une occurrence rare d'événement ne souffre pas de l'excès de latence de transmission, cette étude se focalise sur les réseaux à événements fréquents. Dans de tels réseaux, le temps de séjour d'un paquet au niveau de l'émetteur domine le temps de traitement et de propagation.

Le temps de séjour d'un paquet au niveau de la file la plus prioritaire est $d_{q1} = R + SN_{q1}$, où $\overline{R} = \frac{1}{2}\sum_{k=1}^{K} \lambda_k \overline{S^2}$

- est le temps moyen de séjour dans la file.
- Nq1 est le nombre moyen de paquets dans la première file,
- k est le nombre des files d'attentes,
- l est le taux d'arrivé des paquets dans la file numéro k,
- s et s est le temps moyen de service, le second moment de service.

S peut être obtenu pour chaque capteur en observant le temps qu'il prend pour servir un paquet. Selon le théorème de little, $\overline{N_{q_1}} = \lambda_1 \overline{d_{q_1}}$ La charge imposée à k est égale $\rho_k = \lambda_k \overline{S}$, donc le temps d'attente du capteur k dans la première file est : $\overline{d_{q_1}} = \frac{\overline{R}}{1-\rho_1}$.

Le temps d'attente dans la deuxième file est :

$$\overline{d_{q_2}} = \overline{R} + \overline{SN_{q_1}} + \overline{SN_{q_2}} + \overline{S}\lambda_1 \overline{d_{q_2}} = \frac{R + \rho_1 d_{q_1}}{1-\rho_1-\rho_2}.$$

Le temps d'attente moyen dans la k^{eme} file est :

$$\overline{d_{q_k}} = \frac{\overline{R}}{(1-\rho_1-\ldots-\rho_{k-1})(1-\rho_1-\ldots-\rho_k)}.$$

Les capteurs échangent périodiquement des messages de contrôles entre eux. Ces messages contiennent plusieurs renseignements tels que le temps d'attente. Lors du routage des paquets de données, un capteur ne doit pas choisir un nœud occupé par la transmission des données importantes. Au contraire, il sélectionne celui dont le temps d'attente est le plus petit.

Considérons le noeud **i** qui reçoit les données de l'événement **e** à travers le message envoyé par **j**. Le noeud **i** obtiendra les informations suivantes:

- **A**: l'actionneur de destination.
- **S**: the expected service time of node j,
- $\mathbf{A_{high}}$: $\sum_{k, imp(data_k) \rangle imp(data_e)} \lambda_k$ est la somme de tous les paquets reçus λ_k qui sont plus important que **e**.
- $\mathbf{A_{low}}$: $\sum_{k, imp(data_k) \langle imp(data_e)} \lambda_k$ est la somme de tous les paquets λ_k qui sont moins importants que **e**.

Le nœud i doit s'assurer que la latence des données de l'événement **e** est inférieure à une latence limite Be. Pour cela, il estime tout d'abord, la distance **hij** à partir de i à j vers l'actionneur **a**. ensuite, il doit estimer le délai max de i à j.

$$hij = \frac{\|a,i\| - \|a,j\|}{\|a,i\|} \text{ avec } delay_{i,j} <= Be * hi,j$$

$delay_{i,j} = d_q + d_{trans} + d_{prop} + d_{proc}$

donc: $d_{qmax} = Be * hi,j – (d_{trans} + d_{prop} + d_{proc})$;

Les voisins pour lesquels $d_{qmax} > 0$ peuvent être considérés comme prochains sauts afin de ne pas dépasser la latence limite. Le voisin choisi pour transmettre le paquet doit avoir $\mathbf{A_{high}} = 0$ et $\mathbf{A_{low}}=0$.

-$\mathbf{A_{low}}=0$ entraîne que le voisin n'a pas de paquets d'importance inférieure à celles des paquets de i; donc, le paquet de i n'a pas à modifier le temps d'attente des autres paquets.

- $\mathbf{A_{high}=0}$ implique que le voisin n'a pas de paquets d'une importance supérieure à celles des paquets de i; donc, le paquet de i n'a pas a rester dans la file d'attente.

Pour chacun des noeuds satisfaisant aux trois conditions précédentes: le capteur calculera le nombre maximal de paquets qu'il peut transmettre tout en satisfaisant la contrainte temps réel :

$$\rho_{i,j} \langle 1 - \lambda_{higy} * \bar{S} - \frac{\bar{R}}{(1 - \lambda_{higy} * \bar{S}) * d_{q\max}}$$

Où i,j = Ai, S is the maximum affordable load of j for handling data from i on event e.

Les paquets qui seront envoyés au voisin sont ceux dont les valeurs hij et aij sont les plus grandes car les plus proches de la destination. Chaque nœud intermédiaire modifie la latence limite du paquet Be.

$Be=Be- (t_{depart} + t_{arrives})+ d_{tran} + d_{prop}$

- $t_{depart} - t_{arrives}$ est le temps de la cœxistence du paquet au niveau du nœud.
- d_{tran} peut être calculé en utilisant la puissance de transmission et la longueur de la trame de données.
- d_{prop} est le temps de propagation.

Apres la transmission, le capteur modifie S ainsi que le chemin du routage, de manière à vérifier que la contrainte temps réel est assurée. Dans le cas où elle ne l'est pas, il change son prochain saut si celui-ci existe. Sinon, il informe le nœud précèdent pour que celui-ci change de chemin.

III.4.4 Allocation des actionneurs

Lorsqu'un actionneur reçoit le rapport d'événement, il informe les autres actionneurs pour renforcer leur potentiel d'action. Cette coordination peut être assurée à l'aide de la

communication à un saut et dans un canal sans fil spécifique. Il est clair que la réduction des distances entre les capteurs et les actionneurs minimisera la latence.

L'intervention des actionneurs est déclenchée à la faveur un événement et elle dépend de la fréquence de l'occurrence des événements. Les zones à événements multiples doivent être dirigées par plusieurs actionneurs de manière à réduire la distance du rapport. L'allocation des actionneurs peut être opéré à l'état initial avec des fréquences pre-estimées ou périodiquement avec des actionneurs mobiles. Pour équilibrer les charges entre les actionneurs, les fréquences freqg de chaque grille g seront d'abord additionnées. Ensuite, le terrain A sera également divisé en deux, noté par Al et A2, en fonction de la fréquence de distribution. Si Al et A2 ont la même, fréquence chaque zone aura donc la moitié des actionneurs. Le processus se répète récursivement pour Al et A2, jusqu'à ce que chaque sous-champ contienne un seul actionneur. Voici les détails de l'algorithme :

Algorithme III: allocation des actionneurs.

ActuatorAllocation (Field A, int ActuatorNum)
TotalFreq←∑P $freq_{gi}$
TmpFreq ←0;
i←0;
Pour chaque TmpFreq < TotalFreq/2 **faire**
TmpFreq←TmpFreq+ $freq_{gi}$;
i++ ;
Fin
A1← i k=0 gi;
A2←A-A1;
ActuatorAllocation(A1,ActuatorNum/2);
ActuatorAllocation(A2,ActuatorNum-ActuatorNum/2);
FIN

La figure III.4 démontre les résultats de l'allocation avec 6 actionneurs. L'algorithme peut être exécuté après la collecte des fréquences d'événements. Le sink informe ensuite les actionneurs qui peuvent ensuite changer de localisation.

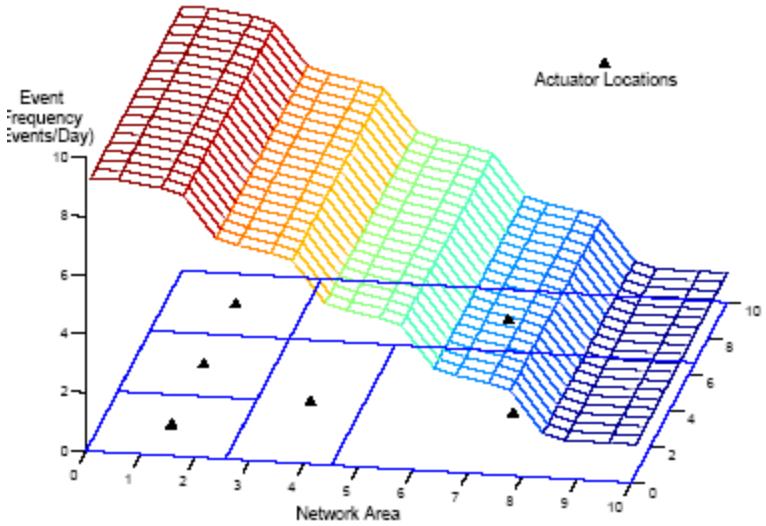


Figure III.4: allocation des actionneurs.

III.5 Evaluation de performances :

Les résultats de simulation présentée dans [29], comparent deux protocoles :

-le protocole PREI qui prend en compte la priorité des événements.

-le protocole de routage géographique GRP.

La comparaison comporte deux phases : la phase de coordination capteur-actionneur (fiabilité durant la phase de rapport) et la phase d'allocation d'actionneur. Elle prend en compte le délai de chaque phase :

- Le délai par nombre de saut.
- Le délai par temps de réponse.

A. fiabilité du rapport

-le nombre de sauts pour atteindre un actionneur est plus petit pour les événements les plus prioritaires dans PREI alors que GRP ne différencie pas les événements.

-les événements d'une même importance n'effectuent pas le même nombre de sauts pour atteindre l'actionneur car cette opération dépend de la distance qui les sépare des actionneurs.

-les résultats montrent que le temps moyen pour atteindre l'actionneur est plus réduit dans PREI et qu'il ne dépend pas de l'importance de l'événement mais de la proximité de l'actionneur.

B. Allocation des Actionneurs

La simulation prend en compte deux scénarios différents : avec ou sans stratégie d'allocation.

-La fiabilité dans le premier scénario est plus respectée.

-Plus le nombre d'actionneurs augmente plus la fiabilité est assurée.

-La stratégie avec allocation de 2 actionneurs est presque la même que celle de trois actionneurs sans allocation et parfois elle est plus efficace si le nombre d'événement diminue.
-Le délai de délivrance des paquets en fonction du nombre d'événements indique les mêmes priorités.

III.6 Conclusion

L'approche adoptée dans ce chapitre privilégie le critère de fiabilité dans les WSANs. Nous avons fait valoir, dans ce cadre, que cette fiabilité est étroitement liée au retard, ou à la fraîcheur des événements qui devraient être conjointement optimisés.

Nous avons également suggéré par ailleurs, que la non-uniformité de l'importance de ces événements peut être explorée pour une plus grande optimisation. Pour maximiser l'indice de fiabilité, ce framework intègrent les quatre modules:

1) le schéma d'agrégation de données multi-niveau qui est tolère les pannes des capteurs ;

2) protocole de transmission avec priorité,

3) l'allocation d'actionneur qui distribue les actionneurs selon les demandes des capteurs.

Ces modules sont présentés de même que leurs interactions. En dernière partie, les résultats des simulations présentées démontrent que ce framework représente une utilisation efficace des actionneurs et qu'il contribue de façon significative à améliorer la fiabilité du système.

Chapitre IV : la réallocation des actionneurs.

IV.1 Introduction

La distribution des robots pour la couverture complète d'une zone donnée représente l'un des défis majeurs du domaine multi robotique. En effet, les différents robots qui constituant ces réseaux sont mobiles et la question qui se pose est celle de leur répartition de manière à ce que tous les points de la région soient à l'ombre d'un robot [54]. Du fait de leur nombre limité à l'intérieur d'une vaste zone, les robots se déplacent donc continuellement pour couvrir l'ensemble du répertoire.

Dans les WSAN [55], très peu de travaux de recherche traitent de ce problème de couverture que les WSAN connaissent pourtant du fait de la faiblesse du nombre d'actionneurs par rapport au nombre de capteurs. Le problème qui se posé le plus dans les WSAN est le moyen avec lequel les capteurs et les actionneurs coordonnent entre eux afin de minimiser la latence et la consommation d'énergie.

Dans ce cadre, la majeure partie des solutions préconisées tendent à regrouper les nœuds selon leurs déploiements initiaux. Les actionneurs entament l'exécution d'un processus de découverte des voisins afin d'établir le contact avec l'ensemble des capteurs; cette phase servira à la formation des clusters collaboratifs. Pour chaque cluster, il y a un actionneur désigné comme cluster head qui rassemblera et traitera les données provenant des capteurs à l'intérieur de son cluster. Chaque capteur déterminera son prochain saut dans son propre cluster qui minimisera la latence et l'énergie. Les actionneurs ayant reçu les données d'événement amorceront la phase de coordination inter-actionneurs pour répondre aux exigences des applications.

Dans les WSAN, le positionnement des capteurs dépend de l'application et des portions qu'ils doivent contrôler. Les actionneurs sont des nœuds qui doivent agir rapidement et prendre les décisions les plus appropriées. Une solution très intéressante consiste à les placer au bon endroit pour qu'ils puissent recevoir l'information le plus tôt possible, tout en préservant les ressources énergétiques des capteurs.

La répartition des actionneurs constitue donc un des problèmes de conception dans les WSANs. L'intérêt que présente les solutions de distribution des actionneurs est du au fait que leur emplacement affecte énormément la réalisation des exigences du système et se répercute sur les performances du réseau. Par exemple, il est souhaitable, dans de nombreuses configurations, d'avoir une couverture maximale de l'ensemble de la zone d'intérêt. Par conséquent, il peut être obligatoire d'avoir autant que possible les actionneurs répartis uniformément dans la région. En outre, dans de nombreuses applications, il est important de réagir rapidement aux nouveaux événements, que ce soit par l'exécution des tâches spécifiques que par la réalisation d'expériences sophistiquées pour recueillir des données plus précises ou même se rapprocher de l'endroit où l'événement s'est produit. Ainsi, en plus de la couverture, l'attention doit être soulignée sur le nombre de sauts qu'un paquet de données doit parcourir jusqu'à atteindre un actionneur.

Avoir une couverture d'actionneur tout en réduisant la latence est un vrai challenge. Dans la pratique, la répartition des capteurs n'est pas uniforme dans toute la zone. Ainsi, placer les actionneurs sur une vaste zone conduit à une topologie du réseau inefficace, ce qui obligera les capteurs à emprunter des chemins de données plus longs. Par exemple, si les capteurs sont à forte densité de population à proximité du centre de la région de déploiement, alors que les actionneurs sont loin du centre, les données seront donc acheminées vers ces actionneurs sur de longues voies. Un autre problème qui peut se poser si plusieurs capteurs sont auprès d'un seul actionneur, est la surcharge de cet actionneur qui ne lui permettra pas de traiter toutes les données d'événements.

Ces deux scénarios ne sont pas acceptables. La surcharge des actionneurs peut conduire à une perte de données et à une incapacité à traiter les données rapidement, ce qui aura un impact négatif sur la demande. D'autre part, l'acheminement des données sur de longues voies augmente la latence, induit une consommation énergétique inutile et limite le domaine de réaction des actionneurs.

Le but de ce chapitre est donc de présenter les différents protocoles qui sont utilisés pour l'allocation des actionneurs aussi que leurs avantages et leurs inconvénients. Le protocole proposé dans [56] ne suppose pas la mobilité des actionneurs et garde les actionneurs au niveau de leurs emplacements initiaux. Cependant, il sépare les zones couvertes par les actionneurs pour éviter la duplication des informations. Il suppose la

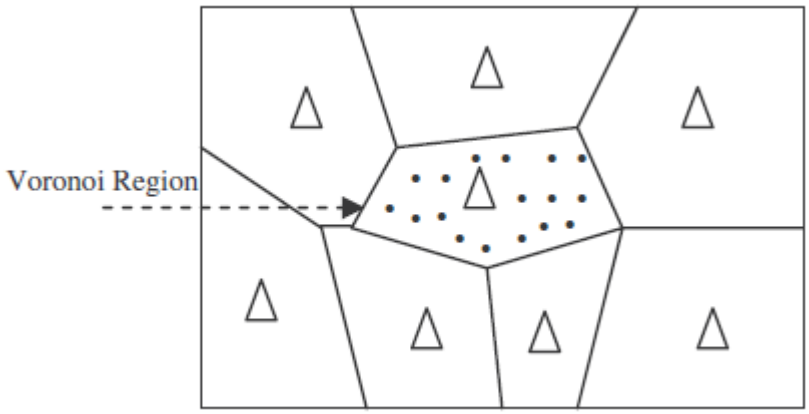
division de la zone selon le diagramme de vironi [52], ce qui permet de s'assurer que chaque capteur ne communiquera qu'avec l'actionneur le plus approprié.

A travers ce protocole, l'utilité de la réallocation apparaît plus clairement car il ne suffit pas d'attribuer le capteur à son actionneur mais plutôt de mettre les actionneurs au service des capteurs. Dans ce qui suit, trois autres protocoles seront présentés. A la fin du chapitre, une synthèse comparative nous permettra de choisir l'approche la plus appropriée pour notre framework qui sera décrit dans le prochain chapitre.

IV.2 Protocole de Coordination pour les réseaux de capteurs avec actionneurs

A partir des chapitres précédents, on peut déduire que la coordination et la synchronisation entre les différents nœuds constitue la clé d'un réseau WSAN efficace. La coordination capteur-capteur fait appel au clustering. Le clustering dans les réseaux de capteur dépend des événements détectés. En d'autres termes, le clustering doit prendre en compte la nature des données senties.

Ce travail [51] ne porte pas sur la conservation de l'énergie mais plutôt sur le besoin des capteurs à partager les mêmes informations ou à détecter le même événement. De nouveaux types de clusters qui prennent en considération la nature des données captées sont donc nécessaires.



FigureIV.1 : WSAN organisé selon le diagramme de Veroni.

Une fois les nœuds déployés, chaque nœud diffuse aux autres capteurs sa position et les fonctions qu'il peut exécuter. En utilisant cette information, chaque actionneur crée son propre diagramme de Voronoï, comme illustré dans la figure IV.1. L'actionneur résidant à l'intérieur de chaque zone (Voronoï) maintient un tableau contenant la liste des capteurs appartenant à cette même zone. Une fois la liste remplie, l'actionneur envoie un message (actionneur_région) aux capteurs de sa région. Ce message indique aussi ID de l'actionneur.

Chaque capteur maintient une table de routage vers les actionneurs. Lors de la réception du message, il crée une entrée pour le voisin dont il a reçu le message. Comme les capteurs peuvent devenir défaillants (par manque d'énergie), un seul chemin peut être insuffisant. Ainsi, chaque capteur crée autant d'entrées que de messages (actionneur-région) qu'il classe en fonction de leur temps d'arrivée, de manière à trouver le chemin le plus rapide.

Comme mentionné précédemment, les actionneurs sont peu déployés par rapport aux capteurs.
Soit s un capteur, i et j deux actionneurs déployés dans le même réseau. Soit **di** (respectivement **dj)** la distance qui sépare le capteur de l'actionneur i (respectivement l'actionneur j). Si di⊳ dj, le capteur est plus proche de i que de j, et donc se situe dans le demi-plan qui contient l'actionneur i. En appliquant cette règle à plusieurs paires de capteur-actionneur, nous pouvons définir l'actionneur de chaque région le plus proche des capteurs.

Comme les capteurs sont statiques, chaque actionneur peut déterminer, initialement, ses capteurs voisins. Ces capteurs sont le résultats du diagramme de Veronoi [57].

-Soit **S={p1,p2,….pn}** un ensemble fini de n dimension dans $\mathbf{R}^{\mathbf{n}}$,

-Soit **d(p,q)** la distance qui sépare 2 points **p** et **q** appartenant à **S**.

-Soit **B** la ligne perpendiculaire au segment **[pq]** au milieu. Cette ligne sépare le plan en deux.

B(p, q)={x| d(p,x)=d(q,x)}.

-D(p,q)= {x/ d(p,x) d(q,x)}

-V(p,S) = $\bigcap_{q\in S, q\neq p} D(p,q)$ est la région de voronoi de p respectivement à S.

-Le diagramme de voronoi de S est défini par : **V(P)=** $\bigcup_{i=1}^{N} V(p_i)$

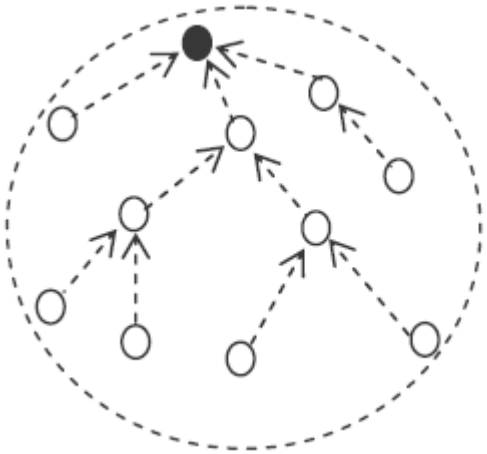


Figure IV.2 : arbre formé par les membres du cluster.

A la détection d'un événement, le protocole de clustering à base d'événement, déjà présenté dans [58, 59] est utilisé. Les protocoles proposés dans ce travail utilisent un arbre d'organisation au sein de chaque cluster. Le but de cette organisation est de prévenir les défaillances des nœuds par la création de nombreux chemins entre les membres du cluster et le cluster head, tout en transmettant l'information périodiquement.

Toutefois, comme l'objectif de ce travail est de détecter l'événement et de prévenir l'actionneur, la redondance des chemins n'est pas nécessaire. A cet effet, une simple organisation où le cluster head constitue la racine d'un arbre (comme illustré en figureIV.2) suffira. Chaque membre du cluster envoie sa lecture, l'ID du plus proche actionneur et la distance qui le sépare de cet actionneur.

En agrégeant les distances capteurs-actionneurs pour chaque paquet reçu par le nœud parent, le cluster head peut déterminer l'actionneur le plus proche de la zone d'événement ainsi que le plus proche capteur (dans cluster) par rapport à cet actionneur. Ce dernier prendra en charge l'exécution des réactions nécessaires.

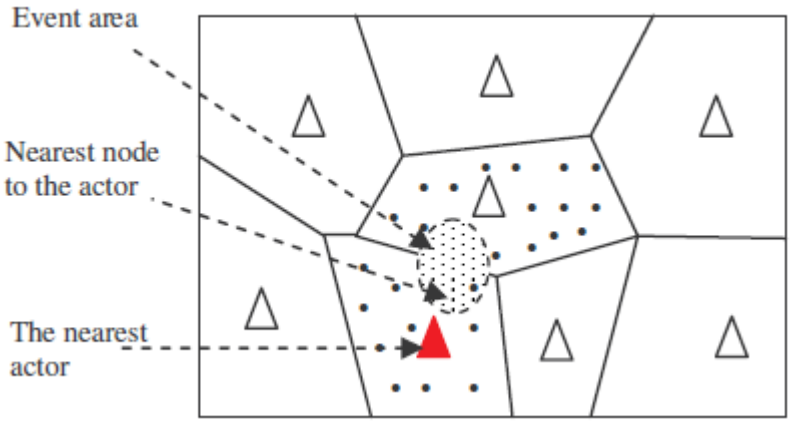


Figure IV.3 : le cluster head détermine le plus proche actionneur.

A ce stade, le cluster head envoie une requête actor-invocation au membre du cluster le plus proche de l'actionneur spécifique. Outre l'ID de l'actionneur et du membre, le message Acteur-invocation contient la position géographique de l'événement détecté.

À la réception du message Acteur-invocation, le capteur le plus proche, consulte sa table de routage et cherche le chemin le plus rapide pour l'actionneur. Une fois l'entrée trouvée, le capteur envoie le message Acteur-invocation. Bien que le WSAN est divisé en régions d'influence, un actionneur peut accéder à d'autres régions d'influence si l'événement se trouve dans plus d'une région.

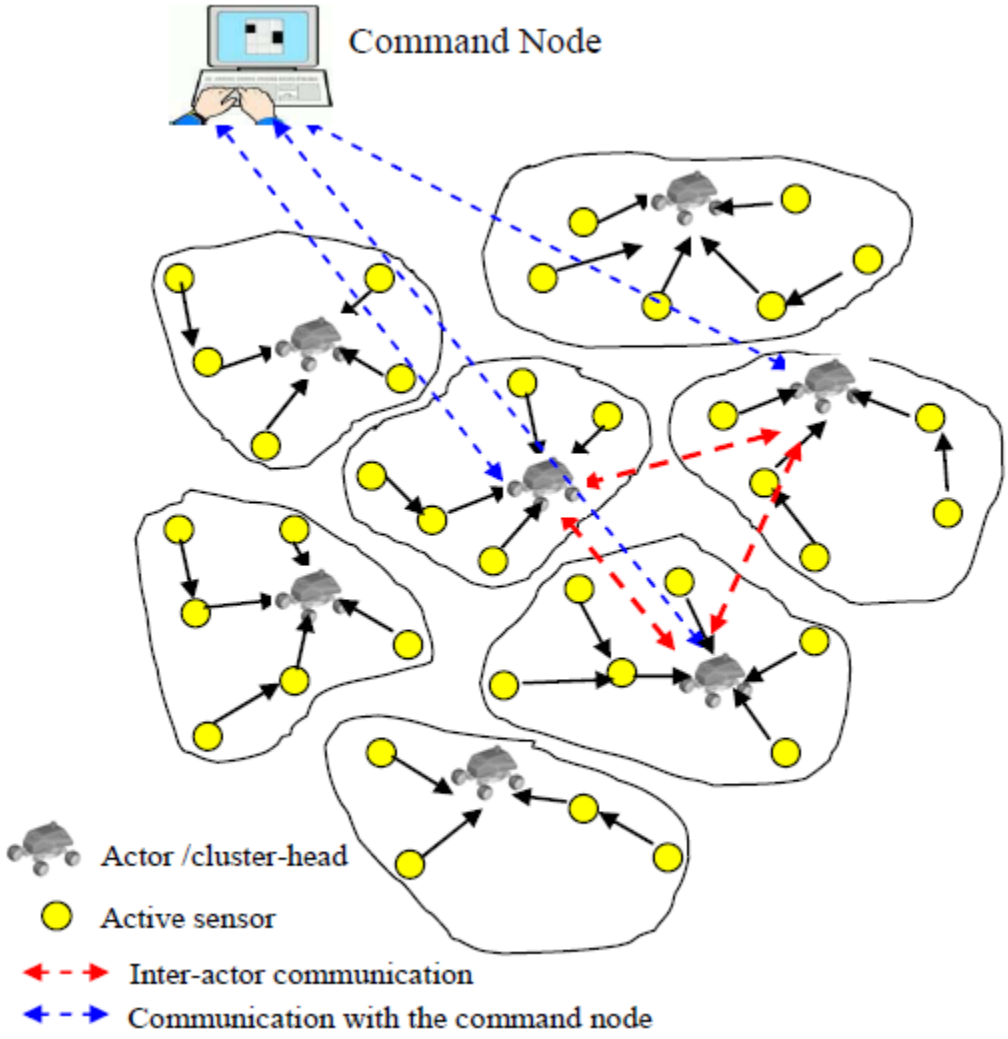
IV.3 Le framework de communication temps réel pour les WSAN ***(Reliable reporting of delay sensitive events in wireless sensor actuator networks)***

Le but de ce protocole, déjà décrit dans le chapitre précédant, est d'assurer une coordination fiable. De ce fait, le facteur le plus important est la latence que met un paquet de données senti pour atteindre un actionneur. Lorsqu'un actionneur reçoit le rapport d'événement, il informe les autres actionneurs pour renforcer leur potentiel d'action. Cette coordination peut être assurée à l'aide d'une communication à un seul saut. Cependant, la communication entre les capteurs et les actionneurs est à multi saut puisque le nombre d'actionneurs est beaucoup moins important que le nombre de capteurs et la puissance de transmission des capteurs n'est pas assez importante. La solution peut nécessiter le repositionnement des actionneurs tout prés des capteurs. Toutefois, il est préférable de placer les actionneurs selon l'occurrence des événements du fait que l'intervention des actionneurs n'est déclenchée qu'après l'occurrence d'un événement.

Dans ce protocole, l'allocation des actionneurs est faite à l'état initial avec des fréquences pre-estimées. Pour équilibrer les charges entre les actionneurs, les fréquences seront tout d'abord additionnées. Ensuite, le terrain sera également divisé en deux sous-terrains, en fonction de la fréquence de distribution. Chacun des deux terrains doit avoir la moitié des actionneurs si ceux ci ont les mêmes fréquences. Le processus se répète récursivement pour chacune des sous-zones, jusqu'à ce que chaque sous-champ contienne un seul actionneur. Les détails de l'algorithme ont été décrits dans le chapitre précédant.

IV.4 COVERAGE AND LATENCY AWARE ACTOR PLACEMENT (cola) [60]

Le réseau se compose d'un ensemble de capteurs fortement déployé sur la zone et d'un ensemble d'actionneurs. Ces derniers sont peu déployés et mobiles avec deux puissances de transmission : Une large portée de transmission pour la commination inter actionneurs et une portée limitée pour la communication capteur-actionneur. La puissance de transmission est la distance maximale qu'un nœud peut couvrir. Elle est limitée et présumée égale pour tous les actionneurs.



FigureIV.4 : réseau de capteurs avec actionneurs multi cluster.

Comme le montre la figure IV.4, chaque actionneur est responsable de la collecte des données de son propre groupe où il devra réagir aux différents événements sur la base des données recueillies. Il est supposé aussi que les actionneurs peuvent se déplacer à la demande, afin d'agir sur de plus grandes zones ou d'améliorer l'exploitation du réseau.

Le problème peut être défini comme suit: «Étant donné un domaine d'intérêt, avec un ensemble de capteurs et d'actionneurs initialement déployés au hasard dans ce domaine, quel est le meilleur placement pour les actionneurs qui fournira une couverture optimale et qui réduira au maximum le délai de la collecte de données tout en considérant la consommation d'énergies des capteurs ». Les économies d'énergie peuvent habituellement se régler en minimisant le nombre de sauts. De ce fait, le problème peut être posé ainsi : "une couverture Max avec une latence Min", où la latence de la collecte de données doit être réduite au maximum. En fonction de la portée de transmission, on peut se situer dans deux situations possibles. Lorsque les actionneurs sont en mesure de

communiquer sur un lien de longue distance, le placement devient moins contraignant. Dans ce cas, la préoccupation concernera le retard de livraison des paquets d'un capteur à tout actionneur. Toutefois, dans le cas contraire, les actionneurs n'ont pas suffisamment de couverture pour garantir la connectivité avec tous les autres actionneurs. Dans cette section, nous ne considérons que le premier cas. Le second cas est bien détaillé dans [60].

Compte tenu des limites liées a la qualité des liens sans fil, trouver les positions optimales pour les actionneurs est un problème très complexe qui a fait ses preuves pour être NP-complexe puisqu'il est de classe P-centre [61]. Ainsi, même si nous connaissons la position optimale d'un actionneur donné, il n'existe aucune garantie que l'actionneur sera accessible à partir d'un capteur voisin. L'idée de base de cola est de considérer à la fois la topologie du réseau de capteurs et les limites de la région de déploiement. Pour répondre à l'exigence de couverture, cola minimise les zones de chevauchement. COLA est divisé en deux phases, la phase bootstrap et la phase de réallocation des actionneurs.

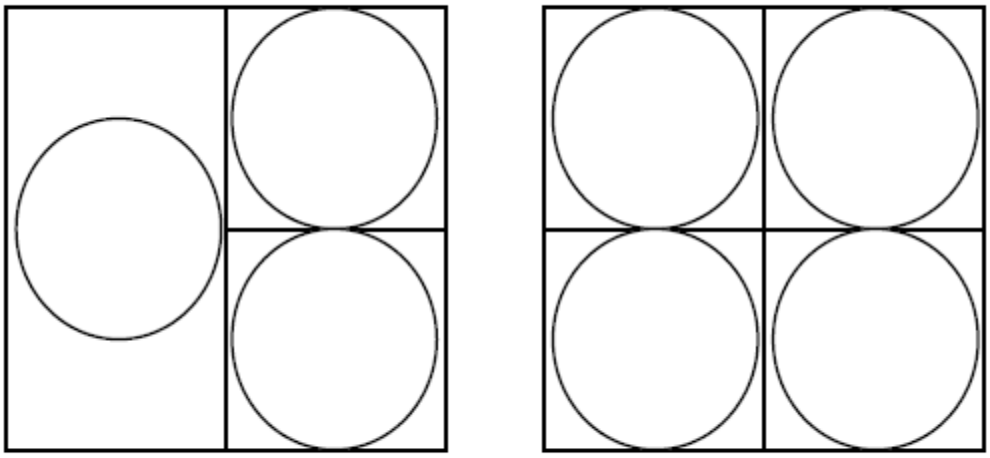
IV.4.1 Réorganisation du réseau (Bootstrap)

Cette phase comprend le placement initial des actionneurs et le regroupement des capteurs en cluster.

IV.4.1.1 Placement des actionneurs:

Le problème de couverture peut être défini comme la maximisation de la superficie totale couverte par des actionneurs. Cet objectif peut être atteint d'une manière statique ou d'une manière dynamique [62]. La Couverture statique déploie les actionneurs de telle sorte que chaque point dans la zone est sous la responsabilité d'un actionneur (c'est-à-dire couvert) à chaque instant. Il est clair, pour compléter la couverture statique que le nombre d'actionneurs devrait être suffisant pour couvrir la zone d'événement. La Couverture dynamique est traitée par des algorithmes qui explorent le mouvement des actionneurs pour couvrir la zone complète [62]. Dans ce protocole, la couverture dynamique est utilisée suite à une exigence de réaction. L'actionneur ne se déplacera donc que s'il ne couvre pas entièrement la zone dont il est responsable.

Notons qu'il s'agit d'un cas différent de la portée de transmission qui dépend de la fréquence radio de transmission.



FigureIV.5 : placement initial des actionneurs

En Cola, le placement initial des actionneurs se fait par division de la région en cellules de tailles égales dont le nombre correspond au nombre d'actionneurs, puis par repositionnement des actionneurs au centre de ces cellules. La Figure 4.5 illustre cette idée, où les cercles représentent la gamme d'action des actionneurs. Ce placement fournira la couverture maximale possible, en évitant les chevauchements entre les diverses gammes d'action. Afin de calculer ces endroits, un algorithme distribué est utilisé. Soit n le nombre d'actionneurs, Ai désigne l'actionneur de la zone i et Posi indique sa position.

Algorithme 1:

---

1. /*Chaque Actionneur A_i Diffuse son ID et sa position*/

Pour chaque actionneur A_i

Début

    BroadCast(i, Pos_i);

Fin Pour

2. /*chaque actionneur utilise un algorithme récursif qui détermine les cellules en fonction du nombre et des positions de l'ensemble des actionneurs ainsi que de la taille de la zone d'évènement :*/

```
Deviser (Zone, n)
Début
    Si (n==1)
```

```
        Début
        |       /*Insérer les coordonnées de la cellule dans l'arbre binaire*/
        |       InsertTree (CellTree, Centre Zone(Zone));
        Fin Si
        SINON
        Début
        |   /*Diviser récursivement la zone en cellules*/
        |   Calculer(Zone, Cell1, Cell2);
        |   Diviser(Cell1, [n/2]);
            Diviser(Cell2, [n/2]);
        Fin Sinon
Fin
```

3. /*Enfin, chaque actionneur calcule la cellule la plus proche pour se déplacer, en comparant ses positions initiales avec les positions des centres stockés dans l'arbre binaire. Sachant que ce mouvement consomme, du temps et de l'énergie, cette approche réduit la distance globale parcourue par les nœuds actionneurs ainsi que le temps global.*/

```
Pour chaque  actionneur Ai
Début
|       /* Rechercher CellTree pour  la cellule non-visitée la plus proche de Ai
|       min=FindClosest(i, posi, CellTree);
|
|      /*Stocker la position de la cellule la plus proche dans une liste avec l'ID de Ai
*/
|
|       NewLoc(i) = min;
|
        /*Marquer la cellule comme visitée*/
        MrkVisited(min);
Fin
```

4. /*En retrouvant la position la plus proche dans la liste, l'actionneur se déplace vers cette position*/

```
Move(i, NewLoc(i));
```

Cet algorithme ne contient aucun conflit, c'est à dire que chaque actionneur dispose lui seul d'une zone donnée, ce qui est prouvé dans [60]

IV.4.1.2 formation des Clusters

Après avoir positionné les actionneurs, l'étape suivante consiste à regrouper les capteurs en groupes distincts, chacun étant dirigé par un actionneur. Un actionneur sera responsable de la collecte des données des capteurs dans son groupe. Il existe de nombreuses techniques publiées, par exemple [63], qui peuvent être appliquées pour l'inclusion des capteurs en groupes. Dans sa forme la plus simple, chaque actionneur diffuse un message avec son ID et son emplacement. Lorsqu'un capteur reçoit ce message, il estime son actionneur. Le capteur attend encore d'entendre d'autres actionneurs, puis répond à son cluster head.

IV.4.2 Réallocation des actionneurs pour minimiser la latence

Le fait que les actionneurs sont positionnés pour une couverture maximale, ne signifie pas que c'est le meilleur emplacement pour réduire au maximum la latence de la collecte de données. L'emplacement devrait être choisi dans le but de réduire au maximum le délai q'un paquet traverse de la source à l'actionneur correspondant (cluster head). Ceci revient à minimiser le nombre de sauts qu'un paquet traverse. Par conséquent, COLA replace les actionneurs du centre des cellules à de nouveaux endroits, en réduisant significativement la latence. Ce déplacement permet également de rendre les actionneurs de plus en plus proches des sources de données et, par conséquent, de réduire le nombre de transmissions du capteur qui étend leur durée de vie. Alors que cette délocalisation peut avoir une incidence négative sur les performances en termes de couverture, il est important de noter que le nouvel emplacement ne serait pas très éloigné de l'emplacement initial puisque le déplacement ne dépassera pas la gamme de couverture de chacun des actionneurs.

Étant donné que l'un des facteurs qui influent sur la qualité des réactions est le temps entre l'instant où l'événement sera senti et celui où les paquets d'événements seront reçus par les actionneurs. Il importe particulièrement que ce temps soit réduit au maximum afin d'accélérer le processus de prise de décision pour les actionneurs. Cet objectif peut être atteint en déterminant un emplacement adéquat, pour chacun des actionneurs, qui assure que même les capteurs les plus éloignés peuvent transmettre leurs données à un actionneur dans un délai acceptable. Trouver la position optimale qui répond à ces

objectifs est un problème classé 1-centre, qui est connu pour être NP-difficile. Par conséquent, COLA poursuit une formulation en sommet 1-centre qui est une solution polynomiale [64]. Le sommet 1-centre a de nombreuses applications telles que la planification urbaine. Lorsqu'on place les centres de services comme les hôpitaux dans une ville, il est souhaitable que les services d'ambulance puissent être expédiés rapidement aux patients ou aux scènes d'accidents, quel que soit le lieu où ces derniers se trouvent. Dans notre contexte, cette formulation signifie que la nouvelle position pour un actionneur doit être recherchée entre les lieux qui servent de sommets. Résoudre le sommet 1-centre se traduira par un nouvel emplacement qui garantit que la latence maximale, en termes de saut, sera réduite.

Algorithme 2:

---

1. /* l'actionneur calcule la matrice de distance minimale pour le cluster */

Pour chaque i, j appartenant au Cluster
Début
    M=d[i,j];
Fin Pour

2. Pour Chaque ligne k de M

Début
    /* Trouver le plus long chemin pour ce nœud par rapport aux autres nœuds*/
3. Trouver Max(ligne(k));
4. maxlist=Max(ligne(k));

Fin Pour

5. /*sélectionner la plus courte entrée parmi l'ensemble des chemins les plus long*/

minCost= Min(maxlist)

6. vertexcenter=position(premier noeud dans le chemin (mincost)) ;
7. Ré-allocation des actionneurs au vertexcenter.

Le pseudo-code pour calculer le sommet 1-centre est indiqué ci-dessous. L'idée principale est d'abord de créer une matrice qui indique la distance minimale pour chaque capteur dans son cluster (ligne 1). d [i, j] désigne essentiellement le nombre de sauts minimum possibles du capteur i à j. Puis, pour chaque noeud, nous prenons le chemin le

plus long de ce nœud à l'un des autres nœuds (ligne 2-3). Ces valeurs sont stockées dans une liste (notée MaxList) et la plus petite valeur est calculée en ligne 4-5. Le sommet 1-centre sera la position du noeud qui a la plus petite valeur dans sa ligne (ligne 6). L'actionneur est donc transféré à cet endroit comme le montre la ligne 7.

Dans cet algorithme, une fois la matrice M des distances minimales calculée, la complexité de la partie restante est de l'ordre du nombre de capteurs. M peut être calculée en temps polynomial en utilisant les algorithmes de Floyd-Warshall pour le calcul des plus courts-chemins [63]. Il convient de noter que chaque actionneur se sert de ces algorithmes et identifie le meilleur endroit pour servir son groupe indépendamment des autres actionneurs. Depuis, le sommet 1-centre est en fait la position de l'un des capteurs du cluster. L'actionneur reste dans son même cluster et les chevauchements des actionneurs restent minimaux. Cette propriété a été validée par simulation dans la section 6 du [60].

IV.5 Coverage-based Clustering of Wireless Sensor and Actor Networks (IDS)

Ce protocole est dédié aux WSANs à actionneurs mobiles. Le problème traité par ce protocole peut être définit comme suit : «Étant donné un ensemble de capteurs initialement placés au hasard dans un domaine d'intérêt et un certain nombre d'actionneurs, l'intérêt est de mettre les actionneurs en groupe de telle sorte que chaque cluster head est un actionneur et que le total des actionneurs couvre d'une manière optimale la zone entière. »

Dans ce contexte, le protocole [65] se réfère au nombre de capteurs qui sont sous le contrôle des actionneurs. Notons que ceci est une nouvelle interprétation de la couverture, par opposition aux travaux antérieurs. L'idée est de considérer la répartition des capteurs dans la région contrôlée car certaines parties ne disposent pas de capteurs déployés. Même si certains événements se produisent dans ces domaines, les actionneurs ne seront pas en mesure de les connaître car aucun capteur ne fera de rapport sur eux. Déployer un actionneur dans un tel lieu équivaut donc à un gaspillage de ressources.

IDSC contient deux étapes importantes: la phase de placement des actionneurs et celle de la formation des clusters.

IV.5.1 Placement des actionneurs

Une solution idéale à ce problème consisterait à permettre de réduire au minimum le chevauchement entre les gammes d'action des actionneurs et de couvrir tous les capteurs déployés dans la région contrôlée. Toutefois, cela nécessite un noyau central qui devrait connaître le nombre et l'emplacement de tous les capteurs déployés. Dans la plupart des

applications où les capteurs sont positionnés au hasard, cela n'est pas toujours possible. En outre, si le nombre d'actionneurs est faible, le problème se complique davantage. En fait, le problème de déterminer les meilleurs emplacements pour chacun des actionneurs s'est avéré être NP- complexe étant donné le nombre infini d'emplacements possibles [66]. Par conséquent, une solution distribuée dans laquelle les capteurs peuvent déterminer un rapprochement de l'emplacement optimal du cluster head a été présentée.

Le réseau de capteurs sera modélisé par un graphe G = (V, E) où V est l'ensemble des sommets et E est l'ensemble des arcs. Dans un tel graphe, deux noeuds sont reliés par une arête si et seulement si ils peuvent communiquer les uns avec les autres. On distingue les définitions suivantes pour G:

• ***Independent Set (IS):*** c'est l'ensemble de *V pour lesquels il n'y a pas deux vecteurs qui sont adjacents dans V*.

• ***Dominating Set (DS):*** *soit* S une DS. S est un sous ensemble de V tel que chaque sommet dans V - DS est adjacent à au moins un sommet dans S.

• ***k-Dominating Set (k-DS):*** *soit* S un *k*-DS. S est un ensemble de V tel que chaque sommet dans V-S peut atteindre au moins un sommet de S au bout de *k*-sauts.

• ***Independent Dominating Set (IDS):*** *est une* DS tel qu'il n'existe pas deux dominateurs qui y sont adjacent. Les dominants doivent au minimum être séparés d'un saut. Trouver l'IDS d'un graphe est un problème NP-complexe [67].

• ***k-Independent Dominating Set (k-IDS):*** *est une* *k*-DS tel qu'il n'existe pas deux dominants qui y sont adjacents. Les dominateurs doivent au minimum être séparés d'un saut (dans cet ensemble).

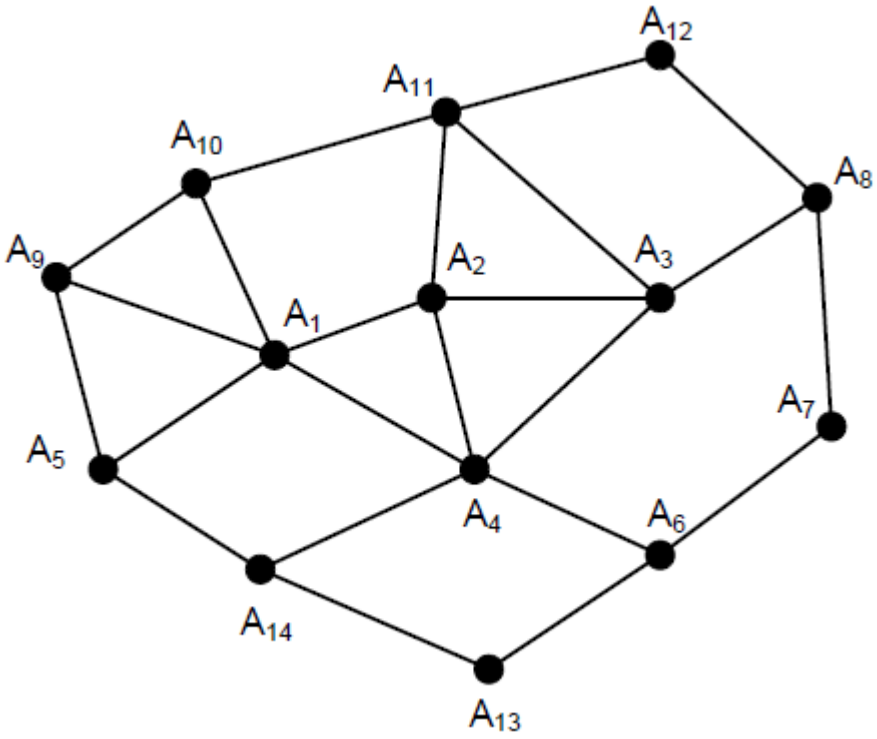


Figure. IV.6 : les ensembles DS, IDS d'un réseau WSAN.

Le réseau de 14 noeuds indiqué dans la figure IV.6 clarifie plus les notions cites ci-dessus :

1) A2, A4, A8, A9, A13 forment une DS.

2) A4, A8, A9, A11 et A13 forment une IDS.

3) A2 et A4 forment une 2-DS.

4) A1 et A7 forment une 2-IDS.

5) A4 forme a 3-DS.

Une bonne couverture peut être atteinte par la détermination d'un IS du réseau de capteurs. Les actionneurs seront placés à côté de ces noeuds. En l'absence d'un lien direct entre deux nœuds, le nombre de chevauchement des gammes d'actions sera réduit. Toutefois, la taille de l'IS devrait être suffisante pour couvrir l'ensemble du réseau de capteurs. C'est exactement la même chose que de déterminer une DS du réseau de capteurs de sorte que chaque capteur sera sous le contrôle d'un seul dominateur (actionneur). Par conséquent, la solution idéale est de déterminer l'IDS du réseau de capteurs. Cependant, afin de réduire au minimum le nombre d'actionneurs à déployer et donc le nombre de clusters, il est nécessaire de réduire la taille de IDS, ce qui pose le

problème de la détermination de k-IDS où chaque actionneur est au plus loin de k saut des capteurs qu'il domine.

Bien que la détermination d'un IDS constitue un problème NP-complexe [68], quelques algorithmes approximatifs ont été proposés [69]. En effet, IDS est un cas spécial du *k*-IDS où k=1. Par conséquent, *k*-IDS est NP-complexe [69][70]. Dans ce qui suit, sera décrit l'algorithme distribué proposé dans [60] pour la détermination d'un *k*-IDS d'un réseau de capteurs avec actionneurs.

IV.5.1.1 les Détails de k-IDS

Afin de déterminer k-IDS, trois types différents de messages sont définis:

1) ALIVE: message transmis par chacun des capteurs à ses k^{eme} voisins pour vérifier leur existence.

2) DOMINATOR: message transmis par chaque capteur dominateur à ses k^{eme} voisins

3) *BORDER: message transmis par le capteur voisin du dominateur de k sauts*.

Pour chaque capteur, ces messages doivent s'échanger entre k saut pour minimiser le coût de transmission. L'idée est de déterminer les nœuds dominants en se basant sur la probabilité calculée. Ces noeuds diffusent un message Dominator à leurs k^{eme} voisins. La probabilité de devenir un dominateur dépend de nombreux facteurs comme le nombre de messages reçus ALIVE, la distance par rapport au K^{eme} saut et quelques paramètres fixes comme il sera expliqué en détail ci-dessous. Tout noeud recevant un message Dominateur sera dominé par l'expéditeur du message. Si un nœud ne reçoit pas un tel message et il n'est pas dominateur, il décide également de l'être. Afin de garantir l'indépendance de propriété, les nœuds à la frontière (c'est-à-dire, au K^{eme} saut d'un dominateur) diffusent des messages BORDER à leur k^{eme} voisins. Avec l'augmentation de la valeur de k, la probabilité de devenir un dominateur se réduit, ce qui donne la chance d'éliminer les chevauchements possibles entre les gammes d'actions des dominateurs. L'algorithme est donné ci-dessous:

Algorithme 3:

/* Pour chaque capteur i */

```
k-IDS(i) ;
1. broadcast(msg(i,  « alive », voisins(i),k))
2. tant que true faire
3. wait(I,T)
4. tabulate le message recu (,"alive",i)
5. tabulate le message recu (,"dominator",i)
6. tabulate le message recu (,"border",i)
7. forward(msg(j, »alive »,voisins(i),k-1)

/* lors de la reception du message « dominator »  */
8. Si !i=dominateur et  received msg (j, »dominator »,i,k) alors
9.          dominated(i)←TRUE
10.                 forward(msg(j, »dominator »,i,k-1)
11.                    Si  Dist(j)< plus_proche_donimateur(i) alors
12.                     plus_proche_donimateur(i) ←j
13.                    Finsi
14.                 Si k=1 alors //le message dominator expire
15.                 broadcast(msg(i,  « border »,voisins(i),k))
16.                 Finsi
17.       Finsi

/* lors de la réception du message « border »  */
18.       Si i= !dominator et received msg (j, »border »,i,k) alors
19.                 si BTTL(i)<msg(j,"border",I,k) alors
20.                       BTTL(i) ←(j,"border",I,k)
21.                 finsi
22.                 forward(msg(i,  « border »,voisins(i),k-1))
23.       finsi

/* utilisation de la fonction S pour devenir dominator  */
24.       S←f(# msg(j, "alive",i), TTL(i))
25.       générer un nombre aléatoire R;
26.       Si (r< S) alors
27.       dominator(i) ←true;
28.       broadcast(msg(i, »dominator »,voisins(i)),k)
29.       finsi
30.       FIN
```

Comme le montre la ligne 1, le noeud diffuse un message ALIVE à ses voisins. Comme pour tous les nouveaux messages dans cet algorithme, le message à une durée de vie (TTL) égale à k. Le TTL est le nombre de saut qu'un paquet a déjà parcouru. Le noeud commence alors une boucle de la ligne 2 qui se termine à la ligne 30. En ligne 3, le nœud reste en attente durant une période aléatoire T. Pendant ce temps, il prendra note de tous les messages qu'il entend. Cette période évite que les nœuds voisins agissent en même temps. La période exacte pour chaque nœud change à chaque itération.

Lignes 4-6 totalise les messages reçus par le capteur i. Il s'agit des messages **Dominator, BORDERS et ALIVE**. Le nombre de messages ALIVE reçus représente le nombre de noeuds se situant à k sauts. Dans la ligne 7, le capteur i transmet chaque message reçu ALIVE à chacun de ses voisins en décrémentant le TTL des messages à envoyer. Lignes 8-17 détermine si le capteur sera dominé par un autre noeud, basé sur les messages reçus. Si le capteur i reçoit un message Dominator du noeud j et i n'est pas déjà un dominateur, il sera dominé par j tant qu'il est le plus proche dominateur qu'il a entendu. Le capteur i envoie également le message Dominator à ses voisins si le TTL du paquet reçu n'a pas expiré. Sinon, le capteur i diffusera le message BORDER à ses voisins avec le TTL fixé à K.

Les lignes 18-23 seront exécutées si le nœud i entend un message BORDER et il n'est pas dominateur. Si cette condition est vérifiée, le TTL du message BORDER est enregistrés et il est comparé aux autres TTL des messages BORDER déjà reçu. Le maximum de ces valeurs TTL est enregistré sous forme de BTTL la valeur qui sera utilisée dans le calcul de la formule d'aptitude pour devenir un dominateur. Le capteur i transmet ensuite le message BORDER à ses voisins si le TTL du message n'a pas encore expiré.

La ligne 24 détermine l'aptitude d'un capteur à devenir un cluster head. Cette aptitude est une fonction S qui compte le nombre de ALIVE et BORDER messages reçus. La fonction suivante est utilisée dans l'algorithme:

$$S_i = \left(\frac{B + M_i * AC_i}{{}_2 BTTL} \right)$$

Où:

• ***B est une constante exprimant la chance d'être un*** **dominateur.**

• M_i **est une constante attribué aux voisins distant de k saut par rapport a i.**

• AC_i **est le nombre de message *ALIVE reçu par le capteur i*.**

• ***BTTL est le maximal*** **TTL des messages *BORDER* déjà reçu.**

Cette formule prévoit plusieurs comportements. Elle permet d'abord à n'importe quel noeud de devenir un dominateur quel que soit le nombre de ses voisins. Cela permet aux noeuds isolés d'avoir un dominateur et donc un cluster head. Elle favorise ensuite les nœuds qui ont plus de voisins pour être dominateurs par rapport à ceux qui ont moins de voisins. En outre, le BTTL (dans la formule) réduit les chances de devenir un dominateur s'il est proche des frontières d'un autre dominateur, ce qui réduit les zones de chevauchement des gammes d'action des actionneurs qui seront employés comme cluster head à la position de dominateurs. Par exemple, dans un réseau où k = 3, un nœud de trois sauts du dominateur diffusera un message BORDER avec ttl=3 qui sera rediffusé pour une nouvelle période de trois saut. Un noeud qui reçoit le message avec TTL = 3 saura qu'il est directement sur la frontière d'un groupe déjà formé. Par conséquent, s'il est appelé à devenir dominator, il créera un chevauchement important entre les gammes d'actions. Ce nœud ne sera probablement pas un dominator à moins qu'il n'a pas reçu un message Dominator depuis un bon moment. Toutefois, les nœuds éloignés de la frontière (par exemple, ceux qui reçoivent un message BORDER avec un TTL petit) ont de petites chances de devenir un dominator. Lignes 24-30 généré une probabilité aléatoire qui est ensuite comparée à une valeur bien définie afin de déterminer si le nœud deviendra un dominator. Si tel est le cas, le noeud diffuse un message Dominator à tous ses voisins.

IV.5.1.2 Complexité de k-IDS

Ce protocole introduit trois petits messages généraux. Chaque nœud envoie des messages ALIVE pour découvrir ses voisins et des messages Dominator s'il décide d'être un dominateur. Puis, ce message traversera k saut et chacun de ces nœuds relais retransmettra le message. Enfin, les noeuds au KTH saut diffusent des messages BORDER.

Dans le pire des cas, un noeud peut entendre un message Dominator de tous les dominateurs. Soit s = | k-IDS |, puis le nombre de messages Dominator qu'un nœud transmet dans le pire des cas serait égal à s ceci si k est considéré comme suffisamment important pour atteindre tous les capteurs dans la région. En conclusion, au minimum chaque nœud transmettra (s +2) messages conduisant à une complexité de O (s).

IV.5.2 Le CLUSTERING

Dans cette section sera décrite la manière dont les clusters se forment et se maintiennent après le placement des actionneurs selon IDSC.

IV.5.2.1 la Formation des Clusters

Une fois les dominateurs déterminés, la prochaine étape consiste à former les groupes. Dans la phase de détermination des dominateurs, chaque capteur peut entendre le message Dominator de multiples dominateurs et peut choisir son dominator basé sur la distance et son identité comme indiqué dans la section précédente. Ensuite, les actionneurs seront placés à côté des dominateurs qui peuvent agir comme des mandataires du cluster-head jusqu'à ce que les actionneurs soient déployés. Par conséquent, un capteur envoie un message JOIN pour rejoindre son dominateur ainsi que son groupe. Le message JOIN n'est pas diffusé, mais il est unicasté.

Le chemin du message JOIN est obtenu à partir du chemin du message Dominator. Ce message contiendra le chemin de retour au noeud dominator car chaque noeud recevant Dominator ajoute son ID et le rediffuse à ses voisins, tel qu'il est utilisé dans DSR [10]. Un dominateur recevant des messages JOIN construit une liste des capteurs appartenant à son groupe qu'il transmet ensuite à l'actionneur placé juste à côté de ce dominateur.

Quand un tel actionneur est placé / transféré à cet endroit, il envoie un message ACTOR. Chaque dominateur recevant ce message envoie les identificateurs et les itinéraires de tous les noeuds du cluster à l'actionneur. L'actionneur peut alors unicasté un message CLUSTER-HEAD à chaque capteur afin de l'informer de son existence et d'être prêt à recueillir des données. Ce message comprendra également l'ID de l'actionneur qui identifie le groupe. Notons que, avec l'ajout du message JOIN, chaque capteur transmet trois messages afin de compléter le processus de regroupement. Comme dans le cas des messages Dominator, les messages JOIN sont également transmis à travers le dominator. Par conséquent, dans le pire des cas, les capteurs voisins directs d'un dominator devront transmettre les messages JOIN à tous les k^{eme} voisins. Dans ce cas, il transmet les k messages JOIN y compris son message. Par conséquent, pour l'ensemble du processus de regroupement dans le pire des cas, un capteur devra envoyer (1 + s + k) messages qui conduit à une complexité de O(s) ou O (k) en fonction des valeurs de s et de k.

IV.5.2.2 maintenir la mobilité des actionneurs

Du moment que l'ensemble des actionneurs couvrent la zone entière, le cluster déjà formé ne changera pas très souvent (stationnaire la plupart du temps). La seule situation qui impose une réorganisation du groupe se présente l'actionneur (c'est-à-dire, le cluster head) se déplace pour agir dans des zones plus vastes ou pour aider d'autres actionneurs. Dans ce cas, l'actionneur sera absent pour quelques instants et donc les données doivent être collectées par d'autres actionneurs. À cette fin, les capteurs doivent être ré-associés temporairement au plus proche actionneur. Cette communication se fait à l'aide des nœuds situés à la frontière du cluster le plus proche.

Rappelons que les noeuds de la frontière d'un dominator ont envoyé le message BORDER lorsque le ttl du message Dominator a été 1. Par conséquent, ces noeuds s'identifient comme étant la limite du cluster auquel ils appartiennent. Lorsque les routes sont créées, ces nœuds de frontière rajoutent leur ID et leur localisation à tous les capteurs se trouvant sur le chemin vers le cluster head.

De cette façon, chaque capteur peut apprendre son noeud de frontière correspondant et garder le chemin de sa frontière (c'est-à-dire, l'inverse sur la route du message). Quand l'actionneur se déplace, il envoie un message LEAVING à tous les capteurs de son groupe grâce à l'inverse des itinéraires de capteurs. Un capteur qui reçoit ce message entame l'envoi de ses données au noeud de frontière.

Le nœud de frontière diffuse un message GATWAY à ses voisins afin de déterminer la frontière la plus proche dans un autre groupe. Il va donc transmettre les données à ce noeud de frontière de sorte que les données seront acheminées vers un nouvel actionneur dans ce nouveau groupe. Quand l'actionneur revient, il envoie un message BACK aux capteurs et commence la transmission de leurs données à l'actionneur. Il est à noter que la détermination de la frontière pour les capteurs ne coûte qu'un seul message échangé avec les voisins pour identifier la frontière la plus proche dans d'autres groupes.

IV.5.2.3. Communication avec le sink

Afin d'utiliser ce protocole deux questions doivent être examinées:

1) quel est la valeur du paramètre k ?

2) comment obtenir la taille de k-IDS à partir du réseau de capteurs ?

Dans ce protocole, la valeur de k est prédéfinie et, par conséquent, les capteurs peuvent limiter les diffusions des messages Dominator. La valeur de k peut être décidée

sur la base du rapport ar/sr où ar est la gamme d'action et sr la puissance de transmission des capteurs.

Par exemple, si ar = 2SR, il serait judicieux de choisir k au moins 2 afin d'éviter les chevauchements et de maximiser la couverture. En outre, la valeur de k peut être choisie en fonction des exigences de l'application comme le nombre d'actionneurs disponibles, la latence de bout en bout et le temps de réaction.

Par exemple, les applications temps réel nécessitent une latence trop réduite et donc la valeur choisi pour k doit être aussi minime que possible.

Une autre question importante concerne le nombre d'actionneurs nécessaires et leur emplacement. Ces informations peuvent être collecté à partir des capteurs, puis les actionneurs peuvent être programmés pour passer à l'emplacement désiré où ils peuvent directement être placés à ces endroits selon le type d'application. Si la région est accessible, le placement direct serait plus efficace car il permettra d'éliminer le mouvement des actionneurs. Sinon, il n'y a pas d'autre choix que de les laisser se réinstaller à leur nouvel emplacement. Le nœud de command peut être notifié de deux manières: soit les dominateurs envoie directement les informations nécessaires au nœud de command soit par le recours à un sink déployé dans la région de contrôle et qui permet aux dominateurs de relayer l'information.

IV.6 COMPARAISON

Dans les WSAN, il est nécessaire de délivrer les données des événements avec un minimum d'énergie, tout en respectant la contrainte temps réel. La majorité des études de coordination dans les WSAN se concentre sur la construction des clusters. Pour chaque cluster, un actionneur s'occupera de la collecte des données. La communication au sein de chaque groupe se fait en respectant les contraintes énergie, latence, charge des voisins, priorité d'événement, etc. ces protocoles supposent que les nœuds du WSAN (capteurs + actionneurs) sont pré-déployés.

D'autres études dérivées des études du multi robots ont été lancées. Elles répondent à un souci principal de couverture c'est-à-dire que les robots devraient couvrir toute la zone de surveillance. De la, est apparu le problème de réallocation des actionneurs. La question qui se pose est la suivante : comment positionner les actionneurs pour :
-Couvrir toute la zone d'événement.

-Minimiser la latence et l'énergie consommer par la communication au sein de chaque cluster.
-Eviter la surcharge des actionneurs en prenant en compte la distribution des capteurs.

Le premier protocole [56] suppose la décomposition selon le diagramme de vironi, ce qui permettra de distribuer dans des zones de taille plus ou moins égale et n'impliquera aucun déplacement des actionneurs. Il présente l'inconvénient de déployer le même nombre d'actionneurs qu'il s'agisse d'une zone à grande ou à faible densité de capteurs, ce qui pourrait surcharger les actionneurs. Ce protocole présente l'inconvénient supplémentaire de gaspiller inutilement un actionneur dans une zone sans capteur

Le deuxième protocole [29] suppose l'allocation des actionneurs en fonction de la fréquence d'occurrence des événements. Les zones à forte probabilité d'occurrence d'événement disposeront de plus d'actionneurs. Ce protocole présente le désavantage de diviser la zone d'événement en sous zones de taille différente. Dans une vaste sous zone, les paquets parcourent plusieurs sauts d'où une augmentation de la latence.

Le troisième protocole [60] permet de couvrir la totalité de la zone d'événement. Pour cela, il divise celle-ci en cercles non chevauchables, à rayon égal à la porté de transmission des actionneurs. Chaque actionneur se positionnera au milieu d'un cercle puis il se déplacera dans ce cercle pour être le plus proche des capteurs. Son objectif est donc de minimiser la distance entre les capteurs et les actionneurs (minimiser latence et énergie). Son inconvénient est qu'il ne distingue pas entre les zones à grande densité de capteurs et les autres zones. (un actionneur est déployé pour une zone avec 50 capteurs et un autre pour une zone à 2 capteurs).

Le quatrième protocole [65] est un protocole distribué qui vise à séparer les zones d'événement en utilisant la coordination capteur-capteur. Au terme de cette coordination, des capteurs dominateurs seront déterminés. Il s'agit de nœuds qui permettent de séparer les zones en fonction de la distribution globale des capteurs. IDSC comprend deux étapes importantes celle du placement des actionneurs et celle de la formation des clusters. Le cluster utilisé dans ce protocole se base sur les délais de réception des messages de diffusion, ce qui n'est pas vraiment efficace pour retrouver les chemins de transmission dans WSAN. Pourtant il est toujours possible à ce niveau de choisir le chemin adéquat en prenant d'autres critères en considérations. IDSC qui traite les inconvénients des autres protocoles de réallocation constitue l'algorithme de réallocation choisi pour notre framework.

IV.7 CONCLUSION

Les réseaux de capteurs avec actionneurs (WSANs) gagnent en popularité dans un certain nombre d'applications civiles et militaires. Les actionneurs recueillent les données captées et exécutent les tâches appropriées. Dans la majeure partie des applications, ils sont déployés de manière aléatoire. Dans ce chapitre, nous avons présenté les algorithmes qui permettent de repositionner les actionneurs avant de les comparer et de dégager parmi eux celui qui répond le mieux aux exigences environnementales et applicatives des WSANs : économie de l'énergie et réduction de la latence. L'algorithme choisi est IDSC car il répond parfaitement aux exigences des WSAN et traite l'ensemble des inconvénient des autres protocoles déjà présenté au cours de ce chapitre [56][60][65].

Chapitre V : la solution proposée

V.1 Introduction

Les WSAN sont constitués d'un ensemble de capteurs et d'actionneurs distribués dans une zone géographique d'intérêt. Les capteurs contrôlent l'environnement physique alors que les actionneurs exécutent la ou les actions désirées en fonction des données collectées et rapporté par les capteurs à l'apparition d'un événement. Pour atteindre les performances nécessaires, l'architecture d'un WSAN doit implanter une stratégie de communication coopérative efficace permettant aux nœuds de collaborer avec une consommation optimale de ressources et d'exécuter les actions appropriées dans les meilleurs délais. Tout en s'efforçant de répondre aux besoins applicatifs, cette collaboration est possible grâce à l'échange d'informations et de décisions négociées.

Le présent chapitre s'appuie en premier lieu sur une classification des différents mécanismes de coordination qui permettra d'étudier et d'analyser les grandes lignes de notre mécanisme de coopération. Il s'agira ainsi d'exposer notre framework de coordination dédié aux applications temps réel où la rapidité de traitement est une condition nécessaire pour garder la consistance de l'environnement.

D'un autre coté, les pannes fréquentes pouvant provoquer des pertes de données et des zones non couvertes, un contrôle de la consommation d'énergie au moment du rapport est indispensable. Afin de bien prendre en charge ces deux paramètres, un schéma de clustering est proposé. Ce clustering prend en charge plusieurs critères pour la sélection du prochain saut: le nombre de pannes, le nombre de paquets perdus ou de congestion, etc.

Le positionnement des capteurs dépend de l'application et des portions qu'ils doivent contrôler. Les actionneurs sont des nœuds qui doivent agir rapidement et prendre les décisions les plus appropriées. Une solution très intéressante consiste à placer ces actionneurs au bon endroit pour qu'ils puissent recevoir l'information le plus tôt possible. Cependant, le choix des bonnes positions nécessite le respect d'un ensemble de contraintes ou d'objectifs. Ce sera le propos de la seconde partie.

Apres la phase de description générale des différents axe de notre mécanisme de coordination, les procédures collaboratives seront détaillées, en troisième partie. Ces procédures permettent la construction des arborescences de communication. Les clustering dépendent de l'environnement et de son évolution. En fonction de la fiabilité recensée par le cluster head (l'actionneur localisé durant la phase initiale), chaque capteur bascule d'un état à un autre. Le processus ainsi développé dans le présent chapitre comporte quatre phases : une phase de compromis énergie latence (start up), une phase d'accélération des transmissions (fiabilité) et deux phases d'économie l'énergie (agrégation, énergie).

V.2 Principe général de fonctionnement du protocole

Si les travaux sur la classification des mécanismes de collaboration pour les WSAN ne sont pas légion [71], ils n'en sont pas moins indispensables à toute compréhension du principe général de fonction de notre protocole. Nous en citerons quatre :

Farinelli's[72] présente une classification pour les systèmes multi robots basée sur : la coopération, l'apprentissage, la coordination et l'organisation. Cette classification ne considère que les systèmes coopératifs qui incluent les systèmes robotiques et les systèmes de coordination permettant la prise de décision, sans toutefois intégrer le partage des données et de contextes.

Salkhamen et al [73] présentent une classification pour les systèmes collaboratifs basée sur les différents contextes constitufs. Cependant, il ne s'agit pas d'occurrence d'une classification spécifique aux WSANs mais d'une large classification qui s'applique aussi bien aux petits systèmes qu'aux réseaux extrêmement distribués. Composée de trois axes (but, approche et sens), cette structure ne fournit pourtant pas d'éléments nécessaire à une bonne classification des mécanismes de coordination dans les WSANs.

Sammer et al [74] développent pour leurs part un service de middleware comportant : la synchronisation, l'agrégation et localisation pour faciliter la coordination à travers un réseau de capteur auto-organisé. Une nouvelle approche proposée dans [71], se base sur la classification de salkhamem[73] et sameer[74]. Elle se divise en quatre sections :

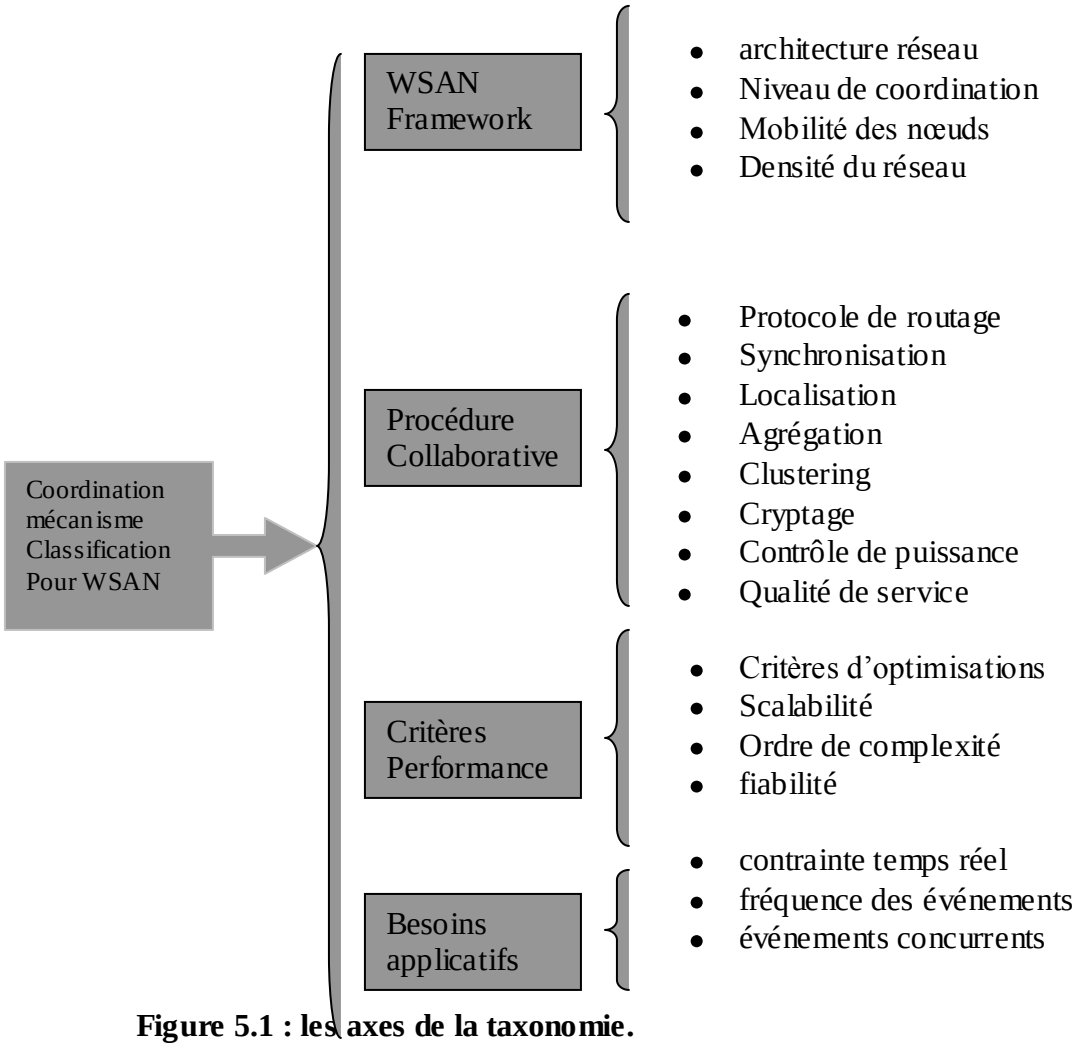


Figure 5.1 : les axes de la taxonomie.

V.2.1 WSAN framework :

Il s'agit de la structure d'un WSAN qui inclut :

1. l'architecture réseau : c'est la manière dont les informations sont captées et rapportées aux actionneurs. On distingue deux types d'architectures : automatique et semi automatique. Notre protocole utilise l'architecture automatique car la communication entre les capteurs et les actionneurs est directe. Cette architecture minimise la latence du fait que les capteurs sont plus proches des actionneurs. De plus, le rôle des actionneurs est de réagir promptement sans l'intervention du sink et donc il n'est pas obligatoire de parcourir un long chemin pour informer le sink des différents événements. En outre, dans l'architecture semi automatique, les nœuds d'un seul saut du sink peuvent tomber en panne, ce qui entraîne une perte de connectivité vers ce dernier. Les nœuds à un seul saut des actionneurs sont moins touchés par ce problème car un actionneur est généralement entouré par un grand ensemble de capteurs. Deux techniques seront utilisés pour palier cette difficulté : l'agrégation qui minimise le nombre de transmissions et la réallocation des actionneurs qui permet de centraliser chacun des actionneurs au milieu d'un groupe de capteurs.

2. le niveau de coordination : Notre protocole utilise la coordination capteur-capteur et capteur-actionneur pour la formation des clusters et la création du schéma d'agrégation ainsi que pour le positionnement des actionneurs. Les données rapportées sont envoyées à l'actionneur. La coordination actionneur-actionneur n'a pas été traitée dans ce travail car on suppose que les actionneurs sont identiques et qu'ils peuvent donc exécuter les mêmes tâches.

3. la mobilité des nœuds : les nœuds concernés par cette mobilité sont : les capteurs, les actionneurs et le sink. Dans notre protocole, l'ensemble des nœuds sont statiques. Du fait que la mobilité des actionneurs engendre un surcroît de la consommation d'énergie et parce qu'il est supposé que les actionneurs peuvent exécuter les mêmes tâches, notre approche nécessite un nombre suffisant d'actionneurs de manière à couvrir la zone entière.

4. la densité du réseau : c'est le ratio entre le nombre de nœuds déployés et la dimension de la surface. Dans notre cas, elle importe peu car l'algorithme de réallocation choisi prend en compte la densité et la distribution des capteurs sur les zones d'événements.

V.2.2 Procédures collaboratives :

Ces procédures (services) qui permettent la collaboration entre les éléments d'un WSANs, sont utilisées pour supporter l'échange d'information par le mécanisme de coordination. Elles intègrent:

1. un mécanisme de routage: il désigne les procédures implémentés pour sélectionner le prochain saut menant à la destination. Dans notre cas, le routage utilise les informations de localisation tout en prenant en considération la contrainte d'énergie et de latence. La construction du chemin de routage est à base d'événement, ce qui évitera l'intrusion de nœud non concerné. Selon l'état du réseau, les capteurs sélectionnent leurs prochains sauts, ce qui sera détaillé en section. 5.4.

2. une synchronisation : ce sont les techniques de conservation d'énergie nécessaires à l'association du temps et de la localisation de l'information avec les données captées. Aucun protocole n'est utilisé pour la synchronisation. Cependant, périodiquement chaque nœud diffuse sa position à l'ensemble de ses voisins afin de déterminer les routes défectueuses et mettre à jour les informations de routage. De plus, les paquets transmis doivent être contrôlés puisque chaque paquet a une durée de vie bien déterminée dans le réseau.

3. une agrégation : ce type de procédure permet de réduire l'énergie. Pour atteindre cet objectif, un seul paquet est crée comme étant la fusion des données de multiples sources. De plus, dans les WSAN, les capteurs doivent rapporter les informations nécessaires sans aucune redondance ni inconsistance, ce qui peut être vérifié par les nœuds d'agrégation et on évitera ainsi la circulation de données inutiles. Cependant, si chaque nœud attend les paquets de ces voisins, il est important de noter que les données ne doivent pas dépasser la latence exigée par le système. Un schéma d'agrégation est nécessaire. Les détails de ce dernier seront décrits dans la section 5.4.

4. une localisation : elle est implémentée pour fournir les informations de localisation géographique des capteurs et des actionneurs. Il est supposé que chaque nœud connaît sa position géographique. Plusieurs techniques peuvent être utilisées dans ce cas, GPS (global positionning system), ……

5. un clustering : il fournit une hiérarchie entre les nœuds. La substructure des noeuds représente un cluster et il y a un seul membre du cluster nommé cluster head. La formation des clusters est à base d'événement. A la suite de l'apparition d'un événement, seules les noeuds qui détectent l'événement participeront au routage. Pour chaque état, la construction du cluster diffère; ce qui sera expliqué dans la section 5.4.

6. un contrôle de puissance: Il permet de varier la puissance de transmission et de modifier ainsi la gamme de transmission/réception des noeuds. Une haute puissance permet de diminuer efficacement la latence et d'augmenter l'énergie consommée par le capteur.

7. une qualité de service: c'est le mécanisme utilisé pour fournir une garantie de service dans une architecture WSAN. Dans les WSAN, il est possible de garantir la qualité de service de deux manières différentes : La première fonctionnalité est basée sur la fourniture des réservations des ressources nécessaires à un nœud concerné par le rapport d'un événement. La seconde consiste à prioriser les actions à exécuter par les actionneurs en réponse à un événement donné. Afin de prendre en compte ce dernier point, les paquets d'événement sont classés par priorité, les paquets les plus importants étant transmis en premier pour qu'ils respectent les contraintes temps réel.

8. un cryptage des données : c'est l'implémentation des techniques de chiffrement pour protéger l'intégrité des données dans les WSAN. Aucun mécanisme de chiffrement ou de sécurité n'est implémenté dans ce protocole.

V.2.3 Performance

Cet axe est lié au critère utilisé pour estimer les performances du mécanisme de coordination.

1. critères d'optimisations : ces critères représentent les métriques sur lesquels se base le mécanisme pour atteindre les objectifs applicatifs. Notre framework prend en compte les critères suivants : fiabilité, énergie et nombre de pertes de paquets.

2. scalabilité : elle est liée à la capacité du réseau de maintenir le niveau de performance. Lorsque le nombre de nœuds tout comme les besoins augmente ; le nombre d'actionneurs nécessaires doit être plus grand pour éviter de saturer les actionneurs. Ces besoins peuvent être constitué par la fréquence des événements ou la taille de la zone d'événement.

3. Ordre de complexité : il fournit les mesures de la complexité du traitement de l'algorithme proposé en terme de temps d'exécution et de quantité mémoire nécessaire.

4. complexité de communication : pour son évaluation, différentes mesures tels que le nombre total de messages, le nombre de messages échangés et le temps de synchronisation sont les plus couramment utilisées. Notre protocole utilise les messages hello et les paquets de données pour transmettre les informations de contrôle.

5. Fiabilité : elle fournit les mesures du niveau de sécurité et de solidité (robustesse) du mécanisme de coordination. Un mécanisme de coordination sûr doit implémenter des fonctionnalités additionnelles pour garantir l'intégrité des données et pour éviter l'accès au WSAN par les intrus. D'autre part, la solidité est liée à la capacité du mécanisme de coordination de prendre en compte la tolérance aux pannes, tout en assurant la délivrance des données des capteurs aux actionneurs par le biais des connaissances ou des procédures de retransmission. Dans le cas de notre protocole, l'intégrité est vérifiée au niveau de chaque nœud de relais et cette fonctionnalité est assurée même en cas d'agrégation. Il convient de noter que tout au long de notre thèse, cette notion de fiabilité est appréhendée par référence à la réception des rapports dans les délais requis par le niveau applicatif.

V.2.4 Besoins applicatifs

Le mécanisme de coordination doit prendre en considération les obligations applicatives. Certains besoins incluent les contraintes temps réel, fréquence des événements et la possibilité de supporter les événements concurrents.

1. contrainte temps réel : elle est liée au temps nécessaire au mécanisme pour rapporter les événements et exécuter les actions. Deux types d'application sont possibles : celles qui, comme c'est le cas pour notre framework, tiennent compte de cette contrainte et celle qui ne le font pas.

1. fréquence des événements : le framework proposé est capable de produire la réponse adéquate quelque soit la périodicité des événements.

2. support des événements concurrents : la prise en charge par le système des événements concurrents se différencie selon quatre modèles : continu, à base d'événement, par observant des initiatives et hybride.

Continu : les paquets d'événements parcourent un chemin prédéfini pour atteindre un actionneur.

A base d'événement: l'itinéraire des paquets est déterminé par l'apparition d'événements.

En observant les initiatives : les informations recensées ne sont expédiées aux actionneurs que si ceux-ci les demandent.

V.3 Réallocation des actionneurs

Dans le chapitre précédant, le besoin d'algorithme d'allocations est clairement apparu. Ce framework exécute durant la phase initiale l'algorithme de réallocation. La coordination capteur-capteur, permettra de déterminer les positions des dominateurs qui seront communiquées au sink lui permettant ainsi de positionner physiquement les actionneurs. Les actionneurs sont statiques et ne changeront plus de position. Dans le but de répondre aux objectifs cités ci-dessous, le nombre d'actionneurs doit être suffisant pour couvrir l'intégralité de la zone d'événement. Comme notre choix s'est basé sur IDSC, le nombre d'actionneurs doit être égal au minimum au nombre de cercles pour couvrir l'ensemble des capteurs. Le rayon du cercle est égal à la gamme d'action des actionneurs.

Cet algorithme garantit que :

-la totalité de la zone d'événement est couverte c'est-à-dire que chaque partie de la zone est sous la responsabilité d'un actionneur. Chaque capteur sera sous la couverture d'un actionneur puisqu'il est inutile de mettre un actionneur pour une zone où on ne trouve pas de détection.

-l'ensemble des actionneurs sont équitablement distribués entre les capteurs. Ainsi, la densité des capteurs détermine le nombre d'actionneurs, ce qui permet d'éviter la surcharge des actionneurs.

-la distribution des actionneurs n'imposera pas de longs parcours aux paquets, ce qui permet en même temps de rationaliser la consommation d'énergie et de ne pas augmenter la latence, étant entendu que si le parcours est plus long, la puissance de transmission doit être plus grande ou alors il faudra transiter par une multitude de nœud de relais. Ainsi, l'algorithme choisi permet d'assurer que le nombre de saut maximal est égal à k sauts.

Les résultats de simulation, présenté dans [65], ont permet d'évaluer IDSC selon trois paramètres : délai point à point, couverture et réduction des chevauchements.

-Le délai est mesuré en fonction du nombre moyen de sauts qui sépare les nœuds de leurs cluster head. Ce délai est constant malgré l'augmentation du nombre de capteurs, ce qui rend l'algorithme scalable. IDSC décrémente le délai de 25% en comparaison avec un algorithme qui ne change pas de position aux actionneurs et qui relie le capteur à son actionneur le plus proche. Cette situation résulte de la limitation du délai à k saut.

-IDSC offre une couverture aussi complète que COLA, présenté dans le chapitre précédant, mais il présente l'avantage de diminuer le nombre d'actionneurs nécessaire et le délai requis.

La capacité d'IDSC à maintenir une haute couverture est due à la nature dynamique de l'algorithme. Les cluster-heads sont formés en fonction des besoins ce qui rend chaque capteur distant de k saut au maximum de son actionneur. Il est utile de souligner ici que la couverture est beaucoup moins assurée dès lors que les actionneurs sont maintenus dans leurs emplacements initiaux et si la liberté est laissée aux capteurs de choisir leurs actionneurs.

-Afin d'assurer une couverture maximale, il faut éviter les redondances. Pour cette raison, la figure suivante illustre une comparaison entre les zones de chevauchement pour IDCS (la figure droite) et une distribution sans réallocation. Les deux réseaux sont constitués de 4 actionneurs et d'une cinquantaine de capteurs. Visuellement, on constate qu'il y a moins de chevauchement dans IDCS ceci permet d'éviter la duplications des informations rapportés aux actionneurs.

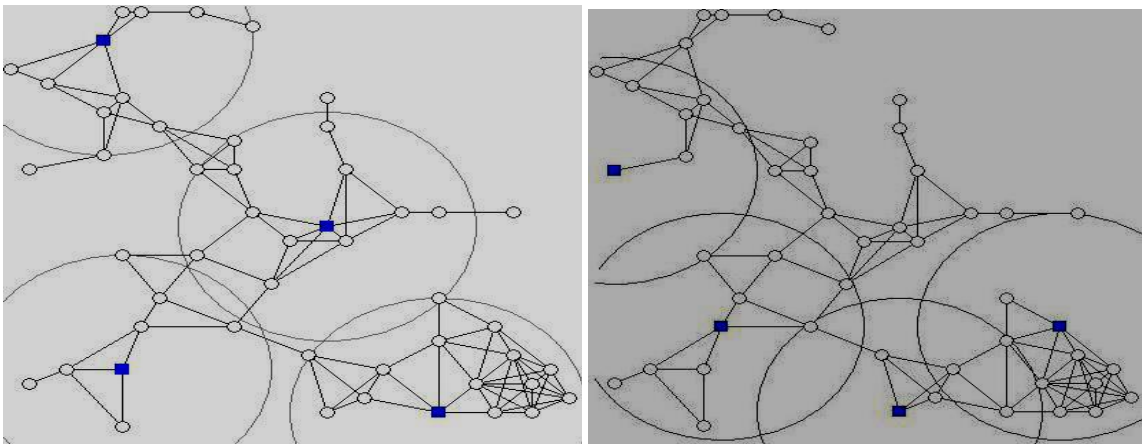


Figure V.1 : les zones de chevauchement pour IDCS et une distribution sans réallocation

V.4 Description détaillée de la solution

Lors de l'apparition d'un événement dans les WSANs, il n'y a pas un actionneur spécifique pour chaque message envoyé. Cette incertitude, due à l'existence de plusieurs actionneurs, pose un problème au niveau du routage. Sélectionner un actionneur est un challenge pour le nœud source. Le protocole de routage doit supporter la communication en temps réel, prendre en compte les priorités et économiser l'énergie. Enfin, il doit veiller à la stabilité du réseau en diminuant le nombre de panne provoqué par le manque d'énergie.

Apres l'exécution de l'algorithme de réallocation durant la phase initiale, les membres d'un même cluster sont connus. Le problème qui se pose alors réside dans le choix, à l'intérieur de ce cluster, du meilleur voisin qui doit être dans la même portée de transmission. Afin de ne pas introduire des nœuds inutiles dans le routage, la construction du cluster sera effectuée dés l'apparition d'un événement. De plus, les chemins statiques impliquent que les nœuds empruntent les mêmes chemins, ce qui les expose davantage aux pannes. Le modèle de routage à base de clutering est constitué par une arborescence dont la racine ou le cluster head est le membre le plus puissant. Dans notre cas, il représente un actionneur responsable de la collecte de toute donnée de son cluster.

L'arborescence garantit que les chemins ne contiennent aucune boucle de routage. Sa construction dépendra de l'état du réseau et de l'exigence de l'application. Le règlement de ce problème implique que les objectifs soient classés par ordre de priorités, l'actionneur étant chargé de contrôler l'état du réseau. Ce contrôle appelle obligatoirement la définition d'un certain nombre de seuils, exigés par l'application.

On définit donc :

- Be : pour chaque événement, Be qui détermine la durée de vie d'un paquet, est associée.
- Exp : c'est le seuil de tolérance du nombre de paquets expirés.
- Fail : c'est le taux de pannes des nœuds à ne pas franchir.

- Lost : c'est le nombre de paquets perdus à ne pas excéder.

Le réseau peut connaître quatre états de fonctionnements: énergie, fiabilité, start-up et agrégation. Idle représente l'état initial d'un capteur. Lors de la détection d'un événement ou la réception d'un paquet d'événement d'un voisin (source ou nœud de relais), il passe à l'état start-up dans le but d'assurer un compromis énergie-latence. Afin de garantir des chemins sans boucle, les capteurs créent, au début de chaque état, une arborescence qui diffère d'un état à un autre. A l'état initial, les nœuds n'ont aucune connaissance de la situation du réseau. L'actionneur est chargé de surveiller le fonctionnement du réseau, en calculant le nombre de paquets bien reçus ainsi que le nombre de pannes et de paquets perdus.

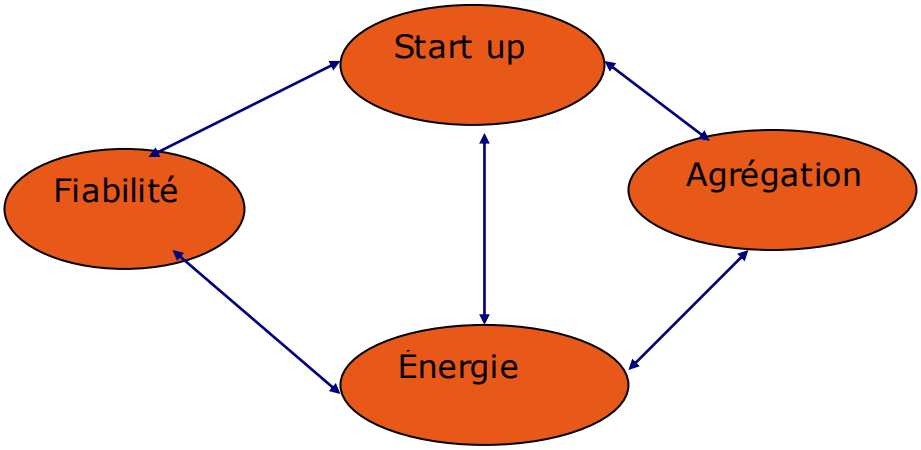
Il se base, à cet effet, sur les informations recueillies par les capteurs qui appartiennent à son arbre. Chaque capteur détermine le nombre de pannes grâce aux messages hello diffusés périodiquement. Le capteur recense les identités des voisins en panne (ceux qui n'ont pas envoyé hello ou qui ne lui ont pas répondu) et les communique à son supérieur hiérarchique lors de l'envoi des paquets de données.

Deux autres types de données, perdues et expirées (arrivée hors délai), ont pour mission de compléter le contrôle. Chaque capteur de niveau h+1 connaît par l'indexation des paquets déjà reçus de chaque voisin à la profondeur h, le nombre de ceux qui sont perdus et qu'il communiquera à son supérieur hiérarchique. Il est tenu toutefois de ne déclarer la perte d'un paquet qu'après avoir observé un délai qui peut varier en fonction du domaine d'application.

S'agissant des données expirées, chaque capteur compare le TTL[6] du paquet reçu avec le Be exigé par l'application et si un paquet possède un TTL>be, il est déclaré expiré et ne sera pas transmis. Cette situation conduira cependant à une incrémentation du nombre de paquets expirés.

Il est admis que dans un environnement caractérisé par la périodicité des événements, les informations de contrôle parviennent nécessairement à destination compte tenu de la transmission fréquente de paquets de données. Dès lors que les événement se raréfient, des paquets de contrôle devront être crées pour palier l'absence des paquets de données.

[6] Le temps que met un paquet de données pour atteindre la destination.



Les trois informations de contrôle sus-évoquées parcourent l'arborescence pour atteindre l'actionneur qui les compare aux seuils exigés par l'application de manière à pouvoir évaluer l'état du réseau. Pour cela, l'actionneur exécute l'algorithme ci-dessous.

Début
Si (nb_perdu<lost) et (taux_recu<exp) et (nb_pane<max_pane)
>passer a l'état start up
Si ((nb_perdu>=lost) ou (taux_recu>=exp)) et (nb_pane<fail)
>passer à l'état fiabilité
Si (nb_panne>fail) et ((nb_perdu<lost) et (nb_recu<exp))
> Énergie avec quantum fixe
Si (nb_panne>fail) et ((nb_perdu>lost) ou (nb_recu>exp))
> Agrégation avec quantum variable
Fin

S'il constate que la fiabilité du système est inférieure au seuil requis, l'actionneur décide de franchir l'état de fiabilité. Il convient de souligner ici que le franchissement de cet état est la conséquence d'une perte ou d'une expiration de paquets, elles-mêmes générées, pour l'essentiel, par une surcharge des nœuds. Cette situation engendre un problème de file d'attente dont la résolution dépend de la capacité d'assimilation des nœuds. Un classement des événements par ordre de priorité s'imposera également si l'on veut limiter les dégâts dus à l'expiration des paquets.

Le passage à l'état d'agrégation est le résultat d'un manque d'énergie décelé grâce au nombre de pannes. En effet, l'agrégation fusionne les paquets reçus durant un laps de temps donné qui peut être variable, si la fiabilité n'est pas assurée, ou fixe si elle l'est. La variabilité représente une tendance décroissante qui s'apprécie par rapport à la durée de vie du paquet arrivé et au chemin restant à parcourir et qui se rapproche de la situation d'envoi instantané des messages nécessaire à la diminution du nombre d'expiration des paquets.

Suite a l'exécution du protocole de réallocation, chaque actionneur couvre l'ensemble de son cluster et peut communiquer directement (à un seul saut) avec les nœuds appartenant à sa zone d'action. Lors de l'envoi du message qui permet aux nœuds de changer leur état, la diffusion s'opère à travers l'arbre. Afin de répartir les charges entre les actionneurs, la communication inter-actionneurs fait appel aux nœuds « border », déterminé à l'aide d'IDSC.

V.4.1 État Start up

Les nœuds connaissent leurs positions et celles de leurs voisins directs. L'information de localisation peut être obtenue par un système GPS [49] ou par d'autres techniques [40]. Chaque nœud connaît son dominateur grâce à l'algorithme de réallocation. La diffusion périodique des messages hello permet d'obtenir les informations sur les nœuds voisins. Ces messages contiennent l'identité de l'actionneur auquel ils appartiennent. Les décisions du routage seront basées sur la localisation, ce qui réduit l'utilisation de l'énergie d'une manière presque optimale.

Soit Vi l'ensemble des voisins de i ayant le même actionneur que i; Le capteur i sélectionne son voisin en calculant l'énergie minimale pour transmettre un paquet du nœud i à l'actionneur (en passant par le voisin j). L'énergie est calculée de la manière suivante :

$$\mathbf{E = 2Eelec + \beta d_{ij}^{\alpha} + 2Eelec + \beta d_{ja}^{\alpha},}\ \mathbf{[28]}$$

Où :

-α est l'exposant de perte de chemin ($2 \leq \alpha \leq 5$),

-β est une constante exprimé en [Joule/ (bits $*m^{\alpha}$)],

-Eelec représente l'énergie nécessaire pour transmettre ou recevoir un seul bit, exprimé en [Joule/bits].

Les résultats de simulations dans [28], prouvent que le coût du chemin entre la source et le nœud intermédiaire garantit un vrai compromis énergie-latence. Chaque capteur informe son actionneur du nombre de sauts et de la distance qui le sépare de l'actionneur. Ces informations seront expédiées aux capteurs et réutilisées pour calculer le quantum (laps de temps) durant la phase d'agrégation.

Chaque capteur évalue son état d'énergie, si celle-ci ne suffit pas pour l'envoi de trois paquets, il diffuse une alarme pour empêcher ses voisins de le choisir comme prochain saut.

Si le manque d'énergie concerne l'ensemble des voisins d'un capteur donné, celui-ci devra obligatoirement recouvrir à face2 [45] pour la sélection du prochain chemin.

Le code ci-dessous est exécuté par chaque capteur i à l'état start-up :

Algorithme 2:

Debut
Cout= ∞
Tq j dans Vi:
Faire: si ($2Eelec + \beta d_{ij}^{\alpha} + 2Eelec + \beta d_{ja}^{\alpha}$) < cost
Debut
$cost = 2E_{elec} + \beta d_{ij}^{\alpha} + 2E_{elec} + \beta d^{\alpha}{}_{js_k}$
nexthop=j
finsi
Fin

Cet algorithme garantit des chemins sans boucle et permet la construction d'une arborescence.

Selon l'algorithme de réallocation, un nœud de capteur peut recevoir le message « dominator » de plusieurs dominateurs dont il choisira le plus proche. Le choix du voisin sera effectué parmi ceux qui partagent le même actionneur. Comme le nœud i choisit son voisin selon la règle des deux sauts c'est-à-dire en calculant la distance en i et j et entre j et son plus proche actionneur qui représente le même actionneur que celui de i, le prochain saut élu, plus proche de l'actionneur que de i garantit un chemin sans boucles.

V.4.2 Etat fiabilité

Le but est de minimiser la latence au maximum car cet état n'est franchi que si le taux de paquet reçu avec un ttl expiré est très élevé et n'est pas toléré par l'application. La minimisation du nombre de saut en envoyant le paquet aux nœuds le plus distant minimise la latence. Cependant, l'envoi à chaque fois au même nœud surtout que l'occurrence des événements est très fréquente peut causer un retard remarquable. Une approche hybride qui intègre ces deux contraintes est dès lors proposée. L'approche que nous proposons prend en compte la priorité des paquets ainsi que la surcharge des nœuds destinataires.

Les événements sont donc classés selon leur importance dans des files d'attentes. Chaque nœud échange avec ses voisins l'état de ses files. Apres chaque envoi, le nœud i met à jour la valeur de Sj qui représente le taux de paquets déjà envoyé par i au noeud j. Chaque nœud a un nombre limité de paquets qu'il peut envoyer à l'un de ces voisins pour ne pas le surcharger;

Le nombre maximal de paquets que peut recevoir j à partir de i est égal :

$\lambda_{ij} = [1-\lambda_{high} * S- (R/(1-\lambda_{high}*S)*d_{qmax})]/S;$ --------- [29]

Au niveau de l'algorithme 3, **hij** représente la distance à partir de i à j vers l'actionneur **a**.

Parmi les nœuds satisfaisant la contrainte **λij**< Sj, le capteur i choisit celui qui est le plus proche de Ai. Avant la transmission du paquet, i changera son TTL.

Algorithme 3 :

```
Debut
Tq j in Vi:
Début
        Calculer λij
        Calculer hij
        Si (λij< Sj) et (hij >hmin)
       Debut
                hmin=hij;
                next=j;
        fsi

Fin
```

5.4.3 État Agrégation

Le but est de minimiser l'énergie consommée. Afin de ne pas gaspiller des ressources pour la communication et la construction d'un arbre, le même arbre que celui de la phase start-up sera utilisé. La nouveauté réside ici dans la tolérance aux pannes. A la réception des données d'événement de chaque descendant, chaque capteur procède à l'agrégation des données après avoir éliminer les données inconsistantes. Chaque nœud attend un certain délai pour transmettre ses paquets regroupés. Le délai que nous introduisons dans notre solution d'agrégation dépend de plusieurs paramètres : le nombre de saut pour atteindre l'actionneur (c'est-à-dire le nombre de saut restants à parcourir), la distance qui sépare le capteur de l'actionneur et le TTL du paquet. Les deux premiers paramètres sont fixes pour chaque capteur et pour un événement donné et seront calculées à l'aide de à l'état start up. Le dernier paramètre (TTL) est pris en compte car un nœud de profondeur h peut tout aussi bien recevoir un paquet originaire de profondeur h-1 qu'un autre paquet de profondeur h-2.
L'intervalle sera égal à :

I= (Be- (nombre de saut* temps de traitement + (dst1+ dst2+ ….+ dstn)*bp))/ nombre de saut

Comme le temps total que met un paquet pour atteindre sa destination est égal à :

D=nombre de saut* temps de traitement + (dst1+ dst2+ ….+ dstn)*bp + I* nombre de saut ;

Avec : **D<be ;**

Où :

- **bp :** représente la bande passante.
- **dsti :** représente la distance pour atteindre le voisin i (suivant l'arborescence établie en phase start up).
- **I :** represente le delai moyen d'attente au niveau de chaque saut.

Chaque nœud calcule le délai d'attente qu'il compare avec l'intervalle restant du quantum en cours. Si la différence est négative, le capteur devra modifier le quantum courant pour l'adapter au délai d'attente considéré. Dans le cas contraire, le paquet sera transmis à l'expiration du quantum en cours.

Algorithme4 :

```
Debut
Cout= ∞
Tq j dans Vi:
Faire : si (2Eelec + βdαij + 2Eelec + βdαjsk) < cost
        cost = 2Eelec + βdαij + 2Eelec + βdαjsk
        nexthop=j
      finsi
Si nexthop voisin faire
                    dataj = dataj+ data_recu;
                    Calculer i
                   Si (now-timer) >I faire
                   timer= now ;
                  cancel timer;
                  relancer timer avec i;
                  finsi
finsi
Fin
```

5.4.4 État Energie

La méthode précédente regroupe les paquets de manière à ce qu'aucun d'entre eux n'expire. Pour cela, à la réception d'un paquet, le délai de séjour au niveau du nœud dépend du temps restant pour l'expiration du paquet ainsi que du chemin qui reste à parcourir. Cette méthode tend à préserver un taux de fiabilité proche de 100% ;

Bien que cet algorithme s'efforce de préserver la fiabilité de chaque paquet reçu, le quantum est variable et il peut diminuer. Cette diminution implique que le nombre de paquets à regrouper n'est pas assez grand, ce qui provoque une consommation d'énergie. C'est

pourquoi un délai maximum est fixé au départ en calculant d à l'aide le l'équation indiquée ci-dessus. Dans cet état, il s'agit d'un délai invariable même si des pertes de paquets sont enregistrées car il importe de garder les nœuds en vie pour éviter toutes défragmentation du réseau.

5.5 Résultat des simulations

Dans cette section, nous présentons l'évaluation de performance de notre framework de coordination. Afin de comparer les performances avec les solutions existantes, nous avons implémenté les protocoles déjà évoqués dans les chapitres précédents [29],[28],[60]. L'implémentation de ces quatre protocoles a été réalisé par utilisant du simulateur J-Sim [75], qui implémente la pile des protocoles des capteurs de la couche physique à la couche application, en incluant le protocole de la couche MAC 802.11, UDP transport et CBR traffic.

J-sim est open source. C'est un environnement de simulation à base de composant développé par les deux universités UIUC et OHIO. J-sim est dédié aux réseaux de capteurs puisqu'il contient les packages nécessaires à ces types de réseaux.

On considère trois scénarios de simulation. L'épicentre de la zone d'événement est sélectionnée d'une manière complètement arbitraire. Quatre actionneurs sont aléatoirement déployés dans chaque scénario. Au niveau du scénario 2, le nombre d'événement est égal à deux avec une fréquence d'occurrence qui représente le nombre d'arrivée par seconde pour le problème de file d'attente égale 4event/s pour le premier événement et 5event/s pour le second. L'importance du premier événement équivaut à 1 alors que celle du second est égale à 0.3. A la différence du scénario 1, le scénario 3 suppose que la fréquence d'occurrence des événements est de 5 événements par secondes.

Les paramètres de simulation du modèle d'énergie sont choisis comme suit: Eelec = 50nJ/bit, β = 100pJ/bit/m_, α = 4. Le tableau ci dessous indique les paramètres communs aux trois scénarios :

Parameteres de Simulations	**Valeurs**
Taille du réseau	500*500 métre
Nombre de capteurs	100
Nombre d'actionneurs	4
Placement des noeuds	Aléatoire
Couche Mac	IEEE 802.11
Bande passante	2Mbps
Taille des paquets	32 bytes
Latence limite	2sec
Puissance radios des capteurs	12 métres
Taille de la zone d'événement	250 mètres
Fréquence des evenemnets	20 events/seconde pour le premier scénario. 5 events/seconde pour le troisième scénario.

Comme le comportement du réseau dépend des différents paramètres de l'application, nous nous référons successivement dans cette section aux différents configurations : start-up, fiabilité, l'agrégation et énergie. Cette approche nous permet de décrire l'évaluation des performances des différents états et de montrer les avantages de la solution proposée, indépendamment du choix des paramètres qui régissent les transitions entre les états qui seront étudiées ultérieurement.

Puisque les résultats des simulations [28] ont démontré que le choix du chemin selon la règle des deux sauts assure un meilleur compromis energie-latence, nous l'avons donc choisi pour la construction des chemins de communication et pour prouver ainsi le rôle de la réallocation dans la coordination capteur-actionneur.

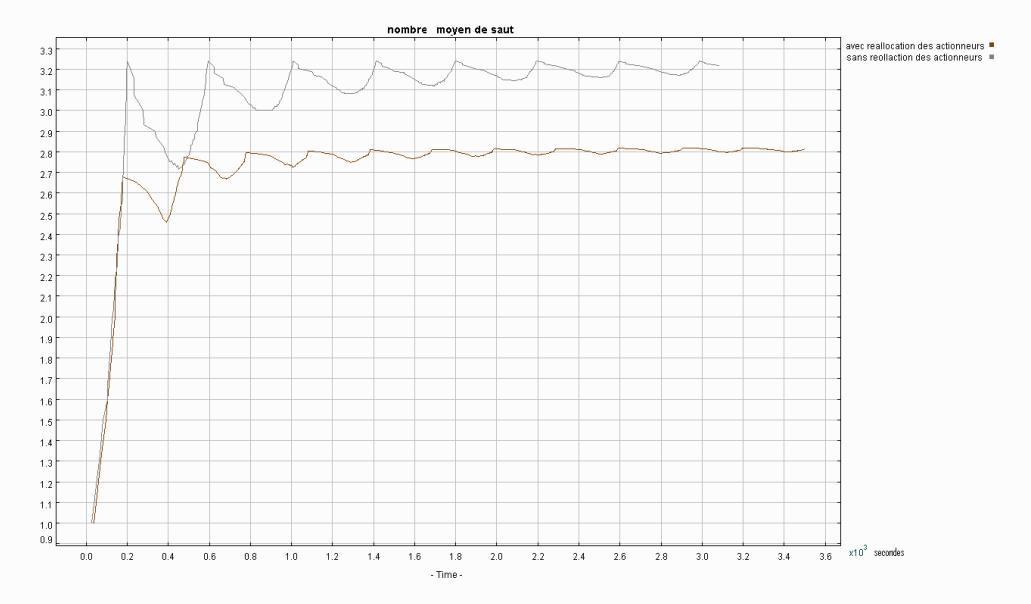


Figure 3: le nombre moyen de saut en fonction du temps.

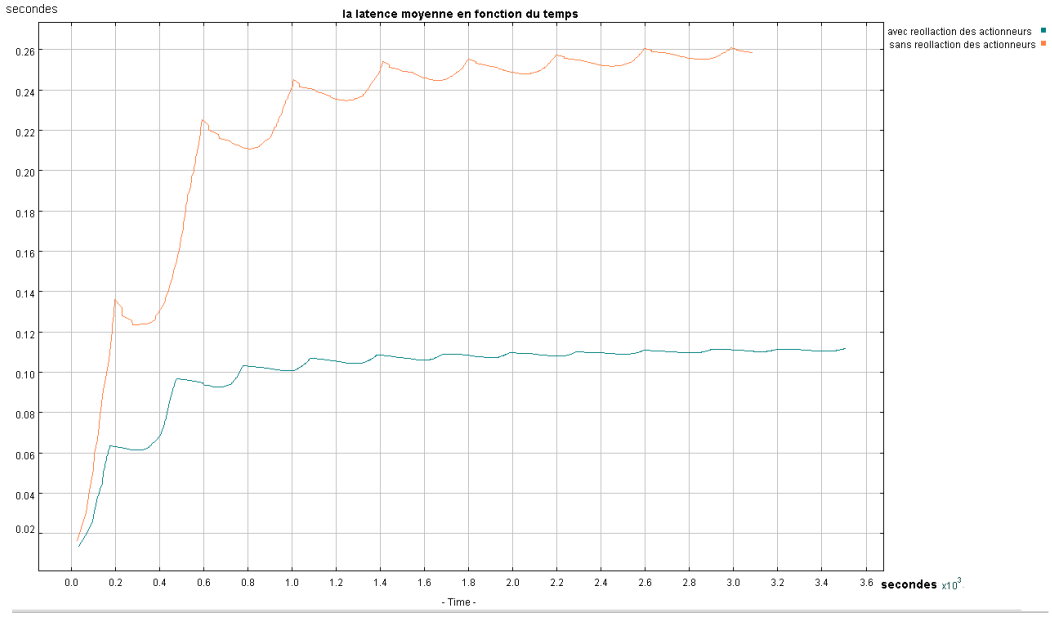


Figure 4: la latence moyenne en fonction du temps.

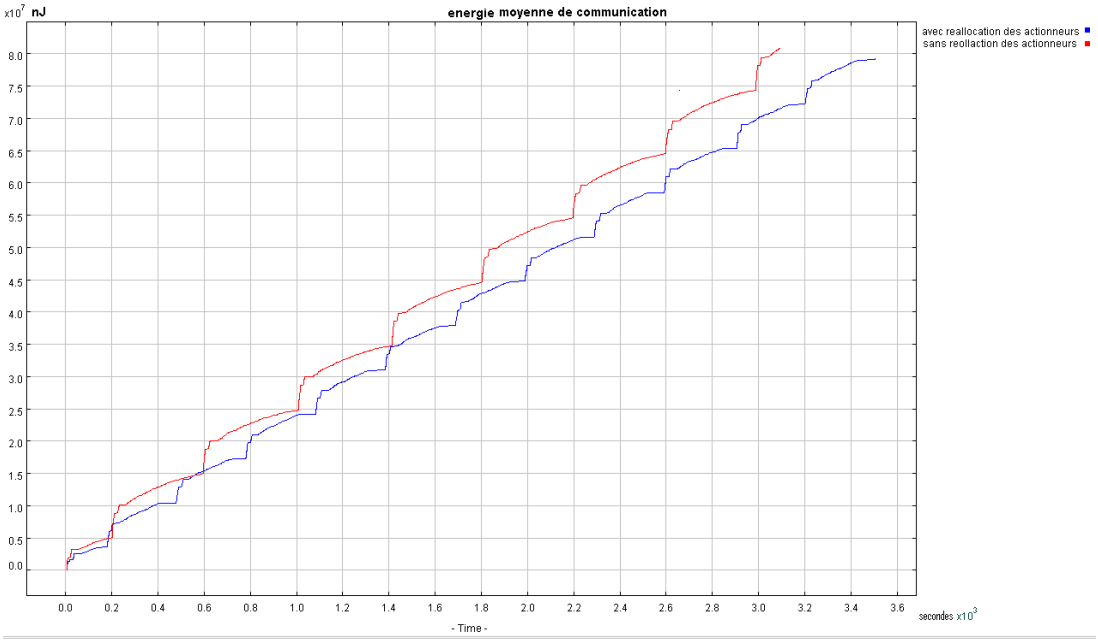


Figure 5: l'énergie moyenne en fonction du temps.

Les figures 4 et 5 montrent que dans un réseau avec réallocation des actionneurs, la consommation d'énergie et de latence diminue sensiblement grâce à la réduction du nombre de sauts (voir figure 3) provoquée, lorsque l'on utilise le protocole IDSC, par la proximité des actionneurs par rapport aux capteurs. Avant toute comparaison, il convient d'exécuter l'algorithme d'allocation des actionneurs.

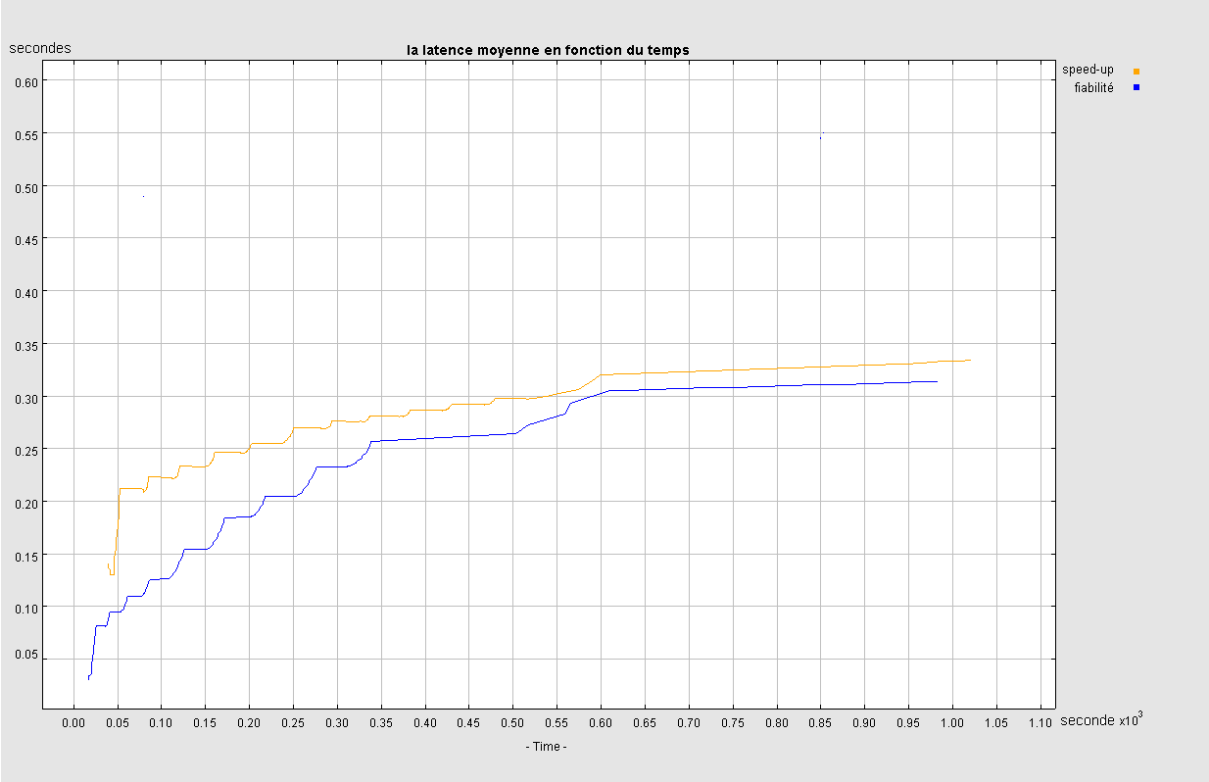


Figure 6: la latence moyenne en fonction du temps.

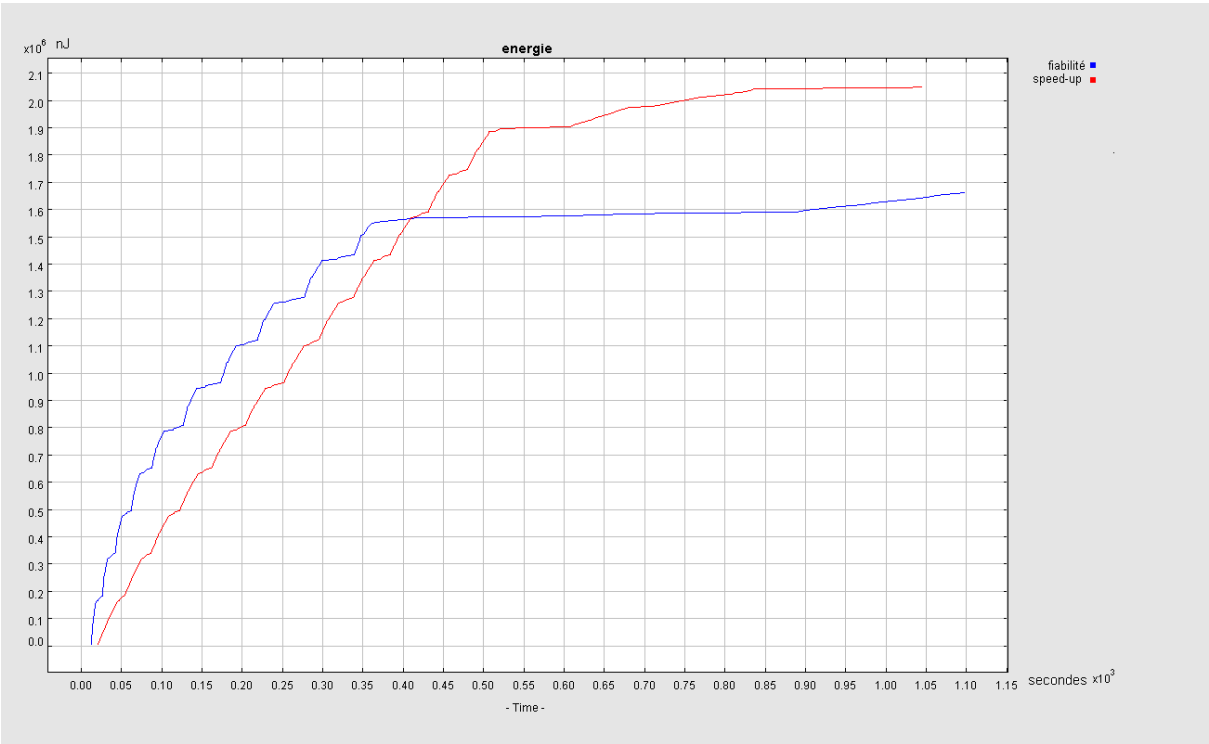


Figure 7 : l'énergie moyenne en fonction du temps.

Dans les figures 6 et 7, l'état speed up [28] a été comparé avec l'état fiabilité de notre protocole en utilisant le scénario 2. Dans ce cas précis, la réduction de la latence résulte aussi bien de la réduction du nombre sauts que de la prise en compte de la charge des voisins qui permet de conserver l'énergie. En effet, lors du routage, il n'est pas indispensable de choisir le voisin le plus distant et d'utiliser une puissance de transmission élevée.

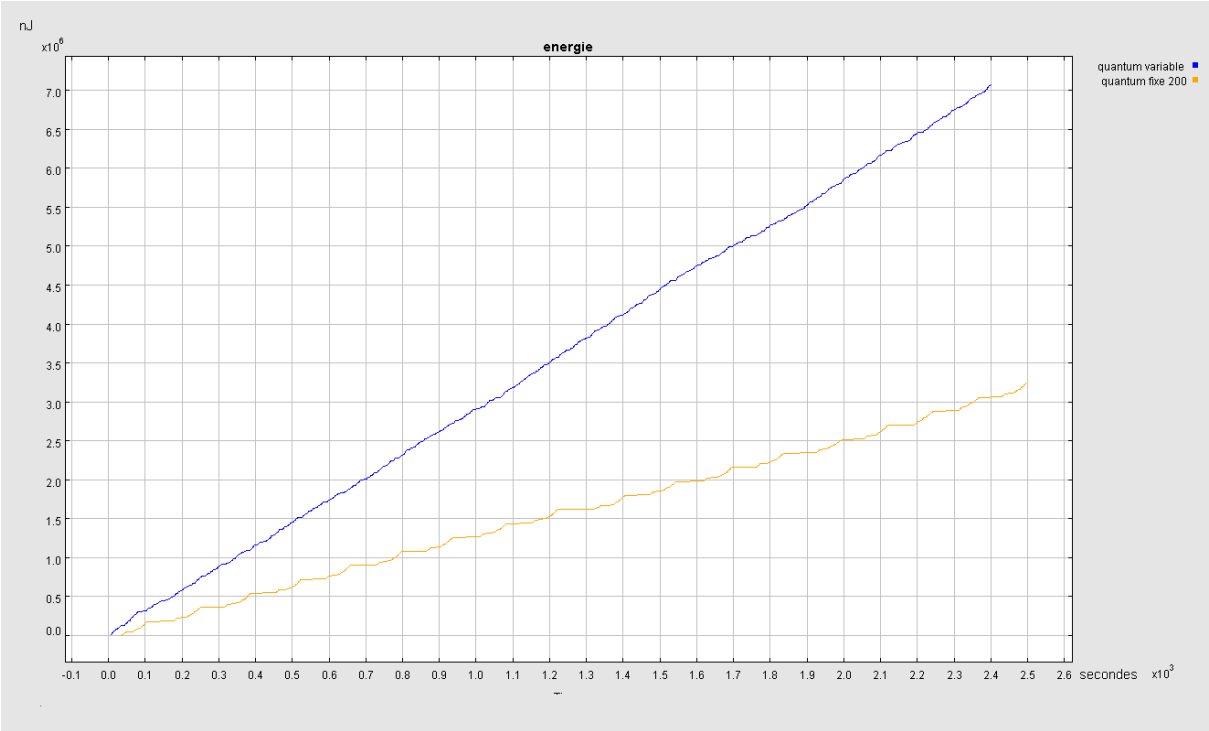


Figure 8 : l'énergie moyenne en fonction du temps.

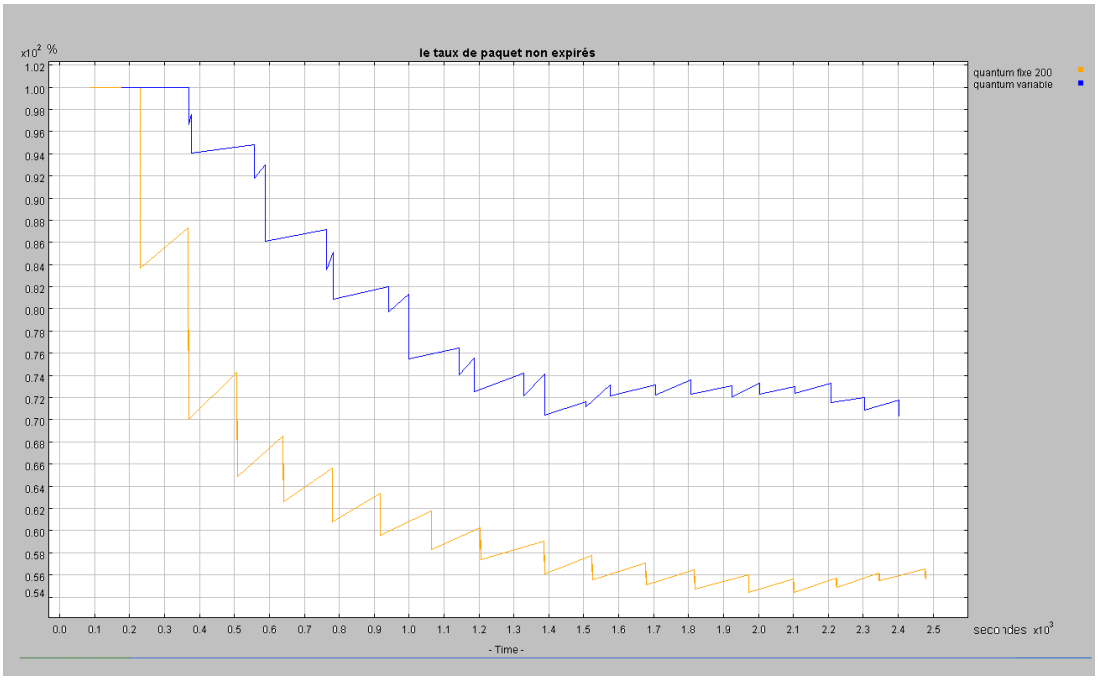


Figure 9: le taux d'expirations en fonction du temps.

Dans les figures 8 et 9, les deux schémas d'agrégation (avec quantum variable et fixe) ont été comparés. Si la fiabilité est meilleure dans un schéma à quantum variable qui prend en charge les ttl des paquets à transmettre, il n'en demeure pas moins que la diminution de l'intervalle d'agrégation engendre un surcroît de consommation d'énergie.

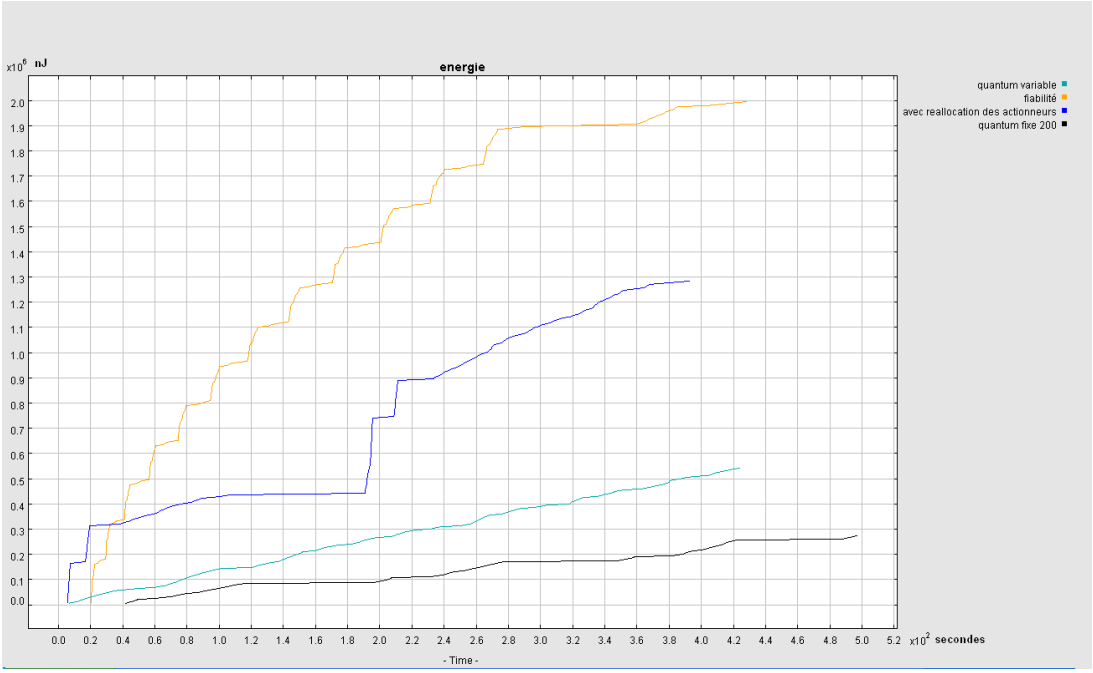


Figure 10 : énergie consommé en fonction du temps.

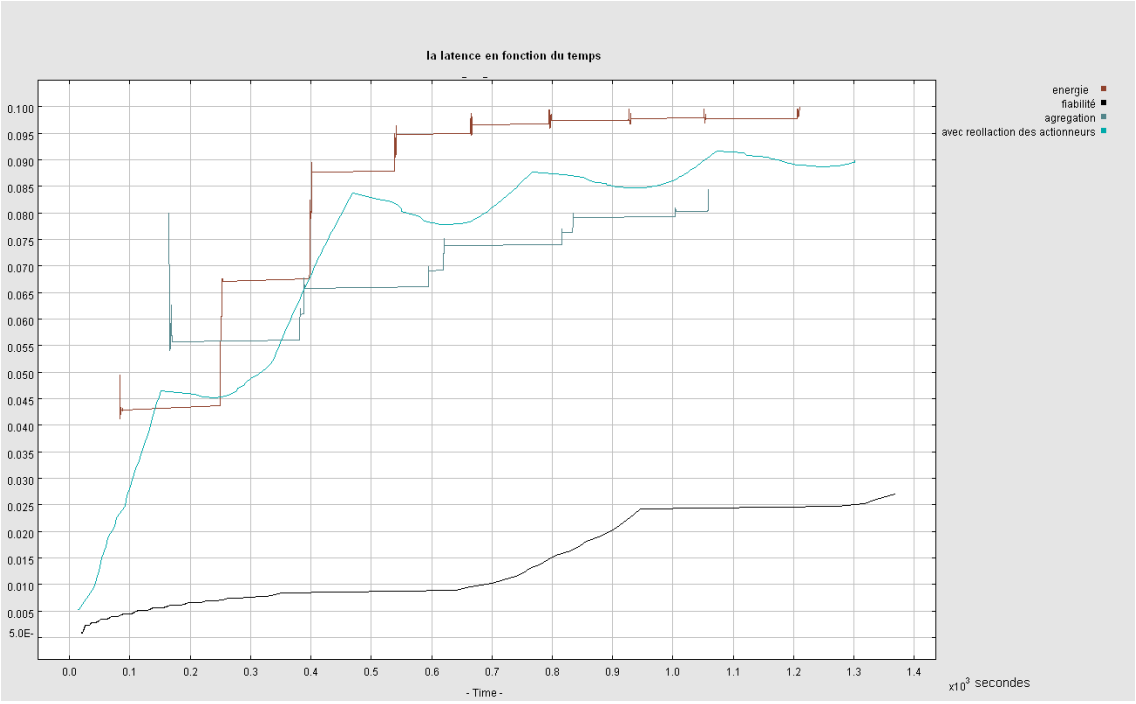


Figure 11: la latence moyenne en fonction du temps.

Dans les figures 10 et 11, les quatre états ont été comparés, en terme d'énergie et de latence en utilisant le troisième scénario. D'après ces graphes et ceux déjà présentés plus haut, on peut constater que la consommation d'énergie est meilleure, dans l'état énergie puis dans l'état agrégation puis dans l'état start up et enfin dans fiabilité, alors que la latence varie dans l'ordre croissant suivant : fiabilité d'abord, agrégation et start up ensuite, énergie enfin. Ces résultats confirment notre schéma de transition.

5.6 Conclusion

A la différence des frameworks développés dans les études antérieurs, Ce travail s'inspire des systèmes multi robots pour la réallocation des actionneurs ajoutant ainsi un nouveau critère à la taxonomie de référence [71]. Dans cette approche qui permet une meilleure organisation du réseau préalablement à toute opération de clustering, chaque actionneur joue le rôle d'un cluster head et est chargé de contrôler l'état du système selon le nombre de pannes, d'expirations et de données perdues. La finalité consiste ici à proposer un framework qui garantit un compromis energie-latence en adaptant le schéma d'agrégation en fonction de la fiabilité du réseau.

Notre approche ouvre ainsi des perspectives intéressantes de recherche, qu'il s'agisse d'étendre le framework en vue de la prise en compte des actionneurs mobiles, de développer les travaux de simulation destinés à l'évaluation des transitions d'état, de proposer un mécanisme de coordination actionneur-actionneur ou enfin de placer le framework en milieu réel d'application.

Conclusion générale

Par leur simplicité et leur utilité dans divers domaines d'activité, les réseaux de capteurs WSN se sont imposés en milieu industriel. Toutefois, ces réseaux se limitent à la fonction de contrôle d'environnement sans être capable de réagir.

L'émergence d'une nouvelle classe de réseau « Wireless Sensor and Actor Networks (WSAN)» a permis d'apporter une amélioration dans ce domaine car les WSAN sont tout autant capables de capter les contraintes environnementales que de réagir.

L'hétérogenité d'un tel réseau impose, cependant, l'existence d'un mécanisme de coordination capteur-actionneur et actionneur-actionneur. La coordination s'attache à la manière dont les nœuds gèrent leurs réseaux. Nous nous sommes intéressés à cette exigence de coordination et dans ce cadre nous avons études deux frameworks existants qui nous ont particulièrement inspiré [28][29]. Dans ces deux solutions, les capteurs coordonnent les uns avec les autres pour former des clusters. Pour chaque cluster, il y aura un actionneur rassembleur de données. La formation des clusters satisfait les conditions suivantes:

- Le temps de transmission de l'événement aux actionneurs doit être réduit au minimum puisque une réaction rapide est exigée dans un tel système.
- les voies de communication entre les capteurs et les actionneurs seront choisies parmi celles qui consomment le moins d'énergie.
- L'ensemble des actionneurs doit couvrir la totalité de la zone d'événement.

Le positionnement des capteurs dépend de l'application et des zones à contrôler. Une solution très intéressante consiste à placer ces actionneurs au bon endroit pour qu'ils puissent recevoir l'information le plus tôt possible, tout en préservant les ressources énergétiques des capteurs. L'un des problèmes de conception dans les WSANs réside dans la répartition des actionneurs de telle sorte que l'ensemble des actionneurs doit couvrir la zone entière en minimisant le nombre de sauts qu'un paquet de données doit parcourir jusqu'à atteindre un actionneur. Disposer d'une couverture d'actionneur tout en réduisant la latence est un vrai challenge. Plusieurs travaux nous ont particulièrement intéressé [29][56][60][65].

Le protocole requis pour notre cas été IDSC car il répond à l'ensemble des besoins suivants :

- la totalité de la zone d'événement est couverte c'est-à-dire que chaque partie de la zone est sous la responsabilité d'un actionneur. Chaque capteur sera sous la couverture d'un actionneur puisqu'il est inutile de mettre un actionneur pour une zone où on ne trouve pas de détection.

- l'ensemble des actionneurs est équitablement distribué entre les capteurs. Ainsi, la densité des capteurs détermine le nombre d'actionneurs, ce qui permet d'éviter la surcharge des actionneurs.
- la distribution des actionneurs n'imposera pas de longs parcours aux paquets, ce qui permet en même temps de rationaliser la consommation d'énergie et de ne pas augmenter la latence, étant entendu que si le parcours est plus long, la puissance de transmission doit être plus grande ou alors il faudra transiter par une multitude de nœud de relais. Ainsi, l'algorithme choisi permet d'assurer que le nombre de saut maximal est égal à k sauts.

Apres les études des différents protocoles qui ont fait l'objet des 4 premiers chapitres, nous nous sommes arrivées à une solution qui combinent le mécanisme de clustering ainsi que celui de la réallocation des actionneurs. Pour une description complète de notre framework, une taxonomie a été prise comme référence [71]. Nous avons, par ailleurs, opté pour une solution de réallocation préalable à toute opération de clustering pour laquelle chaque cluster sera dirigé par un actionneur chargé de contrôler l'état du système en fonction du nombre de pannes, d'expirations et de données perdues. Notre système connaît quatre états de fonctionnalités : start up, fiabilité, agrégation et énergie. Le franchissement de l'état fiabilité est la conséquence d'une perte ou d'une expiration de paquets. Le passage à l'état d'agrégation est le résultat d'un manque d'énergie décelé grâce au nombre de pannes. En effet, l'agrégation fusionne les paquets reçus durant un laps de temps donné qui peut être fixe ou variable selon que la fiabilité assurée ou non. Notre schéma d'agrégation s'adapte bien à la fiabilité du réseau contrairement aux travaux antérieurs.

L'extension du framework aux actionneurs mobiles, le développement de simulations de manière à évaluer les transitions d'état, la mise en place d'un mécanisme de coordination actionneur-actionneur constitue autant de perspectives ouvertes par notre approche. Egalement, il importera de tester la fiabilité de notre framework à la faveur de son déploiement en milieu réel.

Bibliographie :

[1]	I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: A survey, Computer Networks 38 (4) (2002) 393–422.
[2]	A. Helmy. Location-free Contact Assisted Poer-Efficient Query Resolution for Sensor Networks, Mobile Computing and Communications Review, Vol. 8, No. 1, pp. 27-47, 2004.
[3]	J. M. Hellerstein, W. Hong, S. Madden, K. Stanek. Beyond Average: Toward Sophisticated Sensing with Queries, F. Zhao and L. Guibas (eds.) Proceedings of Second International Workshop Information Processing in Sensor Networks, IPSN 2003, Palo Alto, CA, USA, April 22-23, pp. 63-79, 2003.
[4]	P. Kourouthanasis and G. Rousos. Developing Consumer-Friendly Pervasive Retail Systems, IEEE Pervasive Computing, Vol. 2, No. 2, pp. 32-39, 2003
[5]	J. Gehrke and S. Madden. Query Processing in Sensor Networks, IEEE Pervasive Computing, Vol. 3, No. 1, pp. 46-55, 2004.
[6]	W. Hu, N. Bulusu, S. Jha, "An anycast service for hybrid sensor/actuator networks" in: Proceeding of the 15th IEEE Int. Symposium on Personal, Indoor and Mobile Radio Communications (PIMRS), Barcelona, Spain, 5-8 September 2004.
[7]	H. S. Kim, T. F. Abdelzaher, W. H. Kwon, "Minimum energy asynchronous dissemination to mobile sinks in wireless sensor networks" in: Proceeding. of the First ACM Int. Conf. on Embedded Networked Sensor Systems (ACM Sensys 2003), November 2003, pp. 193-204.
[8]	T. He, J. Stankovic, C. Lu, T. Abdlzaher, "SPEED: A realtime routing protocol for sensor networks" in: Proceeding. IEEE Int. Conf. on Distributed Computing Systems (ICDCS), Rhode Island, USA, May 2003, pp. 46-55.
[9]	V. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, N. Shroff, "A minimum cost surveillance sensor network with a lifetime constraint" IEEE Transactions on Mobile Computing, in press.
[10]	David B. Johnson. Routing in Ad Hoc Networks of Mobile Hosts. In Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications, pages 158–163, December 1994
[11]	Charles E. Perkins and Elizabeth M. Royer, "Ad-hoc On-Demand Distance Vector Routing," In Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99), pp. 90–100, New Orleans, LA, February 1999.
[12]	P. Jacquet, P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum, L. Viennot «Optimized Link State Routing Protocol for Ad Hoc Networks». Proceedings of the 5th IEEE Multi Topic Conference (INMIC 2001)
[13]	Charles E. Perkins and Pravin Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers, " In Proceedings of the SIGCOMM '94 Conference on Communications Architectures, Protocols and Applications, pp. 234–244, August 1994. A revissed version of the paper is available from http://www.cs.umd.edu/projects/mcml/pub.html.
[14]	B. Tavli, W. Heinzelman, "TRACE: Time reservation using adaptive control for energy efficiency" IEEE Journal on Selected Areas of Communication 21(10)(2003)1506-1515.
[15]	J. Kuri, S. Kasera, "Reliable Multicast in Multi-access Wireless LANs", *Wireless Networks*, 7(4):359-369, July 2001.
[16]	M. Caccamo, L. Y. Zhang, L. Sha, G. Buttazzo "An implicit prioritized access protocol for wireless sensor networks" in: Proc. IEEE Real-Time Systems Symp. December

	2002, pp. 39-48.
[17]	J. Hill, R. Szewcyk, A. Woo, D. Culler, S. Hollar, K. Pister, "System Architecture Directions for Networked Sensors" ACM ASPLOS, 2000, pp. 93-104.
[18]	Dung van Dinh, Minh Duong Vuong, Hung Phu Nguyen, Hoa Xuan Nguyen, Wireless sensor actor networks and routing performance ansalysis, in: Proceedings of the International Worshop on Wirless Ad-hoc Networks, 2005.
[19]	M.J. Coates, Evaluating Causal Relationships in Wireless Sensor/Actuator Networks (pdf), Proc. IEEE ICASSP, Philadelphia, PA, Apr. 2005.
[20]	Sundresh, S., G. Agha, K. Mechitov, W. Kim, and Y. Kwon. Coordination services for wireless sensor networks. International Workshop on Advanced Sensors, Structural Health Monitoring, and Smart Structures (Tokyo, Japan, Nov. 2003). Proceedings of the International Workshop on Advanced Sensors, Structural Health Monitoring, and Smart Structures (2003)
[21]	B.P. Gerkey, M.J. Mataric, Sold!: Auction methods for multi-robot coordination, IEEE Transactions on Robotics and Automation, Special Issue on Multi-robot Systems 18 (5) (2002) 758–768.
[22]	C. Lu, B. M. Blum, T. F. Abdelzaher, J. A. Stankovic, and T. He, "RAP: a real-time communication architecture for large-scale wireless sensor networks," in Proc. of IEEE RTAS, San Jose, CA, USA, Sep 2002.
[23]	E. Felemban, C.-G. Lee, E. Ekici, R. Boder, and S. Vural, "Probabilistic QoS guarantee in reliability and timeliness domains in wireless sensor networks," in Proc. of IEEE Infocom 2005, Miami, FL, USA, Mar 2005.
[24]	V. P. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, N. Shroff, "A minimum cost heterogeneous sensor network with a lifetime constraint," IEEE Transaction on Mobile Computing, vol. 4, no. 1, Jan/Feb 2005.
[25]	M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh, "Exploiting Heterogeneity in Sensor Networks," in Proc. of IEEE Infocom 2005, Miami, FL, USA, Mar 2005.
[26]	W. Hu, S. Jha, and N. Bulusu, "A communication paradigm for hybrid sensor/actuator networks," in Proc. Of the 15th IEEE Int. Symposium on Personal, Indoor and
[27]	E. Cayirci, T. Coplu, and O. Emiroglu, "Power aware many to many routing in wireless sensor and actuator networks," in Proc. of the Second European Workshop, 31 Jan-2 Feb 2005, pp. 236 – 245.
[28]	T. Melodia, D. Pompili, V. C. Gungor, and I. F. Akyildiz, "A distributed coordination framework for wireless sensor and actor networks," in Proc. of ACM Mobihoc, Urbana-Champaign, IL, USA, 2005, pp.99-110.
[29]	Edith Ngai; Yangfan Zhou; Michael Lyu; Jiangchuan Liu, Reliable Reporting of Delay-Sensitive Events in Wireless Sensor-Actuator Networks, *Proceedings of IEEE MASS 2006*, October 2006
[30]	D. V. Dinh, M. D. Vuong, H. P. Nguyen, H. X. Nguyen, "Wireless sensor actor networks and routing performance analysis," International Workshop on Wireless Ad-hoc Network, London, UK, 23-26 May 2005.
[31]	A. Durresi and V. Paruchuri, "Geometric broadcast protocol for sensor and actor networks," Int. Conference on Advanced Information Networking and Applications, 28-30 Mar 2005, pp. 343 – 348.
[32]	J. Li, J. Jannotti, D. D. Couto, D. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," in Proceedings of ACM/IEEE MobiCom 2000, Boston, Massachussets, 2000, pp. 120–30.
[33]	W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks,"IEEE Transactions on

	Wireless Communications, vol. 1, no. 4, pp. 660–670, Oct. 2002.
[34]	O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," in Proceedings of IEEE INFOCOM 2004, Hong Kong S.A.R., P.R. China, March 2004.
[35]	V. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, and N. Shroff, "A minimum cost heterogeneous sensor network with a lifetime constraint," IEEE Transactions on Mobile Computing, vol. 4, no. 1, pp. 4–15, Jan. 2005.
[36]	F. Kuhn, T. Moscibroda, and R. Wattenhofer, "Initializing newly deployed ad hoc and sensor networks," in Proceedings of ACM MobiCom 2004, Philadelphia, PA, September 2004.
[37]	HEINZELMAN, W., CHANDRAKASAN, A., and BALAKRISHNAN, H., "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, pp. 660–670, October 2002.
[38]	B. Krishnamachari, D. Estrin, and S. Wicker, "The impact of data aggregation in wireless sensor networks," in Proceedings of IEEE DEBS 2002, Vienna, Austria, July 2002.
[39]	AHUJA, R. K., MAGNANTI, T. L., and ORLIN, J. B., *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, New Jersey: Prentice Hall, February 1993.
[40]	HIGHTOWER, J. and BORRIELLO, "Location systems for ubiquitous computing," *IEEE Computer*, vol. 34, pp. 57–66, August 2001.
[41]	SUNDARARAMAN, B., BUY, U., and KSHEMKALYANI, A., "Clock synchronization for wireless sensor networks: a survey," *Ad Hoc Networks (Elsevier)*, vol. 3, pp. 281–323, May 2005.
[42]	KAWADIA, V. and KUMAR, P., "Principles and protocols for power control in ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 76–88, January 2005.
[43]	GOMEZ, J. and CAMPBELL, A. T., "A case for variable-range transmission power control in wireless multihop networks," in *Proceedings of IEEE Conf. on Computer Communications (INFOCOM)*, (Hong Kong S.A.R., P.R. China), March 2004.
[44]	FINN, G., "Routing and addressing problems in large metropolitan-scale internetworks, "tech. rep., ISI res. rep ISU/RR- 87-180, March 1987.
[45]	F. KUHN, R. W. and ZOLLINGER, A., "Worst-Case Optimal and Average-Case Efficient Geometric Ad-Hoc Routing," in *Proceedings of ACM Intl. Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc)*, (Annapolis,MD, USA), June 2003.
[46]	BOSE, P., MORIN, P., STOJMENOVIC, I., and URRUTIA, J., "Routing with guaranteed delivery in ad hoc wireless networks," *ACM Wireless Networks*, vol. 7, pp. 609–616, November 2001.
[47]	NA, J. and KWON KIM, C., "GLR: A Novel Geographic Routing Scheme for Large Wireless Ad-hoc Networks," *Computer Networks (Elsevier)*, vol. 50, pp. 3434–3448, December 2006.
[48]	E. C.-H. Ngai, Y. Zhou, M. R. Lyu, and J. Liu. A delayaware reliable event reporting framework for wireless sensor actuator networks. CSE Technical Report CS-TR-2006-04, The Chinese University of Hong Kong, Mar 2006.
[49]	J. G. McNeff. The global positioning system. *IEEE Transactions on Microwave Theory and Techniques*, 50:645–652, Mar 2002.
[50]	L. Hu and D. Evans. Localization for mobile sensor networks. In *Proc. of ACM MobiCom*, pages 99–110, Philadelphia, PA, U.S., 26 Sep - 1 Oct 2004.
[51]	A. Savvides, C. C. Han, and M. B. Srivastava. Dynamic fine-grained location in ad hoc networks of sensors. In *Proc. of ACM MobiCom*, pages 166–179, Philadelphia, PA,

	U.S., 2001.
[52]	T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale sensor networks. In *Proc. of ACM MobiCom*, pages 81–95, SanDiego, CA, U.S., 2003.
[53]	B. Krishnamachari, D. Estrin, and S.Wicker. Modelling datacentric routing in wireless sensor networks. In *Proc. Of IEEE Infocom*, 2002.
[54]	Maxim Batalin and Gaurav S. Sukhatme, "Spreading Out: A Local Approach to Multi-robot Coverage," in the Proceedings of the International Symposium on Distributed Autonomous Robotic Systems, pp. 373-382, Fukuoka, Japan, Jun 2002.
[55]	I. F. Akyildiz and I. H. Kasimoglu, "Wireless Sensor and actor networks: Research Challenges," Elsevier Ad hoc Network Journal, Vol. 2, pp. 351-367, 2004.
[56]	F.Bouhafs, M. Merabti, and H. Mokhtar, A Coordination Protocol for Wireless Sensor and Actor Networks, in The seven annual postgraduate symposium on the convergence of telecommunications, Computing and Mathematical Sciences School, Liverpool John Moores University, Liverpool, UK, 26- 27 June 2006
[57]	F. Aurenhammer and R. Klein, "Voronoi Diagrams, Chapter V", in Handbook of Computational Geometry, Urrutia J.S.a.G., Editor. 2000, Elsevier Science Publishing. p. 201-290.
[58]	F. Bouhafs, M. Merabti, and H. Mokhtar. "A Semantic Clustering Routing Protocol for Wireless Sensor Networks", in IEEE Consumer Communications and Networking Conference, Las Vegas, NV, USA, 8-10 January, 2006
[59]	F. Bouhafs, M. Merabti, and H. Mokhtar. "A Semantic Clustering Scheme For Wireless Sensor Networks", in The sixth annual postgraduate symposium on the convergence of telecommunications, networking and broadcasting, Liverpool John Moores University, 27-28 June 2005
[60]	K. Akkaya and M. Younis, "COLA: A Coverage and Latency aware Actor Placement for Wireless Sensor and Actor Networks", in the Proceedings of IEEE Vehicular Technology Conference (VTC), Montreal, CA, Sept. 2006.
[61]	A. Efrat, S. Har-Peled, J. S. B. Mitchell, "Approximation Algorithms for Two Optimal Location Problems in Sensor Networks" in the *Proceedings of the 3rd International Conference on Broadband Communications, Networks and Systems (Broadnets 2005)*, Boston, Massachusetts , October 2005.
[62]	Maxim Batalin, Gaurav Sukhatme, "The Analysis of an Efficient Algorithm for Robot Coverage and Exploration based on Sensor Network Deployment ", in the *Proceedings of IEEE International Conference on Robotics and Automation*, Barcelona, Spain, Apr 2005.
[63]	G. Gupta, M. Younis, "Load-Balanced Clustering in Wireless Sensor Networks," in the *Proceedings of the IEEE International Conference on Communication (ICC 2003)*, Anchorage, Alaska, May 2003.
[64]	D.B. Shmoys, ' E. Tardos, and K. Aardal, "Approximation algorithms for facility location problems," in the *Proceedings of the 29th ACM Symposium on Theory of Computing*, pp. 265--274, El Paso, TX, May 1997.
[65]	McLaughlan,Brian;Akkaya,Kemal Pervasive Services, "Coverage-based Clustering of Wireless Sensor and Actor Networks", IEEE International Conference on Volume , Issue , 15-20 July 2007.
[66]	A. Efrat, S. Har-Peled, J. S. B. Mitchell, "Approximation Algorithms for Two Optimal Location Problems in Sensor Networks" in the *Proceedings of the Broadnets 2005*, Boston, MA , October 2005.
[67]	M. R. Garey and D. S. Johnson, "Computers and Intractability: A guide to the theory of

	NP-completeness," *Freeman*, San Francisco, 1978.
[68]	P.Alimonti and T. Calamoneri, "Improved Approximations of Independent Dominating Set in Bounded Degree Graphs," in the *Proceedings of Int. Workshop on Graph-Theoretic Concepts in Computer Science*, Como, June 1996. Springer-Verlag.
[69]	Alan D. Amis et al., "Max-Min D-Cluster Formation in Wireless Ad Hoc Networks," in the *Proceedings of IEEE INFOCOM, March 2000.*
[70]	F. Garcia, J. Solano, I. Stojmenovic, Connectivity based *k*-hop clustering in wireless networks, *Telecommunication Systems*, 22, 1-4, 2003, 205-220.
[71]	ruiz-ibarra, villasenor-gonzalez, Wireless Sensor And Actor Networks II ;in IFIP international federation for information processing, volume 264, ali miri ; (boston ; springer), pp, 2008 ,62-73.
[72]	Farinelli, A., Iocchi, L., Nardi, D.: multirobot systems: a classification focused on coordination. In: systems, Man and Cybernetics, Part B, IEEE Transactions on. Volume 34. (2004) 2015–202
[73]	As'ad Salkham, Raymond Cunningham, Aline Senart, and Vinny Cahill, "A Taxonomy of Collaborative Context-Aware Systems," in Proceedings of the Workshop on Ubiquitous Mobile Information and Collaboration Systems (UMICS '06), CAiSE '06, pages 899-911, Luxembourg, June 2006.
[74]	Sundresh, Sameer, Gul Agha, Kirill Mechitov, WooYoung Kim, YoungMin Kwon. "Coordination Services for Wireless Sensor Networks," International Workshop on Advanced Sensors, Structural Health Monitoring and Smart Structures, 2003
[75]	A. Sobeih, W. Chen, J. Hou, L. Kung, N. Li, H. Lim, H. Tyan, and H. Zhang, J-Sim: A Simulation and Emulation Environment for Wireless Sensor Networks, In the proceedings of IEEE wireless communications, 2006.