Un protocole de session dans les réseaux de capteurs sans fils

Said Harchi

UNIVERSITE DE LORRAINE

Ecole Doctorale : IAEM Lorraine

CRAN (Centre de Recherche en Automatique de Nancy)

Thèse
pour obtenir le diplôme de
DOCTEUR DE L'UNIVERSITÉ DE LORRAINE

En Automatique, Traitement du Signal et des Images, Génie Informatique

présentée et soutenue par

Said HARCHI

le 06 décembre 2013

Un protocole de session dans les réseaux de capteurs sans fils

Jury :

Rapporteurs :

M. Thierry GAYRAUD	Professeur, Université Paul Sabatier Toulouse III, LAAS
M. François SPIES	Professeur, Université de Franche-Comté, LIFC

Examinateurs

M. Khalid BENZAKOUR	Docteur, Institut supérieur du Génie Appliqué, Casablanca
M. Alexandre GUITTON	MCF, Université Blaise Pascal Clermont-Ferrand, LIMOS
M. Benoît IUNG	Professeur, Université de Lorraine CRAN
M. Thiery DIVOUX	Professeur, Université de Lorraine CRAN, (directeur de thèse)
M. Jean-Philippe GEORGES	MdC, Université de Lorraine (co-directeur de thèse)

A tous ceux que j'aime

Remerciements

En premier lieu, je tiens à remercier mon directeur de thèse **M. Thierry DIVOUX**, Professeur à l'Université de Loraine, pour m'avoir fait confiance et m'avoir donné l'opportunité d'effectuer ces travaux de recherches au Centre de Recherche en Automatique de Nancy. Il m'a encouragé, m'a conseillé et m'a guidé tout au long de la préparation de cette thèse, tout en me laissant une grande liberté. J'espère avoir été à la hauteur de ses attentes.

Je remercie également **M. Jean-Philippe GEORGES**, co-encadrant de ma thèse et Maître de conférences à l'Université de Lorraine, pour sa patience, sa gentillesse, sa qualité d'encadrement et les nombreuses discussions que nous avons pu avoir ensemble.

Je tiens aussi à remercier **M. François SPIES**, Professeur à l'université de Franche-Comté, et **M. Thierry GAYRAUD**, Professeur à l'Université Paul Sabatier Toulouse III, pour avoir accepté de rapporter sur ma thèse et pour l'intérêt qu'ils ont manifesté pour mon travail.

J'adresse mes remerciements à **M. Alexandre GUITTON**, Maître de conférences à l'Université Blaise Pascal Clermont Ferrand, pour avoir accepté d'examiner mon travail de thèse et pour les discussions intéressantes que j'ai eues avec lui.

Je remercie **M. Khalid BENZAKOUR**, pour ses encouragements et pour avoir accepté de faire partie de mon jury de thèse.

Mes remerciements iront aussi à tous les membres de la famille **DIOURI** pour leur soutien moral et financier.

Egalement, je tiens à remercier tous les membres du groupe thématique Systèmes de Production Ambiants pour la convivialité qui a régné tout au long de ma thèse.

Enfin, je tiens à exprimer ma gratitude et ma reconnaissance à mes parents pour leur soutien et leurs encouragements et cela depuis mon plus jeune âge. A ma femme et mes petits anges: **Mohamed Amine**, **Chihab Eddine** et **Rayane**, qui m'ont manqué durant mes séjours à Nancy, pour les moments agréables que je passe avec eux et pour leurs encouragements durant la préparation de ma thèse.

Sommaire

Chapitre 1 *Introduction*

Dans ce chapitre nous introduisons le contexte de notre étude et les problématiques traitées dans cette thèse. Un descriptif de l'organisation de ce manuscrit clôt ce chapitre.

1.1 Introduction

Initié suite à une recommandation adoptée à la Conférence des Nations Unies sur l'environnement, le CDI (*Centre de Documentation et d'Information*) et l'OMM (*Organisation Métrologique Mondiale*) ont lancé un programme de surveillance continue de la pollution des mers par les hydrocarbures (Stockholm, 1972). Ce programme entre dans le cadre du Système Mondial Intégré de Stations Océaniques (SMISO). Les résultats obtenus sont décrits dans le rapport intitulé "Global Oil Pollution : "*results of MAPMOPP, the IGOSS*[1] *Pilot Project on Marine Pollution (Petroleum) Monitoring*" (COI, Paris, 1981). Cette application de surveillance continue de la pollution des mers par les hydrocarbures peut exploiter la technologie des réseaux de capteurs sans fil. En effet, la dérive des nappes de pétrole sur la surface de l'eau de mer, leur consistance et leur étendue pourraient être surveillées de manière continue et en temps réel par un ensemble de capteurs sans fil déployés sur ces nappes.

Un tel système peut apporter des avantages (suivi continu et en temps réel) par rapport aux systèmes de surveillance de l'environnement naturel actuels tel le survol aérien par exemple. Nous savons que l'observation des marées noires à partir des plates-formes aéroportées est meilleure que celle effectuée à partir des navires. En effet, l'observateur dispose d'une vue plus étendue qui lui permet de distinguer mieux les différences de couleur, de reflets et de texture de la surface de la mer. C'est vrai que l'étendue, la forme et la couleur indiquent parfois de manière évidente la présence de pétrole. Mais, il y a risque de confondre ces nappes d'hydrocarbures avec des pellicules naturelles ou des zones de calme, en particulier aux altitudes très basses. Nous proposons donc l'utilisation des réseaux de capteurs sans fil pour assurer un monitoring continu fiable de l'évolution des nappes de pétrole en milieu marin.

1.2 Contexte de l'étude, problématiques traitées

Dès le début de la marée noire, on disperse, aléatoirement, (N) nœuds de capteurs sans fil communicants pour récupérer un ensemble d'informations qui aideront à observer l'évolution des nappes de pétrole dans le temps.

Cependant, utiliser un réseau de capteurs sans fil afin d'observer l'évolution des nappes de pétrole présente des défis à relever pour pouvoir bâtir un système d'information cohérant basé sur ce type de réseau. Le défi majeur à relever pour pouvoir utiliser un réseau de capteurs sans fil dans notre application de surveillance est de faire face à la dynamique du système faisant varier la topologie du réseau. La dynamique du réseau est causée par deux types de mobilité qui sont :

La mobilité des nappes de pétrole qui risque d'entrainer un changement topologique du réseau pouvant créer une rupture de communication qui peut engendrer l'inaccessibilité du réseau. En effet, une fois déversé, le pétrole a immédiatement tendance à s'étaler à la surface de l'eau et former des nappes. En théorie, ces nappes de pétrole forment un cercle régulier dont le diamètre augmente huit

[1] *IGOSS : Integrated Global Ocean Services System*

fois dans l'heure qui suit le déversement, puis cinq fois avant la fin de la semaine. Ensuite, le vent, les vagues et la force de force de Coriolis (liée à la rotation de la Terre) contribuent à l'étirement, au déplacement et à la fragmentation de la nappe jusqu'à sa décomposition en gouttelettes. De ce fait, il est bien évident que le devenir des nappes à moyen et long terme est actuellement très difficile à prévoir. (Voir sites[2])

***La mobilité des nœuds déployés* aléatoirement sur les nappes de pétrole**. En effet, sous l'effet de facteurs externes comme le vent, les nœuds peuvent se déplacer de façon aléatoire ce qui peut affecter les liaisons radio lorsque les nœuds sont éloignés les uns des autres. De même, l'effet des facteurs endogènes comme l'épuisement des batteries ou la défaillance des nœuds peut influencer la topologie du réseau et causer des problèmes d'acheminement des informations.

La dynamique du réseau peut conduire à une perte de connectivité partielle ou totale pour un ou plusieurs nœuds conduisant à une incohérence du système d'information global. Ainsi, nous nous trouvons confrontés à un problème de connectivité dû à la dynamique de la topologie du réseau (graphe non connexe) dont nous ignorons le degré de disconnectivité.

Conscients que la réalisation des protocoles permettant l'exploitation des réseaux de capteurs sans fil dynamique n'est pas triviale, nous proposons, dans le respect des contraintes d'une telle infrastructure, une architecture d'un système d'information cohérent de collecte, de suivi, d'analyse, de traitement et de représentation des données relatives à la propagation de nappes de pétrole en mer.

La Figure 1-1 illustre le système proposé pour la mise en place de l'application visée. En premier lieu, nous proposons de partitionner logiquement le réseau dynamique en groupes distincts. Chaque groupe de nœuds capteurs est représenté par un nœud chef de groupe. Le choix d'utiliser une architecture partitionnée au lieu d'une architecture plate nous permettra d'avoir moins de nœuds capteurs à visiter par le collecteur mobile que nous utilisons pour la collecte d'informations. Ce qui réduit le temps du parcours de visite du collecteur et par conséquent augmente la fréquence de ses visites aux chefs de groupes.

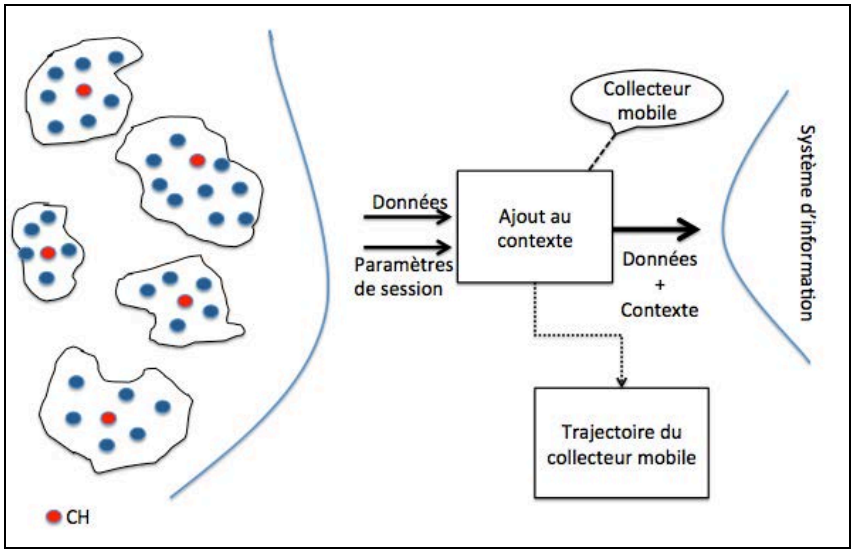


Figure 1-1 : Schéma illustratif du contexte d'étude.

[2] (http://www.labo-analytika.com/html/contribution_aircom2.html
http://envlit.ifremer.fr/surveillance/pollutions_accidentelles/prestige/fr/hydrodynamique)
http://wwz.ifremer.fr/envlit/surveillance/pollutions_accidentelles/prestige/fr/hydrodynamique

En deuxième lieu, sachant que les groupes sont mouvants, en taille, en nombre, et en représentation (le chef de groupe peut avoir changé) et que nous ne pouvons donc pas avoir un continuum de suivi de l'évolution des nappes, nous proposons de mettre en œuvre une couche abstraite dite de « session » dont l'objectif est de recontextualiser chaque retour du collecteur par rapport à sa visite précédente. La trajectoire du collecteur sera définie en fonction des paramètres de session proposés.

Les travaux présentés dans cette thèse traitent donc, des réseaux de capteurs sans fil dynamiques, de clustering dans ce type de réseaux, de collecte d'informations en utilisant un collecteur mobile, et de la mise en place d'un protocole de session permettant de synchroniser deux dynamiques : la dynamique du réseau physique et la dynamique du réseau de collecte. La synchronisation permettra à l'utilisateur final (système d'information) de reconstruire le contexte de l'évolution des nappes de pétroles formées.

1.3 Organisation de la thèse

Cette thèse est organisée en cinq chapitres.

Le chapitre 1 est la présente introduction, dans laquelle sont décrits la problématique soulevée et les objectifs de notre travail.

Le chapitre 2 présente une description d'un réseau de capteurs sans fil et ses caractéristiques spécifiques liées soit aux capteurs qui le constituent comme l'énergie, la portée de transmission, …etc. soit liées au réseau lui-même comme la bande passante, la densité, ...etc. Nous clôturons ce chapitre par la présentation de l'architecture de notre système de surveillance des nappes de pétrole en milieu marin. Cette architecture est choisie après une brève bibliographie des différentes architectures utilisées dans les projets phares exploitant les réseaux de capteurs sans fil dans le domaine de la surveillance de l'environnement.

Le chapitre 3 présente notre première contribution dans cette thèse. Il s'agit de la proposition d'un algorithme de partitionnement basé sur une technique d'élection utilisant une métrique, adaptée à notre application. Cette métrique est composée de la densité des nœuds sur le réseau, de leur mobilité et de leur énergie résiduelle. Ces paramètres sont pondérés par des coefficients dont les valeurs peuvent être adaptées selon le type et le besoin de l'application. Plusieurs travaux présentant les différentes approches de clustering proposées pour les réseaux de capteurs sans fils en général et plus particulièrement pour les réseaux de capteurs sans fil dynamiques sont décrits dans la partie état d'art de ce chapitre. Ces travaux ont utilisé des métriques sous forme de combinaison de plusieurs paramètres tels : le degré, la densité, la puissance de transmission du nœud, la mobilité, la taille désirée du cluster, …etc. Cependant, l'influence de la dynamique sur le processus de clustering n'est que peu ou pas du tout traitée dans ces travaux. A cet effet, notre deuxième proposition consiste en un modèle de nœud capteur développé sous l'environnement de simulation réseau Opnet Modeler. Nous pouvons alors mettre en évidence l'influence de la dynamique du réseau sur le processus de clustering, et définir également les coefficients idéaux de pondération appliqués à notre application de surveillance des nappes de pétrole en milieu marin. Nous clôturons ce chapitre en présentant des évaluations des performances de cet algorithme et l'influence de la dynamique du réseau sur le processus de clustering.

Le chapitre 4 présente notre troisième contribution à savoir un algorithme de session sous deux versions centralisée et décentralisée. La version centralisée, implémentée au niveau de notre nœud photographe, permet de donner des vues (photos) à des instants donnés des nappes de pétrole dispersées en mer. La version décentralisée intègre le protocole de communication entre le collecteur mobile implémentant notre algorithme de session, et le chef de cluster visité.

Après avoir défini la notion de session, les attributs de session et les indicateurs de session que nous nous utiliserons dans la mise en place de notre protocole, nous décrivons les outils de détection d'invariance, de fusion et de scission des nappes de pétrole sur lesquels repose l'algorithme. Une évaluation de l'algorithme par simulation sous Opnet ainsi qu'une discussion termine cette partie. Nous clôturons ce chapitre par une conclusion qui résumera les résultats obtenus et leur évaluation.

Finalement nous concluons cette thèse par un résumé de nos contributions, nous portons un jugement critique sur le travail accompli, et présentons les différentes perspectives et directions des futures recherches.

Chapitre 2 Architectures de collecte des données dans les réseaux de capteurs sans fil

Dans ce chapitre, nous présentons les réseaux de capteurs sans fil en général : leurs caractéristiques, types d'applications pour lesquelles ils sont dédiés,…, et quelques projets phares ayant utilisés les réseaux de capteurs sans fil afin de mieux choisir une architecture de collecte de données pour notre système de monitoring des nappes de pétrole en milieu marin.

Ce chapitre est organisé comme suit :

- *la partie 2.2 décrit un réseau de capteurs sans fil, et la partie 2.3 quelques caractéristiques spécifiques à ces réseaux.*
- *La section 2.4 présente les différents types d'applications dédiées aux réseaux de capteurs sans fil.*
- *La section 2.5 présente un état d'art sur les architectures de collecte de données proposées pour les réseaux de capteurs sans fil utilisant un puits ou/et collecteurs fixes ou mobiles à travers une présentation des projets phares exploitant les réseaux de capteurs sans fil dans le domaine de la surveillance de l'environnement. Une conclusion de cette partie nous permet de nous positionner et de choisir une architecture de collecte des données pour notre système de monitoring des nappes de pétrole en milieu marin.*
- *Cette architecture retenue sera présentée dans la section 2.6.*

2.1 Introduction

Les réseaux de capteurs sans fil (RCSF ou WSN Wireless Sensor Network) sont devenus de plus en plus populaires grâce aux grandes avancées dans le domaine de la microélectronique qui ont réduit le coût de fabrication des nœuds de capteurs d'une manière considérable. Ces réseaux ont montré leur efficacité dans le suivi et le contrôle à distance de l'environnement physique avec une meilleure précision. Actuellement ils sont exploités pour différents domaines d'applications tels l'observation de l'environnement naturel (pollution, inondation,…etc.), le suivi d'écosystèmes (surveillance de niches d'oiseaux, croissance des plantes,…etc.), le contrôle militaire (télésurveillance de champs de bataille, détection d'ennemis,…etc.), l'analyse biomédicale et la surveillance médicale (détection de cancer, rétine artificielle, taux de glucose, diabètes,…etc.).

Cependant la mise en place d'un tel système pose de nombreux problèmes parmi lesquels le problème du maintien de la connectivité du réseau, la collecte des données ou encore le routage de l'information vers les stations de traitement de l'information. D'autres contraintes intrinsèques apparaissent au niveau des nœuds capteurs et au niveau du réseau formé par ces nœuds capteurs.

Notre objectif dans ce chapitre est de donner une vue générale sur les réseaux de capteurs, leurs caractéristiques spécifiques, leurs types d'applications et les différentes architectures de collecte de données déjà exploitées dans des projets célèbres. Pour finir nous présentons l'architecture de collecte de données retenue pour notre système de suivi des nappes de pétrole en mer.

2.2 Description d'un réseau de capteurs sans fil

Un réseau de capteurs sans fil est un ensemble de nœuds capteurs communicant en liaison sans fil. Ces nœuds capteurs disposent d'une unité de stockage, d'une unité de traitement de données, d'unités de captage diverses, d'une batterie d'alimentation et d'une interface de communication radio. Ces nœuds capteurs sont capables de s'auto-organiser en sous-réseaux distribués pour collecter les informations du

monde physique sur lequel ils sont déployés. Les données collectées sont ensuite transmises à une station de traitement de données appelée "station de base" (BS : *Base Station*). Ce transfert des données collectées, de manière périodique ou évènementielle, à la station de base peut être fait directement ou à travers un ou plusieurs nœuds spéciaux fixes ou mobiles appelés "puits" (*Sink*).

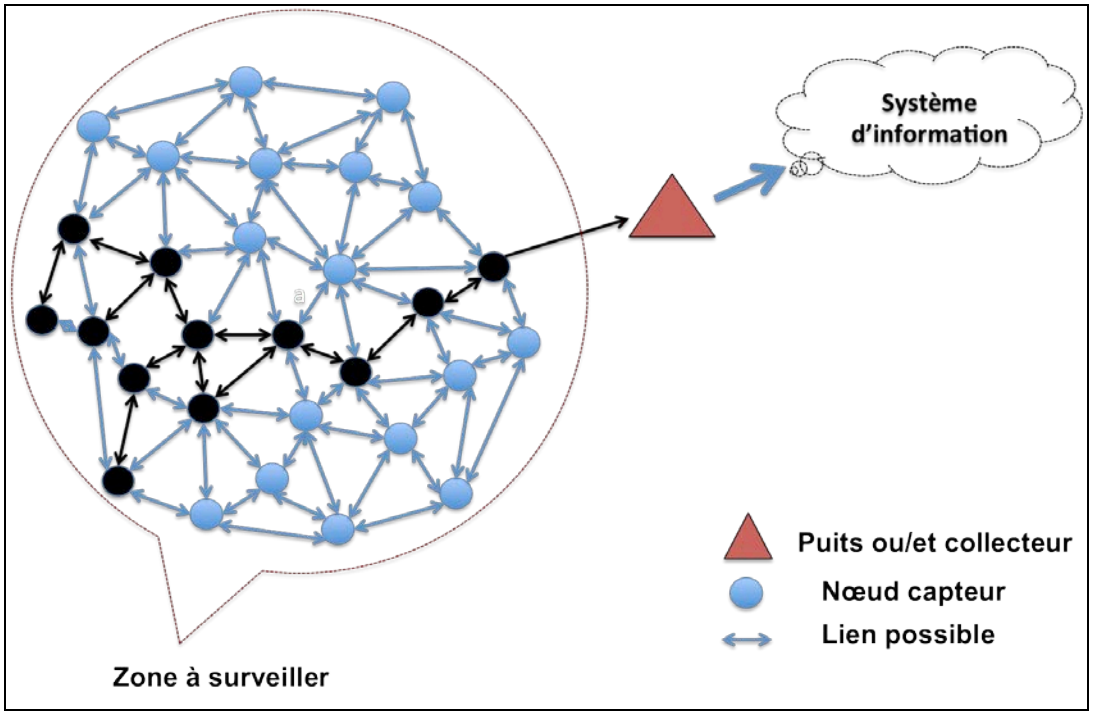


Figure 2-1 : Exemple de réseau de capteurs sans fil

La Figure 2-1 présente un exemple classique de réseau de capteurs sans fil : les capteurs sont déployés de manière aléatoire dans une zone à surveiller, et un puits ou/et collecteur situé à l'extrémité de cette zone, est chargé de récupérer les données collectées par les capteurs. Lorsqu'un capteur détecte un événement pertinent, un message d'alerte est envoyé au puits/ collecteur par le biais d'une communication multi-sauts. Les données collectées sont traitées et analysées par un système d'information.

2.3 Caractéristiques spécifiques des réseaux de capteurs sans fil

Les réseaux de capteurs présentent des caractéristiques intrinsèques au niveau des nœuds capteurs (énergie, portée de transmission et puissance de stockage et de traitement) et au niveau du réseau formé par ces nœuds capteurs (Bande passante, densité, déploiement, type de réseau et topologie dynamique)**.**

2.3.1 Caractéristiques liées aux nœuds capteurs

Les nœuds capteurs permettent de capter et de transmettre des données sur plusieurs types de paramètres du monde physique sur lequel ils sont déployés. Ces paramètres peuvent être la pression, la viscosité, la vibration, la température, la vitesse et la position des objets surveillés.

Ces objets miniaturisés présentent des propriétés propres que les capteurs habituels n'ont pas. Nous ne citons dans ce qui suit que les plus importantes :

2.3.1.1 Energie

L'énergie représente une contrainte forte dans la majorité des applications à base de réseaux de capteurs sans fil : Chaque nœud capteur fonctionne avec une batterie, généralement, non rechargeable et à capacité limitée étant donné sa petite taille. Dans la majorité des cas, ces nœuds capteurs sont déployés dans des zones hostiles ou difficiles d'accès et il est très peu probable qu'ils soient récupérables. Aussi, vu leur nombre très grand (des milliers) on ne peut pas imaginer (faute de temps et coût) de s'occuper de chaque nœud capteur un à un. Vu ce constat (impossibilité de remplacer un nœud hors service pour problème d'énergie ou problème de fonctionnement), tout programme dédié à fonctionner sur un nœud capteur doit prendre en considération ce problème de gestion d'énergie. Surtout en temps de transmission du fait qu'une communication sans fil est une des opérations les plus coûteuses parmi celles dont est chargé le nœud capteur.

2.3.1.2 Portée de transmission

La portée de transmission est limitée par la capacité de rayonnement des antennes utilisées et la puissance du signal mises en jeu. Par exemple, la communication entre deux nœuds capteurs ne peut avoir lieu que si la distance qui les sépare n'est pas trop importante (quelques dizaines de mètres en pratique). Et que plus cette distance est grande, plus le coût énergétique est élevé. C'est d'autant plus vrai s'il y a des obstacles physiques entre les deux nœuds qui risquent d'empêcher toute communication.

2.3.1.3 Puissance de stockage et de traitement

La puissance de stockage et de traitement est relativement faible. Par exemple, les nœuds capteurs de type "*mote*" sont composés d'un microcontrôleur 8-bits 4MHz, 40Koctets de mémoire et d'une radio d'un débit d'environ 10 kbps. Cela reste vrai même pour les nœuds de moyenne gamme, comme les "*UCLA/ROCKWELL'S WINS*", qui ont un processeur *StrongARM* 1100 avec une mémoire flash de 1 Moctets, une mémoire RAM de 128 Koctets et une radio dont le débit est 100 Kbps.

2.3.2 Caractéristiques liées au réseau

En plus des caractéristiques intrinsèques des nœuds capteurs, il existe d'autres caractéristiques qui sont elles liées au réseau telles la communication inter-nœuds, la bande passante, la densité, le déploiement, le type de réseau et la topologie dynamique.

2.3.2.1 Communication inter-nœuds

Les communications inter-nœuds capteurs ne sont pas fiables et l'exécution des processus implantés dans les nœuds est asynchrone ce qui rend la conception d'un système distribué très difficile à réaliser.

2.3.2.2 Bande passante

La bande passante utilisée dans les réseaux de capteurs sans fil est généralement très réduite. A titre d'exemple, le kit de nœuds capteurs sans fil de *Crossbow Technology*3 a été programmé pour fonctionner dans les bandes 900 et 2400 MHz. Cette propriété exige que les échanges des messages protocolaires destinés au bon fonctionnement des protocoles de communication (overhead) soient limités au maximum.

[3] *(http://www.xbow.com)*

2.3.2.3 Densité

Selon (Bulusu et al., 2001), la densité des nœuds déployés dans une région A est définie, par la relation suivante :

$$\mu(R) = (N * \pi R^2)/(A)$$

$$Avec \begin{cases} N : \text{nombre de nœuds éparpillés dans la région d'aire A} \\ R : \text{portée de la transmission radio d'un nœud} \end{cases}$$

Donc, $\mu(R)$ donne le nombre de nœuds se trouvant dans le domaine de transmission *R* d'un nœud donné dans la région *A*.

Généralement, le type de l'application envisagée influe sur le choix de la densité à prendre. Et pour obtenir des résultats de mesure de bonne qualité, on privilégie souvent le facteur nombre de nœuds plutôt que le facteur qualité des nœuds capteurs. On cherche donc à avoir des réseaux de nœuds capteurs denses. Ainsi, une application de diagnostic de machines nécessite par exemple, une densité proche de 300 nœuds par région de $25m^2$, tandis que la densité nécessaire pour le contrôle des véhicules ne peut pas dépasser 10 capteurs pour une région de même taille. (Akyildiz et al., 2002)

Notons qu'une forte densité de nœuds capteurs exclut l'isolement de nœuds et favorise l'utilisation des communications multi-sauts qui consomment moins d'énergie que les communications traditionnelles à un seul saut. Néanmoins elle engendre des problèmes de communication non négligeables (congestion).

2.3.2.4 Déploiement

Les nœuds capteurs peuvent être déployés d'une manière aléatoire ou déterministe.

- méthode aléatoire où les nœuds sont éparpillés (largués par des avions par exemple) sur la zone à surveiller.
- méthode déterministe où les nœuds sont placés de manière individuelle à des endroits bien précis dans la zone de captage.

Le nombre important de nœuds utilisés dans un réseau de capteurs, qui peut atteindre jusqu'à un million (Kone, 2011), empêche souvent leur déploiement suivant un plan initialement établi. Néanmoins, un schéma général pour le déploiement initial doit être conçu pour permettre de réduire les coûts d'installation, d'augmenter la flexibilité d'arrangement des nœuds et faciliter l'auto-organisation des nœuds et leur tolérance aux pannes.

2.3.2.5 Type de réseau

Le type de réseau influe sur le choix du type des protocoles et services à mettre en place. Ainsi, pour un réseau homogène où tous les nœuds disposent de mêmes ressources, il n'est pas judicieux d'utiliser un protocole de type centralisé. Du fait qu'il y a risque d'épuisement de la réserve énergétique du nœud sur lequel tourne ce protocole. Tandis que ce scénario est fort possible dans un réseau hétérogène où il existe des nœuds riches en ressources pouvant assurer une telle fonction (comme le routage par exemple).

2.3.2.6 Topologie dynamique

La dynamique de la topologie est l'un des aspects les plus originaux des réseaux de capteurs sans fil. Car le changement topologique du réseau peut engendrer des problèmes de connectivité du réseau et

donc son inaccessibilité. En effet, la dynamique de la topologie du réseau peut conduire à une redéfinition fréquente des capacités de communication. Cet impact peut alors entraîner une perte de connectivité partielle ou complète pour un ou plusieurs capteurs. La dynamique de la topologie peut être due soit à :

- la mobilité des nœuds : les nœuds capteurs sont mobiles (cas des nœuds capteurs robots) ou ils sont fixés sur des entités mobiles ou encore lorsque les nœuds capteurs bougent sous l'effet de facteurs externes tel que le vent par exemple;
- l'épuisement de la réserve énergétique du nœud : à cause de l'autonomie énergétique limitée des nœuds ou même à cause d'une défaillance, ces derniers peuvent devenir hors service donc du point de vue logique supprimés et ainsi changent la topologie du réseau.
- l'ajout de nouveaux nœuds : en effet, l'ajout de nouveaux nœuds au réseau peut aussi remettre en cause la topologie du réseau. Il suffit juste que le nœud ajouté soit dans la zone de couverture d'au moins un nœud capteur du réseau pour qu'il modifie la topologie du réseau.

De même, la donnée captée peut transiter par plusieurs nœuds capteurs avant d'arriver à la station de traitement. Donc à chaque saut, il y a une probabilité d'avoir des erreurs de transmission dans le cas où le nœud capteur cible ou intermédiaire est hors service.

Le problème de discontinuité de connectivité, due au changement topologique du réseau, est souvent ressenti dans les applications de type environnemental à l'instar de l'application considérée dans notre cas d'étude qui vise l'observation de la dérive des nappes de pétrole en milieu marin en utilisant un réseau de capteurs sans fil.

2.4 Applications dédiées

La miniaturisation des capteurs et leur coût, le développement d'une large gamme de types de capteurs de mesure et l'évolution des médias de communication sans fil ont donné un grand élan à l'utilisation des réseaux de capteurs dans divers domaines.

Les principaux domaines d'application sont catégorisés selon le type d'informations mesurées ou transportées par le réseau. Parmi les applications les plus utilisées nous trouvons, le contrôle environnemental (Werner-Allen et al., 2006), les services médicaux (Doan et Wood, 2005), le suivi de la faune (Guo et al., 2006), le transport (Tran et al., 2011), la domotique (CRUISE, 2006) et les applications industrielles (e-SENSE, 2006) ...etc.

Nous pouvons classifier les applications dans trois catégories: applications à détection d'événements (*orientées événements*) (*Event driven),* applications à prélèvement périodique d'informations (orientées requêtes) (*Time driven*) et les applications hybrides. Ces types d'applications sont caractérisées par des exigences d'utilisation particulières et imposent des caractéristiques spécifiques au réseau de capteurs sans fil.

2.4.1 Orientées évènements (event driven)

Comme exemple de ce type d'applications, nous pouvons citer la détection de feu de forêt ou la détection d'un tremblement de terre ou encore la surveillance de la qualité de l'air dans une région donnée.

Dans ce type d'applications, Les nœuds capteurs doivent pouvoir détecter un événement qui pourrait arriver à tout moment dans n'importe quel endroit de la zone à surveiller. Ainsi, les nœuds capteurs doivent collecter les informations de façon continue et réagir immédiatement à des changements brusques des valeurs captées. Aucun contrôle périodique par le puits n'est exigé et le traitement du signal dans les nœuds capteurs est simple : chaque nœud capteur doit comparer la donnée mesurée à

une valeur seuil donnée. Dès dépassement de cette valeur seuil, le nœud capteur envoie la donnée au puits.

En outre, ce genre d'applications exige :

- une forte densité des nœuds capteurs pour que la détection de l'évènement soit assurée avec une bonne probabilité. Ce qui implique que le réseau doit avoir une couverture suffisante.
- une bonne couverture de captage du nœud capteur qui dépend, généralement, du type d'événement à détecter.
- une localisation exacte de la position du nœud capteur dans laquelle l'événement est senti. Notons que des algorithmes de localisation distribués pourraient être utilisés à cette fin et ainsi, l'information peut être reçue par le puits avec géolocalisation.
- une bonne connectivité du réseau, liée à la couverture de transmission, afin que les protocoles de communication permettent à l'alarme envoyée par un nœud capteur d'arriver au puits avec une haute probabilité et dans le plus bref délai.

2.4.2 Orientées requêtes (time driven)

Les applications de ce type visent souvent à évaluer périodiquement un phénomène physique donné. Telle l'application de surveillance de la pression atmosphérique d'une large zone ou les variations de température au sol dans un petit site volcanique. Ce sont des applications qui réagissent par rapport aux périodes. En effet, quand les capteurs reçoivent une requête de service, ils prennent un échantillon de l'environnement et le transmettent moyennant un protocole de communication approprié au puits final. Pour que les échantillons puissent être reçus par le puits avec une probabilité donnée et que l'évolution du phénomène observé soit suivie à la trace, il faut que la fréquence d'échantillonnage soit bien choisie, que la connectivité du réseau soit gardée sous contrôle et les protocoles de communication doivent être conçus pour que l'erreur d'évaluation de processus soit maintenue sous un seuil donné.

Les nœuds capteurs utilisés dans ce type d'applications peuvent se mettre en veille pendant les périodes d'inactivité et n'enclenchent leur dispositif de capture qu'à des instants particuliers (contrainte d'économie d'énergie).

2.4.3 Hybrides

Ce type d'applications résulte des deux précédents types. Comme exemple, nous trouvons, l'application domotique. Où nous pouvons évaluer la température d'une chambre (*orientée événements*) ou/et surveiller un nouveau né. (*time driven*).

2.5 Architectures de collecte des données

Dans cette section, nous commencerons par un état de l'art sur les architectures de collecte des données proposées pour les réseaux de capteurs sans fil utilisant un puits ou/et collecteurs fixes ou mobiles à travers une présentation des projets phares exploitant les réseaux de capteurs sans fil dans le domaine de la surveillance de l'environnement.

L'objectif de notre travail étant l'utilisation des réseaux de capteurs sans fil pour une application environnementale (monitoring des nappes de pétrole en milieu marin), nous allons donc présenter quelques architectures de collecte des données utilisées dans des projets originaux qui s'inscrivent dans ce cadre. Cette présentation de ces plateformes montrera les applications dans lesquelles elles ont été employées, les conditions de mise en œuvre et surtout les différentes techniques de collecte d'information utilisées. Autrement dit, les différents verrous scientifiques adressés.

Ces projets non seulement suscitent notre intérêt pour les applications environnementales mais également nous aident à voir les différents types d'architectures utilisées pour ce type d'application et nous permettent de mieux choisir une architecture bien adaptée à notre système de monitoring des nappes de pétrole.

La littérature scientifique réalisée dans le domaine des réseaux de capteurs sans fil (Francesco et al., 2011), présente une diversité d'architectures de réseaux de capteurs sans fil qui assurent la remontée d'information du système physique sur lequel les nœuds capteurs sont déployés. Ces architectures sont constituées des nœuds de captage de l'information physique, des puits ou collecteurs de données et de la station de traitement de données. Ces architectures varient selon que ces trois éléments sont fixes ou mobiles. Dans ce qui suit nous décrirons les plus fréquemment rencontrées.

2.5.1 Architecture à puits fixe et nœuds capteurs fixes

Dans cette architecture classique (Figure 2-2) les nœuds capteurs et le puits sont fixes. Les données captées sont relayées au puits fixe.

Ce type d'architecture est caractérisé par une forte densité afin d'avoir toujours au moins un chemin entre deux nœuds quelconques du réseau. Cette forme de collecte de données centralisée peut raccourcir la durée de vie de réseau. En effet, il a été démontré dans les travaux ultérieurs (Li et Mohapatra, 2007) (Haenggi, 2004), que l'architecture à puits ou/et collecteur fixe pose le problème de l'épuisement d'énergie des nœuds relais voisins du collecteur fixe. Le relayage de données à un puits fixe peut causer une consommation d'énergie non uniforme sur l'ensemble du réseau. Et on peut avoir une surcharge au niveau des nœuds relais voisins du puits fixe et former ainsi, un goulot de congestion critique.

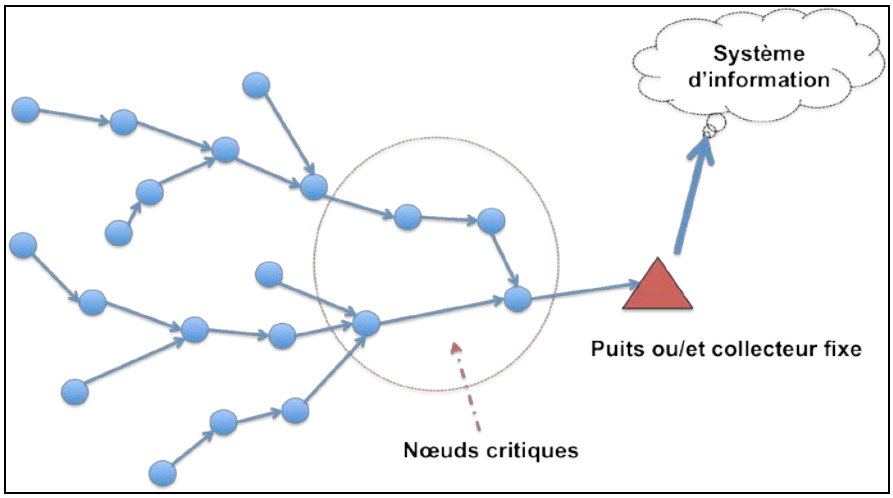


Figure 2-2 : Architecture à puits fixe et nœuds capteurs fixes

Comme type d'application environnementale utilisant ce type d'architecture, nous citons trois projets : le projet *VOLCANO* (Werner-Allen et al., 2006), le projet *GLACSWEB* (Padhy et al., 2005) et le projet *Habitat Monitoring on GDI* (*Great Duck Island*)[4] (Mainwaring et al., 2002):

Projet VOLCANO

Ce projet interdisciplinaire a mis en évidence le potentiel de l'exploitation des réseaux de capteurs dans la compréhension de l'activité volcanique à la surveillance des éruptions des volcans qui sont actifs et dangereux. Ce projet a été mené suite à la collaboration des informaticiens de l'Université

[4] *http://www.greatduckisland.net/*

d'Harvard et des volcanologues de la communauté géophysique de l'Université de North Carolina, de l'Université du New Hampshire et de l'institut géophysique de l'Equateur.

Dans cette application et comme le montre la Figure 2-3, le réseau est constitué de quelques dizaines de nœuds capteurs fixes dotés de microphones et sismomètres collectant les données sismiques et acoustiques sur l'activité volcanique**.**

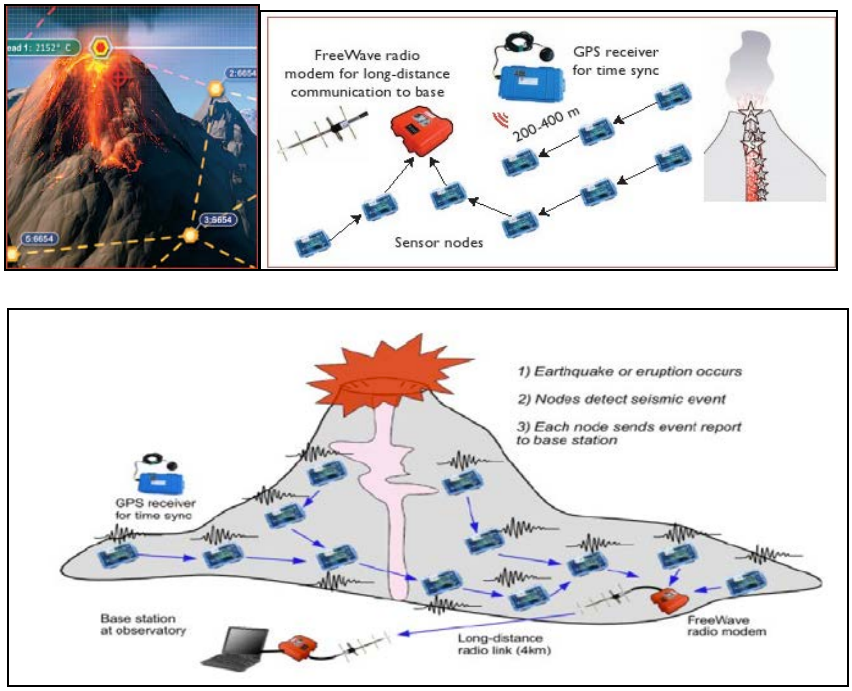


Figure 2-3 : Architecture du réseau de capteurs dédié à la surveillance du volcan (Werner-Allen et al., 2006)

Les nœuds capteurs transmettent les données via un réseau multi-sauts à un nœud passerelle connecté à un modem *FreeWave* de longue distance de communication, fournissant une connectivité radio avec un ordinateur portable à l'observatoire. Un récepteur GPS est utilisé avec un protocole de synchronisation d'horloge visant à établir une base de temps commune pour l'ensemble du réseau.

Le premier déploiement, qui se composait d'une petite gamme de capteurs, a eu lieu en Juillet 2004 au volcan Tungurahua, en Équateur, et a servi de preuve du concept en collectant de manière continue des données infrasonores.

Le second déploiement en Août 2005 au volcan Reventador, en Equateur, se composait de 16 nœuds (*Tmote Sky*) déployés sur 3 km^2 mesurant des signaux sismiques et acoustiques pendant trois semaines. De là, 230 évènements volcaniques ont pu être collectés.

Le troisième déploiement, à Tungurahua en août 2007, était axé sur l'utilité des données collectées par le système développé afin d'améliorer la fidélité des données.

Projet GLACSWEB :

Ce projet (Padhy et al., 2005) représente une collaboration entre les laboratoires d'informatique et de géographie de l'Université de Southampton. Il a pour objectif de développer une technologie permettant d'observer les glaciers en utilisant les réseaux de capteurs sans fil. Ce projet qui a ainsi contribué à la recherche fondamentale en glaciologie et dans le monde des réseaux de capteurs sans fil

a vu son premier déploiement en 2003 à Briksdalsbreen en Norvège.

Ce déploiement comprenait neuf capteurs fixés au glacier à une profondeur allant de 50 à 80m, une station de base placée sur la surface du glacier, une station de référence et un serveur (Figure 2-4). Pour observer le comportement durant la fonte du glacier, les capteurs collectent l'inclinaison, la pression et la température du glacier. Le réseau était capable de fonctionner pendant plusieurs mois.

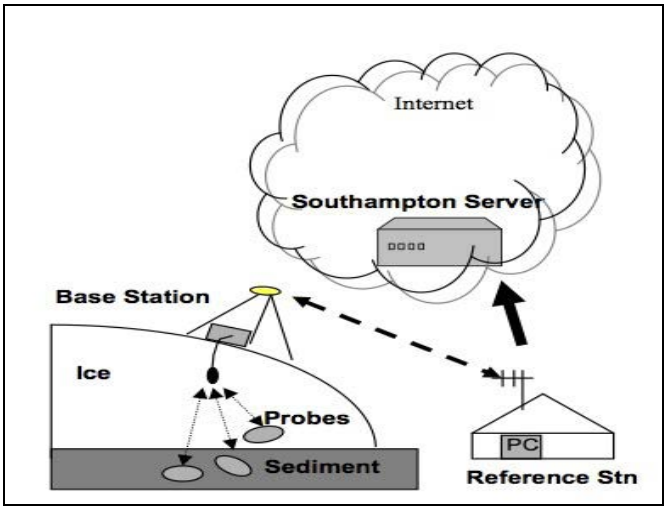


Figure 2-4 : Architecture générale du réseau (Padhy et al., 2005)

Les résultats des données captées montrent que pendant que la pression et la température d'un nœud diminuent tous les sept jours, le degré d'inclinaison reste le même. Les mesures de l'état du glacier et de son mouvement ont été prélevées toutes les quatre heures et celles des conditions climatiques une fois par jour.

Projet : Habitat Monitoring on GDI (*Great Duck Island (USA)*)

Ce projet figure parmi les premiers déploiements expérimentaux de réseaux de capteurs sans fil pour la surveillance de l'habitat. Ce projet s'est déroulé, sur une île du golfe du Maine aux USA appelée « *Great Duck Island* » (Mainwaring et al., 2002). Ce projet a été piloté par une équipe de chercheurs du laboratoire de recherche d'Intel à Berkeley, de l'Université de Californie à Berkeley, et du Collège de l'Atlantique à Bar Harbor.

Ce projet avait pour objectif l'observation du comportement d'oiseaux appelés «*océanites culblanc ou pétrels* » qui vivent en grandes colonies sur l'île pendant la saison de reproduction. Les auteurs s'intéressaient plus particulièrement à mesurer la température des nids en présence ou non des oiseaux adultes, notamment lors de la couvée des œufs. Environ cent capteurs ont été déployés dans et autour des nids à observer.

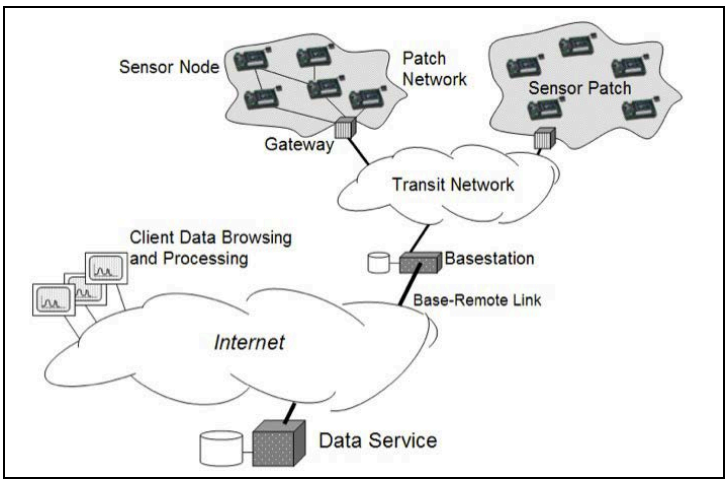


Figure 2-5 : Infrastructure du projet Habitat Monitoring on GDI (Mainwaring et al., 2002)

Le système présenté, comme le montre la Figure 2-5, est basé sur une architecture hiérarchique constituée de trois éléments distincts :

1. les nœuds capteurs fixes, disposés en plusieurs îlots topologiques ;
2. une passerelle fixe par îlot ;
3. une station de base placée sur l'île et reliée à Internet via une connexion satellite.

Les données du microclimat du nid transmises par le réseau de capteurs, ont été mises à disposition sur le web via une station de base sur l'île reliée à un lien de communication par satellite. Ainsi, pour réaliser les mesures, la présence humaine est devenue inutile et donc ne dérange plus les animaux.

2.5.2 Architecture à puits statiques et nœuds capteurs mobiles

Dans ces dernières années, la mobilité des nœuds capteurs a été prise en compte par la communauté scientifique (Chakrabarti et al., 2003) (Kansal et al., 2004) (Ekici et al., 2006) (Gao et al., 2011) (Kondepu et al., 2012). Et une architecture dense n'est plus nécessaire pour assurer une totale couverture de captage de la zone à surveiller et une meilleure connectivité du réseau.

Comme exemple de projets utilisant ce type d'architecture où les nœuds capteurs bougent tandis que le puits est fixe, nous citons le projet *COW* (Handcock et al., 2009) et le projet *CENWITS* (Huang et al., 2005).

Projet CENWITS (Huang et al., 2005)

A travers ce projet, les auteurs ont développé un système d'assistance pour des secouristes dont l'objectif est de simplifier la localisation de randonneurs ou d'alpinistes. Ce système appelé *CenWits* (Figure 2-6) est basé sur l'utilisation de capteurs portés par les randonneurs. Les capteurs enregistrent localement à intervalles de temps réguliers la position GPS de chacun des randonneurs.

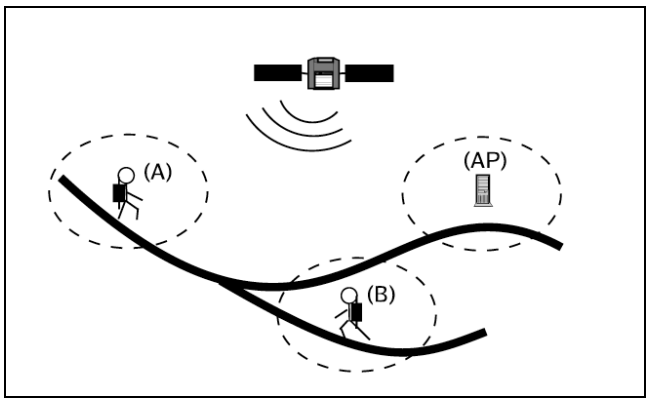


Figure 2-6 : Représentation de l'infrastructure du système Cenwits (Huang et al., 2005)

L'objectif visé par cette collecte est de permettre la constitution d'un historique de points de passage pour chaque randonneur. Lorsque deux randonneurs se rencontrent, ils échangent les données qu'ils ont collectées de façon à les disséminer sur un maximum de nœuds. Lorsqu'un randonneur passe près d'une station de base (AP sur la Figure 2-6) qui est connectée à un réseau global comme Internet, le capteur qu'il embarque transmet les enregistrements qu'il contient afin qu'un système centralisé puisse reconstituer ou estimer le trajet de chaque personne ce qui permettra de les localiser en cas de disparition.

Projet COW (Handcock et al., 2009):

Le projet « Cow » adopte lui aussi ce type d'architecture où les nœuds capteurs sont mobiles alors que le puits est fixe.

Figure 2-7 : Bétail portant colliers GPS et bûcherons de proximité.

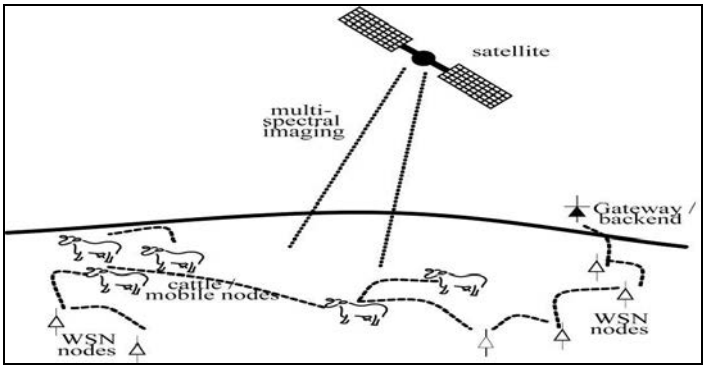


Figure 2-8 : Schéma du réseau adopté (Handcock et al., 2009)

Ce projet est une étude de cas d'un vaste élevage de bétail au nord de l'Australie qui a pu démontrer le potentiel de combiner des colliers GPS et des images satellites (Figure 2-7, Figure 2-8) dans un réseau de capteurs sans fil pour contrôler les préférences comportementales et sociales du bétail.

2.5.3 Architectures à puits mobile et nœuds capteurs fixes

Ce type d'architecture représente une solution intéressante qui est l'utilisation d'agents mobiles omniprésents qui se déplacent d'une manière aléatoire ou contrôlée pour rassembler des données lorsqu'ils se rapprochent des nœuds capteurs fixes. Dans la littérature, ces agents mobiles sont appelés des *MULEs (Mobile Ubiquitous LAN Extensions)* (Shah et al., 2003) (Jea et al., 2005), ou puits mobiles ou encore collecteurs de données mobiles. Les éléments mobiles peuvent être des robots, des humains, des animaux ou des véhicules. Ces éléments mobiles forment une couche de transport de paquets intermédiaire entre les points d'accès et les nœuds capteurs fixes déployés. Ils transportent les informations récoltées jusqu'à un relai ou un centre de traitement de données. Leur mobilité prévisible peut être utilisée pour l'économie d'énergie (Chakrabarti et al., 2003). Le temps de passage de l'élément mobile peut être fixé selon un calendrier de visite prévu. Ainsi, les nœuds capteurs ne passent en mode de transmission de données que lors du contact de l'élément mobile.

La Figure 2-9 illustre cette abstraction à trois niveaux qui correspondent à la récolte des données, leur convoyage par des mules et leur centralisation au niveau des points d'accès:

- une couche composée de points d'accès chargés de récupérer les données portées par les éléments mobiles et les transmettre à la plateforme de traitement de données. Le positionnement des points d'accès peut être fait selon les besoins en connectivité et en économie d'énergie.
- une couche représentant les éléments mobiles dont la mobilité peut être contrôlée, prévisible ou non. Ils assurent la connectivité du réseau et lui offre souplesse et évolutivité à moindre coût.
- une couche contenant les nœuds capteurs fixes destinés à mesurer les valeurs physiques des phénomènes à observer.

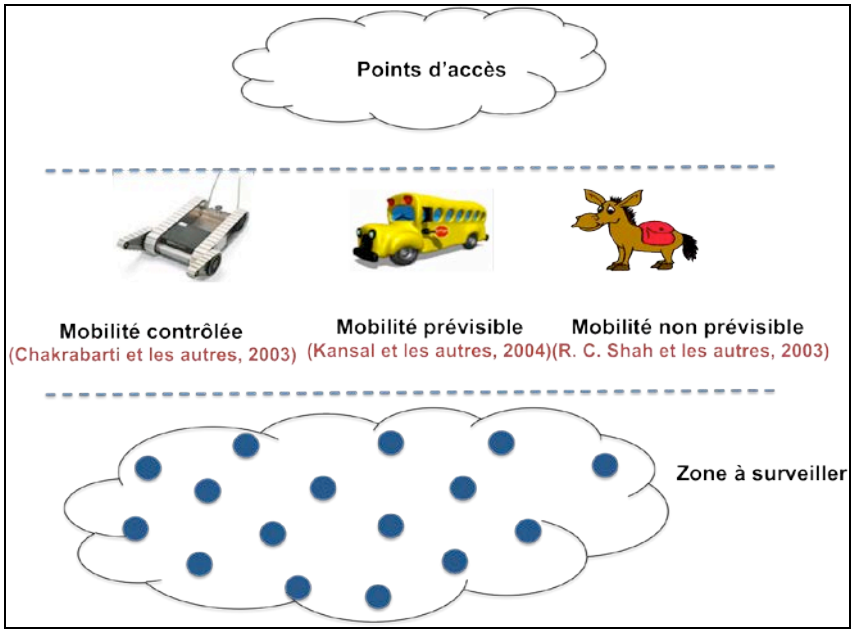


Figure 2-9 : Schéma d'architecture à puits mobile et nœuds capteurs fixes.

Cette architecture présente l'avantage de relaxer la contrainte de forte densité pour assurer une connectivité suffisante du réseau de capteurs sans fil déployé sur une large zone de captage. Ainsi, une architecture clairsemée de réseau de capteurs sans fil devient une option faisable permettant, ainsi, d'éviter les problèmes de collision et de pertes de messages induits par l'utilisation d'un réseau dense. Les éléments mobiles, lors de leur visite des nœuds capteurs fixes du réseau, peuvent établir des

connexions à un nombre de sauts réduit voire même à un simple-saut ce qui permettra de réduire les collisions et la perte de messages.

Un autre avantage apporté par l'utilisation des éléments mobiles dans le réseau réside dans le fait d'éliminer l'épuisement prématuré d'énergie des nœuds proches du puits. Les éléments mobiles peuvent aider à résoudre ce problème en allant collecter dans différentes zones du réseau, tout en assurant un équilibrage de charge énergétique plus au moins uniforme, même dans le cas d'une architecture dense du réseau. (Gandham et al., 2003) (Wang et al., 2007).

2.5.4 Architecture à puits mobile et nœuds capteurs mobiles

Dans ce type d'architecture, la mobilité est considérée aussi bien au niveau du puits collecteur de données qu'au niveau des nœuds capteurs déployés sur la zone d'intérêt.

Comme type d'applications environnementales utilisant ce type d'architecture nous trouvons le projet de suivi des zèbres (ZebraNet) (Juang et al., 2002).

Projet ZebraNet

L'objectif de ce projet est de déployer des capteurs sur des Zèbres de la réserve de Sweetwaters à MPALA en plein centre du Kenya.

C'est un projet interdisciplinaire qui a permis de conjuguer les efforts de différents laboratoires tels que le département d'informatique (*Dept. of Electrical Engineering*) et le département de biologie (*Dept. of Ecology and Evolutionary Biology*) de l'Université de Princeton.

Zèbre avec collier GPS

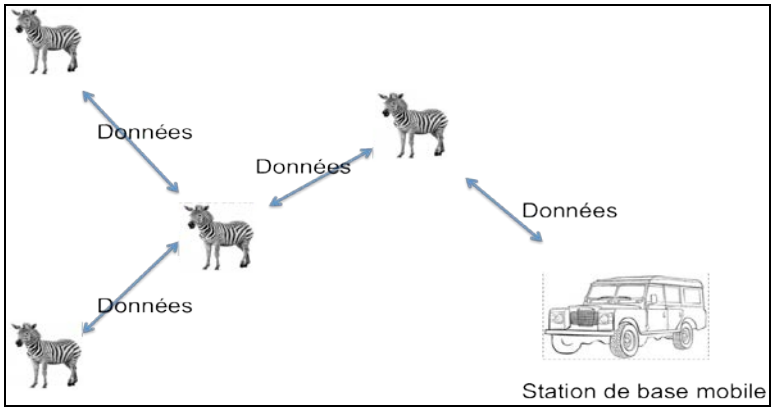


Figure 2-10 : Architecture de collecte de données dans ZebraNet.

Ce projet figure parmi les premières études réalisées dans le domaine de la surveillance de l'environnement qui prend en compte simultanément la mobilité des nœuds et du puits. En effet, dans

ce projet, les nœuds du réseau, «*zèbres équipés de récepteurs GPS*», sont mobiles et la station de base aussi. On considère que les chercheurs collectent les informations lorsqu'ils passent à proximité des zèbres par avion ou par voiture. (Figure 2-10)ZebraNet a pu faire face à plusieurs exigences sévères imposées par les chercheurs biologistes et a pu exiger de véritables percées dans la conception des systèmes informatiques, des protocoles sans fil ou encore dans la gestion énergétique des systèmes conçus.

Parmi les contraintes de départ fixées par les biologistes il y avait :

- le prélèvement des positions GPS des zèbres qui doit se faire toutes les deux minutes. Cette durée a été choisie suite à des travaux de recherche qui ont démontré que cette durée est assez longue pour enregistrer des statistiques comportementales indépendantes et encore assez fréquente pour avoir suffisamment de points.
- une année de fonctionnement du réseau sans aucune intervention humaine directe.
- le système doit être capable de fonctionner sur une grande surface de plusieurs milliers de km^2.
- pas de station de base fixe susceptible être l'objet de vandalisme.
- l'importance de collecter l'intégralité des fichiers logs.
- une taille du collier GPS comprise entre 1,36 et 2,27 Kg est recommandée. (Pour des animaux plus petits la limite serait encore plus petite).

Le premier déploiement a eu lieu en Janvier 2004 avec sept nœuds (six femelles et un mâle) dans la réserve *Sweetwaters* au Kenya sur une surface de 100 km^2. Durant ce déploiement, les positions des zèbres sont stockées toutes les 8 minutes. Par contre, toutes les deux heures chaque nœud cherche un autre nœud dans son voisinage capable de faire suivre les données collectées jusqu'à la station de base.

2.5.5 Architectures à puits multiples

Il existe aussi des solutions proposant une architecture à puits multiples. Cette solution consiste simplement à mettre plusieurs puits dans un seul réseau de capteurs sans fils. Dans cette structure, les données sont transmises aux utilisateurs finaux en utilisant l'un ou l'autre des puits du réseau. Le choix du puits est délégué au protocole de routage du réseau de capteurs. Un gros problème de cette solution réside dans la difficulté du routage des valeurs mesurées que les auteurs de l'article (Ciciriello et al., 2007) ont essayé de résoudre en proposant une solution, un peu complexe, d'optimisation du routage dans une architecture de type plusieurs-à-plusieurs.

L'utilisation de plusieurs puits mobiles au lieu des puits statiques dans la collecte des données s'avère plus efficace et augmente significativement la durée de vie du réseau de capteurs sans fil. Mais ajoute des difficultés de mise en place à surmonter telles la synchronisation et l'emplacement des puits.

Dans (Luo et al., 2005), l'emplacement des puits mobiles est périodiquement calculé pour que la durée de vie de réseau soit maximisée. D'autres travaux, exploitant plusieurs puits mobiles, se sont concentrés sur des approches de minimisation de l'énergie consommée par les capteurs (Gandham et al., 2003), ou la maximisation de la durée de vie du réseau global. (Basagni et al., 2008) (BenSaad et Tourancheau, 2009). Ces propositions souffrent souvent, du manque d'évolutivité. En effet si la taille du réseau de capteurs augmente et pour éviter les problèmes de l'architecture à puits unique, il va falloir rajouter des puits, ce qui ne peut pas se faire automatiquement. De plus, cette solution ne permet toujours pas une connexion directe entre l'utilisateur final et le capteur.

2.5.6 Synthèse

Suite à cet état d'art et comme le montre le Tableau 2-1, nous notons que :

- l'utilisation d'une architecture reposant sur l'exploitation des puits ou/et collecteurs fixes (considérés riches en ressources) s'avère non adéquate dans notre cas. En effet, dans un environnement où les nœuds capteurs sont mobiles, placer un puits ou/et collecteur fixe s'avère inadéquat. Celui-ci risque d'être hors de portée des nœuds capteurs mobiles. En se déplaçant, ils peuvent sortir de la zone de couverture du puits.
- bien que l'introduction d'éléments mobiles dans les architectures de réseaux de capteurs sans fil présente quelques défis à relever (temps de détection d'un nœud mobile entrant dans la zone de couverture, synchronisation des visites des nœuds mobiles avec les nœuds visités, …etc.), elle a permis de résoudre différents problèmes et a apportée des avantages conséquents comme :

❖ le gain en consommation, l'efficacité énergétique, l'équilibrage de charge énergétique et par conséquent un prolongement de la durée de vie du réseau (Wang et al., 2007).
❖ l'optimisation du routage et la diminution du délai de bout en bout. Du fait que le collecteur mobile permet vraiment de réduire le nombre de sauts que les données doivent effectuer.
❖ la résolution du problème de connectivité (Atay et Bayazit, 2009) en assurant un degré de connectivité du réseau très intéressant.

Type d'architecture	Avantages	Inconvénients
A puits statique et nœuds capteurs statiques (Werner-Allen et al., 2006) (Padhy et al., 2005) (Mainwaring et al., 2002)	Facile à mettre en œuvre Protocoles de routage simples	Pas d'équilibrage de charges (Problèmes de nœuds goulots d'étranglement en voisinage du puits) Perte d'énergie à la retransmission des données.
A puits statiques et nœuds capteurs mobiles (Handcock et al., 2009) (Huang et al., 2005)	Communication à 1-saut Pas de problème de connectivité du réseau. Pas de perte d'énergie du à la retransmission des données.	Risque d'isolement du puits, Les nœuds mobiles doivent convergés vers le puits pour faire le transfert des données. Difficulté d'optimisation de déploiement des puits.
A puits mobiles et nœuds capteurs statiques (Shah et al., 2003) (Jea et al., 2005) (Chakrabarti et al., 2003)	Equilibrage de charge Résout le problème de connectivité du réseau. Transfert de données à 1-saut Pas de perte d'énergie du à la retransmission des données.	Problème liés à la gestion de mobilité (aléatoire, contrôlée, prédictible) Temps de visite, durée de contact. Problème de synchronisation entre plusieurs puits mobiles
A puits mobiles et nœuds capteurs mobiles (Juang et al., 2002).	Déploiement aléatoire Gain en consommation l'équilibrage de charge énergétique Optimisation du routage diminution du délai de bout en bout Résolution des problèmes de connectivité du réseau.	Problème liés à la gestion de mobilité (aléatoire, contrôlée, prédictible) du puits mobile et des nœuds capteurs mobiles Temps de visite, durée de contact. Problème de synchronisation entre plusieurs puits mobiles

Tableau 2-1 : Tableau comparatif des différentes architectures utilisées

Ainsi, une architecture à un ou plusieurs collecteurs mobiles apparaît bien adaptée à notre application de supervision des nappes de pétrole en milieu marin. Les collecteurs, comme montré plus haut, servent à résoudre le problème de disconnectivité ressenti au niveau du réseau de captage.

Ces problèmes de connectivité peuvent provoquer

- des délais de bout en bout trop importants et variables
- de longues périodes de rupture de connexion
- un taux d'erreurs de transmission élevé
- une asymétrie de transfert de données (attente trop longue des messages d'acquittement)

Nous pouvons penser également à utiliser les réseaux DTN (Delay Tolerant Network) pour résoudre ce problème de connectivité au niveau de notre réseau. Ce type de réseau, introduit par Kevin Fall (Fall, 2003) et qui ciblait à l'origine les réseaux interplanétaires, peut apporter des solutions à ce genre de problème tout en prenant compte de la mobilité et de l'énergie limitée des nœuds du réseau. Le principe consiste quand une voie de sortie devient indisponible, à stocker sur le nœud tous les paquets qu'il reçoit en attendant le rétablissement de la liaison.

Ce principe a été également exploité dans les réseaux de capteurs sans fil pour optimiser la charge protocolaire pour économiser l'énergie.
Nous citons à titre d'exemple l'algorithme (*Potentiel-based Entropy Adaptative Raouting Optimized Diffusion :* PEAR OPD) proposé par (Vey et al., 2013) qui présente une façon optimisée de la diffusion des informations de routage au sein d'un réseau de capteurs sans fil dédié à la surveillance agricole. Ce réseau repose sur une architecture composée de nœuds de captage statique et de nœuds mobiles de collecte des données assurant la connectivité du réseau. L'objectif visé est de réduire la consommation d'énergie dans un réseau utilisant le protocole PEAR dont les performances ont été prouvées dans (Ochiai, 2008). La réduction de l'énergie se traduit par l'optimisation de la diffusion des messages protocolaires.

Cependant, le principe sur lequel repose le fonctionnement du réseau DTN, à savoir le principe de « store and forward » sans reposer sur une fiabilité de bout en bout, en implémentant une couche bundle entre la couche application et les couches spécifiques des réseaux régionaux a interconnecter, n'est pas adapté pour notre réseau de captage pour les raisons suivantes :

- le principe « Store and Forward » impose aux équipements interconnectés appelés « relais DTN » d'avoir des espaces de stockage trop importants. (Chaque nœud stocke dans son cache les informations reçues d'un autre nœud pour les transmettre quand le lien devient fonctionnel et disponible.). Or notre réseau est constitué de nœuds caractérisés par une faible capacité de stockage ce qui rend difficile l'implémentation de la couche bundle du réseau DTN au niveau du réseau de captage formé des nœuds capteurs de faibles capacité de stockage.
- Même au niveau des chefs de clusters, la donnée est traitée au fur et à mesure qu'elle est reçue des nœuds membres. Ainsi, le chef de cluster n'envoie au collecteur qu'une information synthétique comme par exemple, la valeur moyenne mesurée au sein de son cluster.
- La probabilité pour qu'une liaison se rétablisse entre deux mêmes nœuds et trop faible. En effet, notre réseau est caractérisé par la dispersion. Les nappes se divisent et s'étalent jusqu'à devenir des gouttelettes.

Néanmoins, le principe de base des réseaux DTN est exploité au niveau 2 de notre architecture. Ce niveau est constitué de collecteur(s) et la technique de DTN peut être applicable pour résoudre le problème de disconnectivité entre les collecteurs lorsqu'ils partent en compagne de collecte et donc s'éloignent de leur puits.

En conclusion, une fois l'architecture choisie où les collecteurs doivent visiter les chefs de clusters, nous devons diminuer le nombre de points de collecte que l'élément mobile doit visiter. Car, faire un balayage de tous les nœuds capteurs par l'élément mobile pour récupérer les informations mesurées au niveau de chaque nœud capteur est une tâche très onéreuse en temps de collecte.

Une solution possible et intéressante à ce problème est le clustering : les nœuds s'associent pour former des groupes (clusters), et transmettent leurs informations à leurs chefs de groupes, qui agrègent les données et les communiquent au collecteur mobile lors de sa visite. Ainsi, moins de paquets sont transmis et les mesures peuvent être fortement corrélées par le processus d'agrégation.

L'algorithme de clustering utilisé, sera discuté dans le chapitre suivant. De même, le problème de gestion de l'élément mobile sera discuté dans le troisième chapitre.

2.6 Architecture retenue pour notre système de monitoring des nappes de pétrole

Comme le montre la Figure 2-11, l'architecture de collecte des données retenue pour notre système de monitoring des nappes de pétrole en milieu marin est composée de 3 niveaux :

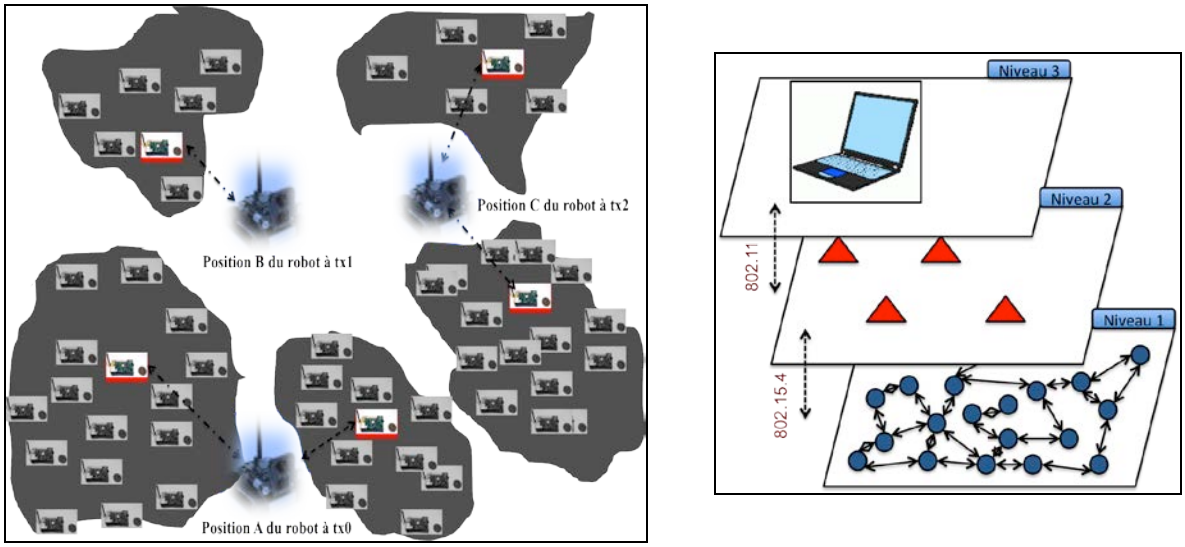


Figure 2-11 : Architecture du système de monitoring des nappes de pétrole.

Niveau 3 : à ce niveau nous retrouvons la plateforme dédiée à l'exploitation et au traitement des données. Cette plateforme dispose de riches ressources en mémoire, processeur,…etc.

Niveau 2 : ce niveau est constitué d'un ou plusieurs collecteurs mobiles destinés à collecter les données rassemblées par les chefs de clusters élus. Le collecteur mobile est doté de riches ressources en énergie, mémoire, processeur et de deux interfaces de communication. Une interface de communication fonctionnant sous 802.15.4 servant à communiquer avec les chefs de clusters (niveau 1) et une interface de communication opérant sous 802.11 servant à communiquer avec le niveau 3. Chaque collecteur mobile se déplace entre les nappes de pétrole, visite les chefs de clusters, établit des sessions de récupération des données. Une fois les données récupérées, il les transmet à la station de traitement (niveau 3). Puisque les chefs de clusters sont des capteurs standards c'est-à-dire non dotés de ressources importantes, nous assumons que la fréquence des visites du collecteur mobile et le temps de collecte d'informations des nœuds membres du cluster sont calculées de manière à ce qu'il n'y ait pas de débordement au niveau du buffer de stockage du chef de cluster. La trajectoire suivie par le collecteur est calculée en fonction des sessions déjà établies avec les chefs de clusters précédemment visités.

La Figure 2-11 illustre un exemple d'évolution de notre réseau durant l'intervalle de temps $[t_0, t_2]$. Le collecteur mobile se déplace entre les nappes de pétrole de la position A (à l'instant t_0) à la position C (à l'instant t_2) en passant par la position B (à l'instant t_1). Une fois arrivé au niveau d'une position, le collecteur mobile établit des sessions avec les chefs de clusters voisins et essaye de retrouver le contexte de ses précédentes visites et récupérer les données disponibles.

Niveau 1 : dans ce niveau, nous trouvons les nœuds capteurs destinés à mesurer les valeurs physiques telles la viscosité, la densité du pétrole sur les nappes, la dérive des nappes (en communiquant leurs positions) et d'autres types d'informations pouvant intéresser le centre de traitement de données. Ces données sensorielles mesurées par les dispositifs de captage du nœud capteur sont transformées en

données significatives facilement interprétées par les niveaux 2 et 3. Les capteurs de ce niveau sont repartis en groupes disjoints formant ainsi une topologie virtuelle. Ces groupes (clusters) sont identifiés par un chef de cluster. Ce dernier permet de coordonner les nœuds membres de son cluster, d'agréger les données collectées et de les transmettre au collecteur mobile lors de sa visite. Ainsi groupés, les nœuds forment un ensemble connexe permettant de remonter l'information vers les niveaux 2 et 3 à moindre coût en communication.

Ce partitionnement logique des nœuds capteurs prend en compte la mobilité des nœuds et la fragilité des liens entre les nœuds en assurant une stabilité satisfaisante de la structure du réseau. Il offre une vue logique du réseau, masque sa dynamique, et rend plus efficace le déploiement et le comportement du protocole de sessions que nous proposons en deuxième lieu. Ce partitionnement est réalisé par l'algorithme de partitionnement que nous présentons dans le chapitre 3.

2.7 Conclusion

Dans ce chapitre, nous avons présenté en premier lieu quelques généralités sur les réseaux de capteurs sans fil, à savoir la description d'un réseau de capteurs sans fil, ses caractéristiques spécifiques liées aux nœuds capteurs et au réseau et les types d'applications dédiées. Ensuite nous avons présenté, à travers des projets phares, les différentes architectures systèmes utilisées dans les réseaux de capteurs sans fil. Nous avons discuté les avantages et les inconvénients de chaque architecture.

Enfin, nous avons présenté l'architecture retenue de notre système de monitoring des nappes de pétrole. Cette architecture, comme montré dans la section 2.6, est composée de trois niveaux où le niveau 1 est composé de nœuds capteurs mobiles partitionnés en groupes disjoints selon un algorithme de clustering adapté à un réseau de capteurs sans fil mobile exploitant une métrique qui combine la mobilité, la densité et l'énergie des nœuds capteurs.

Chapitre 3 Clustering

Ce chapitre est divisé en trois parties :

- *La première partie est consacrée à définir la notion d'auto-organisation et de clustering. Un état de l'art des principaux algorithmes de clustering proposés dans la littérature est réalisé. Nous nous intéressons particulièrement aux algorithmes adaptés à la mobilité des nœuds capteurs. Nous concluons cette partie par notre positionnement par rapport à ces précédents travaux.*

- *Dans la deuxième partie de ce chapitre, nous allons présenter notre première contribution à savoir un algorithme de clustering adapté aux caractéristiques spécifiques de notre application (réseau homogène, forte mobilité). Nous présentons le fonctionnement de l'algorithme et les résultats d'évaluation de la métrique sur la stabilité des clusters formés.*

- *Dans la troisième partie, nous décrivons notre deuxième contribution : le développement d'un modèle de nœud capteur sous le simulateur Opnet. Cette partie débute par un justificatif du choix de la méthode « simulation » comme moyen d'évaluation et d'OPNET Modeler comme simulateur. Le modèle du nœud capteur proposé nous sert de support de test de notre algorithme de clustering.*

Une conclusion fait le résumé de nos deux contributions et propose une discussion des résultats obtenus et de leur évaluation.

3.1 Introduction

La conception des protocoles et services des réseaux de capteurs sans fils présente plusieurs défis à relever. Ces défis sont dus essentiellement aux caractéristiques propres de ce type de réseau. Ces caractéristiques peuvent se résumer dans les points suivants :

- Les nœuds capteurs composant les réseaux de capteurs sans fils sont très limités en puissance de transmission, en capacité de traitement et de stockage de l'information et en ressources énergétiques. Ainsi, les services et les protocoles développés pour les réseaux de capteurs sans fils, doivent être plus simples et peu gourmands en ressources par rapport à ceux fonctionnant sur les réseaux traditionnels.

- Les applications dédiées aux réseaux de capteurs sans fils : plusieurs scénarios d'applications sont envisagés pour les réseaux de capteurs sans fils et il est presque impossible de trouver, pour une tâche définie, un prototype unique de protocole adapté à tous ces scénarios potentiellement très différents. En effet la conception change selon les exigences des applications. Par exemple, les protocoles d'acheminement d'information (routage) tournant sur un réseau de capteurs sans fils dédié à une application environnementale diffèrent de ceux développés pour un réseau de capteurs sans fils dédié à une application industrielle où la notion de déterminisme et de temps réel sont des facteurs incontournables. De même, nous savons que le trafic de données dans les réseaux de capteurs sans fils présente une redondance significative puisque elles sont généralement produites par plusieurs capteurs déployés dans une même zone d'observation d'un phénomène physique commun. Une telle redondance doit être prise en compte, voire exploitée par les protocoles d'acheminement afin d'améliorer l'utilisation de la bande passante et l'énergie par exemple.

- La dynamique des réseaux de capteurs sans fils : dans beaucoup de scénarios initiaux d'applications, après le déploiement, la plupart des nœuds des réseaux de capteurs sans fils restent

stationnaires. Cependant, il existe actuellement aussi des applications où on permet aux nœuds capteurs de se déplacer pour contrôler des événements mobiles, ce qui crée des changements topologiques imprévisibles et fréquents. (Abbasi et Younis, 2007) (Akkaya et Younis, 2005)

En raison de telles caractéristiques, il parait plus judicieux, pour nous, de travailler sur une topologie logique, obtenue suite à une auto-organisation ou répartition du réseau, au lieu de travailler sur une topologie physique réelle offrant une vue à plat du réseau. La répartition logique du réseau va faciliter le déploiement de notre protocole de session.

De même, la topologie dynamique des réseaux de capteurs sans fil rend très difficile le développement d'algorithmes efficaces. Ainsi, nous avons choisi de repartitionner le réseau en espaces logiques bien ordonnés sur lesquels s'exécutera notre protocole de session. Donc pour nous, l'auto-organisation n'est pas une fin en soi mais une étape nécessaire qui permettra d'apporter une stabilité et une persistance dans le réseau afin de mettre en place notre protocole de session sur un réseau de capteurs sans fil qui affiche une dynamique importante.

Bien que les nœuds dans un réseau homogène soient identiques du point de vue des caractéristiques matérielles, chaque nœud possède un ensemble de paramètres qui le caractérisent par rapports aux autres nœuds du réseau. Tels l'énergie résiduelle du nœud capteur, son degré (nombre de ses voisins), sa mobilité, etc. Ces paramètres peuvent être exploités, de manière individuelle ou combinée, par l'auto-organisation.

Avant de voir les différentes techniques de partitionnement des réseaux de capteurs sans fils utilisées, nous jugeons utile de définir les notions d'auto-organisation / auto-structuration, clustering et clusters.

3.2 Définitions

3.2.1 Auto-organisation

Le préfixe « auto » fait référence à l'apparition de phénomènes collectifs dans l'ensemble des nœuds en interaction qui se réalise sans qu'il y ait besoin d'intervention externe ou d'une préparation initiale. Le terme « organisation » implique que le système possède une certaine structure et des fonctionnalités associées à cette structure. Ce qui signifie que les nœuds du réseau sont arrangés selon une logique définie et interagissent entre eux de façon identique et totalement distribuée pour un but commun.

Nous pouvons aussi définir l'organisation comme une structure obtenue suivant une logique propre pour atteindre un but précis. On parle alors de la société dans sa globalité ou de l'organisation d'une partie de l'effectif de la société visant la résolution d'un problème ou l'accomplissement d'une tâche.

L'organisation désigne aussi l'ensemble des responsabilités, pouvoirs et relations entre les personnes permettant à un organisme d'atteindre ses objectifs. Nous distinguons plusieurs types d'organisations du travail: pyramidale (la plus classique : les informations circulent de façon verticale du chef, tout en haut, au personnel de la base, tout en bas, en passant par des échelons intermédiaires), en comité ou jury (un groupe d'employés décide en tant que groupe, par vote par exemple), matricielle (une structure verticale associée à une structure horizontale), …etc.

Ainsi, l'auto-organisation indique également une apparition d'ordre et renvoie aux notions d'entropie (Lu et al., 2008). Par exemple, une structure virtuelle (comme un backbone de nœuds dominants) ou une topologie virtuelle qui émerge d'un ensemble homogène de capteurs donnant une structure permettant de diminuer l'entropie du réseau.

Aussi, la structuration peut être définie comme une fonction qui s'applique sur l'espace physique d'un réseau de capteurs sans fil pour avoir une structure virtuelle globale. Elle s'exécute de façon autonome sans aucune intervention extérieure et a pour objectif l'organisation et la facilité d'interaction de tous les nœuds du réseau. L'approche d'auto-organisation consiste ainsi à partitionner le réseau en un certain nombre de groupes de nœuds appelés « clusters » plus homogènes pour former une topologie virtuelle.

3.2.2 Clustering et cluster

Un cluster est un sous-ensemble de nœuds connexe, et la structuration ou clustering est le processus de regroupement des nœuds en clusters disjoints. Généralement et comme le montre la Figure 3-1, les clusters comportent trois types de nœuds :

- un nœud particulier appelé chef de cluster ou "cluster-Head" (CH). Ce dernier permet de coordonner les membres de son cluster, d'agréger et /ou de traiter les données collectées et de les transmettre au collecteur de données. Le chef de cluster est choisi pour jouer ce rôle soit d'une manière déterministe (chef de cluster prédéfini) ou d'une manière aléatoire (chef de cluster élu parmi les nœuds du réseau selon une métrique bien particulière ou une combinaison de métriques).
- un nœud passerelle ou "gateway" qui possède des liens inter-clusters et peut donc accéder à des clusters voisins et acheminer les données entre eux.
- enfin un nœud ordinaire ne possédant pas de liens avec les autres clusters et quand il s'attache à un chef de cluster il en devient membre.

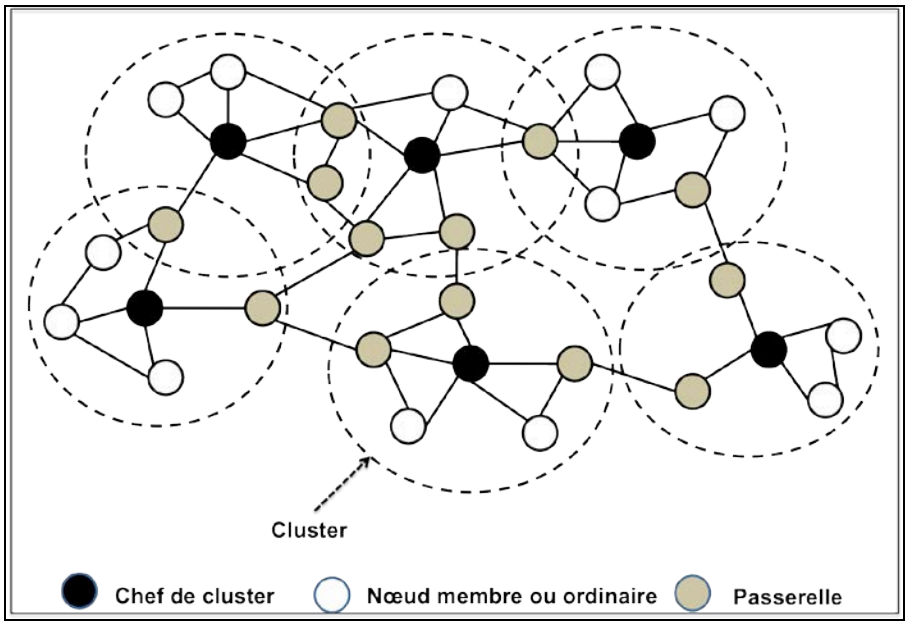


Figure 3-1 : Exemple de structure de clusters

3.3 Approches de partitionnement du réseau

Il existe dans la littérature une diversité d'algorithmes de formation de clusters qui peuvent être distingués et classés selon plusieurs paramètres tels le type de réseau et le type d'algorithmes utilisés :

- le type du réseau (réseau homogène : constitué de nœuds de même ressources ou réseau hétérogène : constitué de nœuds à ressources non égales) (Oyman et Ersoy, 2004) (Chen et al., 2004).

- le type d'algorithme utilisé :

  - centralisé où c'est le concepteur de l'algorithme ou la station de base ou le puits qui désigne les Chefs de clusters parmi les nœuds du réseau et ensuite, c'est le chef de cluster qui initie la construction de la structure virtuelle. Le nœud choisi peut être un nœud ordinaire ou encore un super-nœud ayant moins de contraintes de ressources (réserve d'énergie, portée radio, capacité de traitement,…etc.) ;

  - distribué où le chef de cluster est choisi suite à des interactions entre les nœuds pendant une durée déterminée ;

  - algorithme à 1-saut où chaque nœud à pour voisin un chef de cluster;

  - algorithme à K-sauts qui permet de construire un nombre faible de clusters disjoints où les nœuds sont au maximum à une distance K de leurs chefs de clusters). (Gupta et Younis, 2003) (Heinzelman et al., 2002) (Abbasi et Younis, 2007) (Vivek, 2010) ;

Plusieurs études bibliographiques ont essayé de faire des classifications des algorithmes de clustering selon différents angles de vue. Ainsi, les auteurs de (Abbasi et Younis, 2007) ont réalisé une taxonomie des différents algorithmes de clustering dédiés aux réseaux de capteurs sans fil. Ces algorithmes ont été comparés selon quelques métriques tels le taux de convergence, la stabilité de cluster, le recouvrement des clusters, la géolocalisation et la mobilité des nœuds. Alors que dans (Arboleda et Nasser, 2006) des algorithmes de clustering ont été comparés selon quelques concepts de base liés au processus de clustering, comme la structure des clusters, les types des clusters et les avantages du clustering. Tandis que (Kumarawadu et al., 2008) ont basé leur classification sur les paramètres de construction des clusters et les critères du choix d'un chef de cluster dans le mécanisme d'élection. Les auteurs ont discuté les algorithmes de clustering probabilistes basés sur les informations de voisinage et l'identité des nœuds.

Dans ce qui suit, nous allons classer les algorithmes de clustering selon deux grandes catégories : (i) les algorithmes de clustering qui reposent sur des épines dorsales virtuelles ou backbones» et (ii) les algorithmes de clustering n'utilisant pas cette notion de dorsales virtuelles lors de la formation des clusters.

3.3.1 Création des clusters reposant sur des dorsales virtuelles (Backbones)

La dorsale est un ensemble réduit du réseau constitué de nœuds forts connectés dit dominants connectés (*Connected Dominating Set, CDS*) dont l'objectif est d'optimiser la diffusion de l'information dans le réseau, et par conséquent prolonger sa durée de vie (Theoleyre et Valois, 2008). C'est en fait un ensemble de nœuds tel que tout nœud du réseau est voisin d'au moins un nœud du CDS, et tel que le CDS forme une structure connexe. La dorsale est importante et nécessaire pour la réduction de la charge du réseau en termes d'énergie, bande passante et nombre de messages. Une telle structure doit respecter certaines propriétés à savoir : la robustesse, l'extensibilité, la flexibilité (adaptation aux changements topologiques, paramétriques,…etc.), la persistance temporelle, l'auto-stabilisation,…etc.

La construction d'une telle dorsale respectant les exigences énumérées ci-dessus est basée sur un CDS de cardinalité minimale ou MCDS dont la détermination de façon centralisée ou distribuée est un problème NP-complet dans la théorie des graphes.

Ainsi de nombreux articles proposent des heuristiques de construction du MCDS. Par exemple, (Wu et Li, 1999) proposent un algorithme purement localisé permettant la construction d'un ensemble dominant déjà connecté (CDS) à partir d'un processus de marquage (un nœud v est marqué ou

dominant et membre de la dorsale si au moins deux de ses voisins ne sont pas connectés). Le CDS dérivant du processus de marquage est réduit à partir de deux règles 1 et 2 :

- règle 1 : un nœud dominant (u), ayant pour identifiant id(u), et dont l'ensemble des voisins est $N(u)$, change son marquage ou devient dominé s'il est couvert par un voisin (v), avec :

$$N(u) \subset N(v) \text{ et } id(u) < id(v)$$

- règle 2 : un nœud dominant (u) change son marquage ou devient dominé s'il est couvert par deux de ses voisins (v) et (w), avec :

$$N(u) \subset N(v) \cup N(w) \text{ et } id(u) < id(v) \text{ et } id(u) < id(w).$$

Sur cette même idée de réduire la cardinalité du CDS, les auteurs proposent une extension de leur algorithme dans (Dai et Wu, 2004) à travers la règle k qui est une généralisation des règles 1 et 2. Dans cette règle, un nœud dominant (u) change son marquage ou devient dominé s'il a le plus petit identifiant parmi les k nœuds dominants non voisins de (u).

Une autre amélioration est proposée dans (Dai et Wu, 2004) à travers la "coverage condition" permettant de réduire la redondance. Les auteurs montrent que le CDS dérivant des règles 1 et 2 remplacé plus tard par la règle k est un cas spécifique de la "coverage condition" (un nœud (v) n'est pas dominant ou n'est pas membre de la dorsale si pour tout nœud (u) et (w) voisins de (v), il existe un chemin de nœuds intermédiaires interconnectant les nœuds (u) et (w) tels que ces nœuds intermédiaires aient des plus fortes priorités (id ou combinaison d'autres métriques comme le degré,…) que le nœud (v).

Les auteurs de (Yang et al., 2007) s'inspirent de la méthode de construction du CDS utilisé dans (Dai et Wu, 2004) pour la construction d'un ensemble connecté dominant directionnel (*Directional Connected Dominating Set, DCDS*) en utilisant des modèles d'antennes directionnelles. Le choix du DCDS est dû au fait que l'utilisation d'une antenne directionnelle contribue d'une part à l'amélioration de la capacité du canal et à la conservation de l'énergie car la force du signal peut être augmentée dans une zone de couverture définie, et d'autre part à la réduction de l'interférence due à la zone de couverture limitée. Les auteurs supposent que chaque nœud utilise une antenne directionnelle pour la transmission et une antenne omnidirectionnelle pour la réception. Ainsi ils proposent de diviser l'étendue de transmission des nœuds en K secteurs ou directions identiques. Ce qui permet de restreindre les données envoyées dans certaines directions ou secteurs. Ils démontrent que le CDS construit dans (Dai et Wu, 2004) est un cas spécifique du DCDS où K est infini.

Après l'application du processus de marquage, le DCDS se construit en deux étapes à partir des conditions de couverture des nœuds et des liens (node and edge coverage). La condition de couverture de nœuds pour le DCDS est une modification de celle proposée dans (Dai et Wu, 2004). Selon la condition de couverture des nœuds, un nœud dominant (v) devient dominé ou non marqué si pour tous deux voisins dominants et absorbants (u) et (w),

- il existe un chemin dirigé de nœuds intermédiaires interconnectant ses nœuds voisins (u) et (w) tel que ces nœuds intermédiaires ont de plus fortes priorités que (v), et
- le nœud (u) a une plus forte priorité que le nœud (v) s'il n'y a pas de nœuds intermédiaires reliant les nœuds (u) et (w).

Selon la condition de couverture des liens, un lien ((v) → (w)) est non marqué ou ne fait pas partie de la dorsale s'il existe un chemin dirigé de nœuds intermédiaires interconnectant le nœud (v) au nœud (w) avec de plus fortes priorités que le lien ((v) → (w)). Selon les auteurs, l'algorithme du DCDS

réduit beaucoup plus le nombre de liens dominants dans un graphe non dirigé que dans un graphe dirigé, et génère moins de chemins dominants que l'algorithme de la règle k et l'approche générique de CDS.

Une généralisation de cette "*Coverage Condition*" originale est proposée dans (Dai et Wu, 2006). Ces auteurs proposent la construction d'un "*k-Connected k-Dominating Set*" ou k-CDS à travers la "k-Coverage Condition". Selon cette condition, un nœud (v) n'est pas dominant ou n'est pas membre de la dorsale si pour tout nœud (u) et (w) voisins de (v), il existe k chemins de nœuds intermédiaires interconnectant les nœuds (u) et (w) tels que ces nœuds intermédiaires aient des identifiants supérieurs à celui de (v).

La particularité du k-CDS réside dans le fait qu'après la suppression de k-1 nœuds quelconques, la partie restante de la dorsale forme toujours un 1-CDS ou CDS. Dans (Theoleyre et Valois, 2008), les auteurs construisent tout d'abord un ensemble maximal indépendant (*Maximal Independant Set, MIS* : structure dans laquelle aucun dominant n'est adjacent à un autre dominant et aucun autre nœud ne peut être choisi comme dominant sans violer cette contrainte) qui est par définition un ensemble dominant (*Dominating Set, DS*) et ensuite une structure connexe dite CDS en interconnectant les nœuds du MIS entre eux.

Pour la construction du CDS, Les auteurs définissent quatre états d'un nœud (dominant, dominé, actif, ordinaire) et un poids de stabilité énuméré comme une combinaison non linéaire de différentes métriques telles que la mobilité relative (M), l'énergie (∂), la distance à un degré optimal (Δ) et certains facteurs de pondération (α) et (β) pour l'élection d'un dominant.

$$Pstabilité = \partial(\alpha.(1+\Delta)^{-1} + \beta(1+M)^{-1})$$

Le MIS proposée par (Theoleyre et Valois, 2008) se fait en quatre étapes. La construction des clusters est initialisée par un leader (nœud devenant le premier dominant du CDS et permet par la suite de déterminer successivement les différents états des autres nœuds ; il peut être élu en utilisant l'algorithme décrit dans (Malpani et al., 2000) par un envoi de paquets "Hello" alors, ses voisins à k-sauts deviennent dominés, et les voisins à 2-sauts des dominés deviennent actifs (Etape 2). En effet, un nœud dominé envoie un paquet « Hello » avec un TTL (*Time To Live*) de kcds + 1 car un nœud actif doit être à au plus (2kcds + 1) sauts d'un autre dominant. Passé ce délai, deux nœuds actifs possédant un poids de stabilité maximum dans leur voisinage sont élus dominants (étape 3), leurs voisins deviennent dominés et les autres voisins actifs. La construction du CDS se finalise dans la quatrième étape où les nœuds notifient immédiatement leur changement d'état à leurs k-voisins via des paquets "Hello". Par la suite, cet ensemble dominant ou MIS sera interconnecté pour fournir un CDS. Cette interconnexion est encore initialisée par le leader via des messages "Hello".

Dans (Mnif et Kadoch, 2006), les auteurs construisent tout d'abord un ensemble minimum dominant (*Minimum Dominating Set, MDS*) et ensuite un ensemble connecté du MDS appelé MCDS. Les auteurs se basent sur l'approche de la programmation linéaire pour déterminer le MDS. Il s'agit en fait de minimiser $\sum_{i \in V} x_i$

Avec une variable $x_i = \begin{cases} 1 & si\ le\ noeud\ i \in MDS \\ 0 & sinon \end{cases}$

Sous la contrainte :

$X + M \times X \geq 1$ *ou* $X = [x_1 \quad \dots \quad x_n]$ *et* $M = \begin{bmatrix} m_{11} & \cdots & m_{1n} \\ \vdots & m_{ij} & \vdots \\ m_{n1} & \cdots & m_{nn} \end{bmatrix}$, $m_{ij} = 1$ *si (i) et (j) connectés.*

Ensuite, on détermine l'ensemble connecté du MDS en appliquant un algorithme d'arbre recouvrant minimum (*Minimum Spanning Tree, MST*).

Ces auteurs démontrent que cette méthode fournit une cardinalité optimale du MCDS car la première étape donne des nœuds avec un degré maximum si les nœuds sont placés dans une région avec une distribution de Poisson.

Dans la même approche des algorithmes de clustering reposant sur des dorsales virtuelles, Koné (Kone, 2011) a proposé un algorithme d'auto-organisation pour des réseaux de capteurs de grande taille (plus de 100000 nœuds capteurs). L'algorithme se base sur l'hétérogénéité des nœuds du réseau à 2 niveaux (le niveau 1 est composé de nœuds de captage regroupés autour des puits fixes riches en ressources « chefs de clusters » formant le niveau 2). L'objectif de l'algorithme est d'améliorer les performances réseaux en termes de délai, de pertes de paquets, de passage à l'échelle, et d'économies d'énergie. Dans le but de réduire les interférences radio et l'occupation de la bande passante, la structure des clusters est inspirée du réseau cellulaire en utilisant plusieurs canaux de fréquences. L'auteur a utilisé, de manière distribuée, l'allocation d'un canal de communication par cluster tout en respectant le principe de réutilisation de fréquence. Ainsi, tous les nœuds affiliés à un chef de cluster doivent communiquer sur le même canal. Les puits fixes forment un backbone sur lequel repose la création des clusters.

En résumé, nous pouvons conclure que la technique de formation des dorsales offre des structures assez robustes avec de bonnes performances dans les réseaux *ad-hoc*. Mais elle n'est pas adaptée aux réseaux de capteurs sans fil dynamiques du fait que la stabilité de la dorsale dépend des états des nœuds représentant les feuilles de la dorsale. C'est-à-dire qu'il faut que les états des nœuds restent inchangés pendant un laps de temps suffisant pour la construction de la dorsale.

3.3.2 Formation des clusters ne reposant pas sur des dorsales virtuelles

Cette approche est plus adaptée aux réseaux de capteurs sans fil dynamique. Elle consiste à découper le réseau en plusieurs groupes (clusters), homogènes, créant ainsi une topologie hiérarchisée du réseau. Dans la formation des clusters reposant sur un système d'élection des chefs, en plus de l'algorithme, le choix de la métrique pour évaluer la qualité de la formation des clusters est très important. La métrique représente un facteur de décision du choix du chef de cluster.

Cette métrique peut être de type monocritère :

$$Opt[g(x) \mid x \in A]$$ (Keeney et Raiffa, 1976)

Avec $\begin{cases} A : l'ensemble\ des\ actions\ admissibles; \\ X : l'action\ admissible; \\ g : la\ fonction\ critère\ à\ optimiser. \end{cases}$

L'action admissible peut être l'identité du nœud, le temps, la probabilité, la sémantique, la mobilité …etc.

De même la métrique peut être de type multicritère

$$Opt(g_1(x), g_2(x), g_3(x) g(x), \ldots\ldots g_n(x) \mid x \in A).$$

Et on utilise souvent la somme pondérée :

$$K = \sum_{i=1}^{k} \alpha_i P_i \quad . \text{ Avec } \sum_{i=1}^{k} \alpha_i = 1 \text{ et } \alpha_i \geq 0$$

Avec $\begin{cases} \alpha_i : un\ coefficient\ du\ critère; \\ P_i : la\ valeur\ critère; \\ K : le\ nombre\ des\ actions. \end{cases}$

Selon les cas, il peut être utile d'utiliser la métrique multicritère, appelée aussi métrique composite ou hybride. Ce type de métrique est généralement composée de deux ou plusieurs valeurs de mesure « métrique monocritère ». Ces valeurs de mesure peuvent être le degré du nœud, sa distance par rapport à ses voisins, sa vitesse du déplacement, son énergie restante ou encore son identifiant qui est souvent utilisé comme paramètre servant à départager des nœuds de même poids (Marin-Perianu et al., 2007).

Dans la littérature spécialisée, plusieurs algorithmes de clustering utilisant la métrique multicritère ont été proposés (Yu et Chong, 2005) (Liang et Li, 2007). Ces algorithmes sont conçus selon différents objectifs comme : la maintenance des clusters, l'équilibrage de charge (nombre de nœuds /cluster), la conservation d'énergie ou la mobilité. Ces algorithmes visent généralement à optimiser le nombre des chefs de clusters élus, le trafic de contrôle généré, l'énergie consommée lors de la formation des clusters, le mécanisme de réélection, la stabilité des clusters formés, …etc.

Dans ce qui suit, nous allons présenter quelques algorithmes, de type distribué, conçus sur le principe de l'élection des chefs de clusters dans un réseau homogène. Notre intérêt se portera sur les algorithmes exploitant la mobilité relative ou globale des nœuds comme l'une des valeurs de mesure dans leur métrique. Sans oublier les autres algorithmes basés sur d'autres critères de choix dans l'élection du chef de cluster. Cet aperçu d'algorithmes nous permettra de tirer profit de leurs avantages et éviter leurs inconvénients.

3.3.3 Algorithmes d'élection utilisant la mobilité

La mobilité est une caractéristique importante dans les réseaux de capteurs sans fil. Elle peut être considérée au niveau des nœuds capteurs destinés au captage ou au niveau des puits, ou encore au niveau des collecteurs. Lorsque la mobilité est trop rapide, la détection des voisins et la reconfiguration du réseau exigent habituellement un nombre important de messages de contrôle de la topologie et donc une consommation d'énergie importante.

Dans notre contexte, nous considérons un type particulier de réseaux de capteurs : les réseaux de capteurs mobiles dans lesquels la mobilité des capteurs et du collecteur sont pris en considération.

3.3.3.1 Mobilité globale

Plusieurs travaux ont proposé une métrique de mobilité géométrique globale. La métrique de mobilité est calculée entre toutes les paires de nœuds existant dans le réseau. Ce qui suppose l'utilisation d'un moyen de géolocalisation tel le GPS pour calculer les vitesses des nœuds. Ainsi, les auteurs (An et Papavassiliou, 2001) créent des clusters homogènes en terme de mobilité géométrique globale. L'algorithme présenté crée des clusters selon le niveau de mobilité globale des nœuds. Ainsi, les nœuds qui ont une faible mobilité peuvent appartenir au même cluster. Et inversement, deux nœuds ayant une trajectoire différente risquent de former des clusters disjoints.

Basagni (Basagni, 1999) *a* proposé l'algorithme de formation de clusters distribué (*Distributed Clustering Algorithm*) (DCA) où la métrique utilisée est définie par l'inverse de la vitesse. Ce qui favorise les nœuds les moins mobiles à devenir chefs de clusters et assure ainsi une bonne stabilité des clusters. L'algorithme DCA est une généralisation d'algorithmes de formation de clusters à 1-saut. Il a été conçu pour des réseaux dans lesquels les nœuds sont quasi-statiques.

L'algorithme adopte les propriétés suivantes :

- tout nœud membre a au moins un chef de cluster comme voisin ;
- le chef de cluster de tout nœud membre est le voisin de plus grande métrique ;

- deux chefs de clusters ne peuvent pas être voisins ;
- le nœud ayant la plus grande métrique dans son voisinage, à une distance de moins de deux sauts, sera élu chef de cluster, sinon il doit joindre un cluster existant.

L'algorithme DCA utilise deux messages : « ch » *(cluster head)* et « join ». Le message « ch » est envoyé par un nœud (u) pour annoncer à ses voisins qu'il est chef de cluster et le message « join » est envoyé par un nœud (v) pour avertir ses voisins qu'il a rejoint un cluster. Initialement, un nœud dont la métrique est plus grande par rapport à ses voisins se déclare chef de cluster, et envoie un message « ch ». Un nœud recevant un message « ch » d'un noeud (u), rejoint le cluster (u) si tous ses voisins de métrique plus grande que lui ont rejoint un cluster. En recevant un message « join » du nœud (v) trois cas sont possibles :

- le nœud est un chef de cluster donc il ajoute à sa table des membres du cluster le nœud (v).
- le nœud n'est pas chef de cluster et tous ses voisins de métrique plus grande que lui ont rejoint un cluster, alors le nœud devient chef de cluster.
- le nœud n'est pas chef de cluster et un des ses voisins de métrique plus grande que lui est chef de cluster, alors le nœud rejoint le cluster dont la métrique du chef est la plus grande.

L'algorithme (DMAC) (*Distributed and Mobility adaptive Clustering*) (Basagni et Jonsson, 1999b) est une variante de l'algorithme DCA qui vise la maintenance du cluster en considérant la mobilité du nœud pendant ou après la construction du cluster. Il permet également à chaque chef de cluster d'avoir jusqu'à k-voisins comme chefs de clusters, et réduit ainsi le nombre de réaffectations en ajoutant un seuil h de reformation de clusters. C'est-à-dire qu'il n'y a de réaffectation que si la métrique d'un nouveau candidat au rôle de chef de cluster est supérieure de h à la métrique du chef de cluster actuel.

Notons que l'algorithme DCA est bien adapté pour les réseaux où la mobilité des nœuds est très faible tandis que l'algorithme DMAC sera plutôt adéquat pour les réseaux à moyenne et forte mobilité des nœuds. Cependant, l'attribution des poids aux nœuds n'a pas été discutée dans les deux algorithmes et il n'y a aucune optimisation sur le nombre de paquets échangés entre les nœuds et le contrôle de la puissance.

Une généralisation de DMAC est G-DMAC (Basagni, 1999), où un nœud nouvellement initialisé, rejoint le chef de cluster du poids le plus grand dans son voisinage à 1-saut (semblable à DMAC). Cependant, durant les changements de topologie, le nœud reste membre de son cluster tant qu'il n'existe pas un autre nœud de statut « chef de cluster » dont le poids dépasse celui du chef de cluster auquel il est attaché d'une valeur seuil (h>=0). Ce facteur permet de limiter les ré-affiliations entre les clusters. Ainsi, les nœuds ne se ré-affilient à un nouveau chef de cluster que lorsque son poids est supérieur du facteur (h) au poids du chef de cluster courant. Aussi, pour améliorer la stabilité du cluster, un autre facteur (k) a été introduit qui définit le nombre maximal de chef de clusters voisins directs que l'on permet à un chef de cluster d'avoir. Toutefois cette solution ne permet la formation que de clusters à 1-saut et les facteurs de performance (h) et (k) sont difficiles à spécifier de manière judicieuse.

L'algorithme de clustering C4SD (Marin-Perianu et al., 2008) est spécifiquement conçu pour les réseaux mobiles, où le poids est représenté par la capacité du nœud et son degré de dynamicité. Chaque nœud du réseau choisit un parent ayant le plus grand poids parmi ses voisins. Si un tel nœud n'existe pas, alors ce nœud se déclare chef de cluster. Le résultat de cet algorithme donne des clusters disjoints dont les chefs de clusters forment des ensembles indépendants.

3.3.3.2 Mobilité relative

La mobilité relative est une métrique difficile à mesurer du fait qu'il faut bien modéliser le comportement d'un nœud mobile par rapport aux autres en mesurant sa vitesse. Ce qui implique une connaissance à chaque instant des coordonnées du nœud dans l'espace et sa vitesse relative par rapport à ses voisins. P. Basu et ses collègues (Basu et al., 2001) ont proposé MOBIC (*Mobility Based Metric for Clustering in Mobile Ad hoc Networks*) pour former des 1-clusters. La métrique proposée est la mobilité relative entre deux nœuds. La mobilité relative d'un nœud est représentée dans cet algorithme comme le rapport des niveaux de puissance des transmissions successives reçues par un nœud de ses voisins.

Elle est représentée par la relation suivante :

$$M_y = 10' \log\left({}^{R_x P_{reçue\,(nouv)}^{x \to y}} \Big/ {}_{R_x P_{reçue\,(old)}^{x \to y}} \right)$$

Avec

$$\begin{cases} R_x \text{ la distance entre l'émetteur et le récepteur} \\ P_{reçue}^{x \to y} \text{ la puissance du signal reçu au récepteur Y émis par un émetteur X.} \end{cases}$$

Ainsi, le calcul de la mobilité d'un nœud nécessite juste la connaissance de la puissance du signal émis par rapport au bruit. En effet, le signal reçu peut donner une indication sur la distance entre un émetteur et son récepteur à un saut. En réalité, un calcul exact de la distance entre un émetteur et un récepteur peut ne pas être obtenu à partir de la mesure de l'énergie des signaux, mais plutôt à partir du niveau d'énergie reçue entre deux transmissions successives (réception de deux messages Hello consécutifs) avec le nœud voisin. De même, le calcul de la mobilité d'un nœud peut se faire sur un ensemble de voisins. Pour un nœud ayant (m) voisins, il faut calculer la mobilité agrégée en calculant la variance de toutes les mobilités :

$$M_y = \text{variance}\ (M_y\ (X_1), \ldots, My\ (X_m)).$$

Une grande valeur de M_y indique que le nœud Y est très mobile par rapport à ses nœuds voisins X_i. Par contre, une petite valeur indique que le nœud Y est presque fixe par rapport à ses nœuds voisins, ce qui implique une élection potentielle de ce nœud en tant que chef de cluster.

MOBIC utilise la même technique de maintenance que LCC (*Least Cluster Change algorithm*) (Agarwal et al, 2009) à laquelle il ajoute une règle supplémentaire pour minimiser le coût de l'opération de maintenance : si deux chefs de clusters se trouvent voisins, alors l'un des deux n'abandonnera son statut qu'après une certaine durée (t), sinon ils gardent tous les deux leur statut. Cette règle permet de ne pas faire appel à l'opération de maintenance quand deux chefs de clusters se trouvent dans le même voisinage.

Les résultats présentés par les auteurs montrent que MOBIC assure une bonne stabilité pendant la formation des clusters. Et que l'optimisation apportée par MOBIC dans la phase de maintenance a permis de réduire à plus du tiers le nombre de changements des chefs de clusters relativement au protocole LCC. Cependant, les limitations de l'algorithme LCC ne sont pas totalement éliminées. De plus, MOBIC exige que les nœuds soient capables d'estimer le niveau de signal émis par leurs voisins pour calculer la mobilité relative.

Chatterjee et ses collègues (Chatterjee et al., 2001) présentent l'algorithme WCA qui utilise une métrique composée du degré du nœud, de la somme des distances à tous les voisins, de la vitesse du nœud, et du temps cumulé passé dans le rôle de chef de cluster. Les résultats présentés par les auteurs ont montrés que WCA offre de meilleures performances que les autres algorithmes tels que MOBIC ou LCC. Toutefois, le mécanisme de clustering utilisé dans cet algorithme nécessite une

synchronisation globale de tous les nœuds du réseau. De même, le chef de cluster perd beaucoup de ses performances quand la taille du cluster qu'il gère devient trop grande ce qui dégrade le niveau de stabilité du réseau.

Lehssaini (Lehsaini, 2009) a utilisé aussi la mobilité relative dans la métrique utilisée par son algorithme CSOS (*Cluster-based Self-Organization Scheme*) pour optimiser la consommation de l'énergie dans les réseaux de capteurs sans fil.

La mobilité est définie par la relation suivante :

$$M_y(t) = \frac{1}{\Delta t}(|du(t+\Delta t) - du(t)|)$$

Avec

$$du(t) = \frac{1}{|N1(u)|}\sum_{v \in N1(u)} dist_{(u,v)}(t) \quad \text{et} \quad dist_{(u,v)} = \sqrt{(X_u(t) - X_v(t))^2 + (Y_u(t) - Y_v(t))^2}$$

Où

$$\begin{cases} du(t): \text{la distance euclidienne moyenne séparant le nœud (u)de ses voisins à 1 saut à l'instant t.} \\ dist_{(u,v)}(t): \text{la distance euclidienne entre les nœuds (u)et (w)à l'instant t.} \\ X_u(t) \text{ et } Y_u(t) : \text{les coordonnées du nœud (u)à l'instant t.} \end{cases}$$

Selon Fabrice Theoleyre (Theoleyre et Valois, 2008), la métrique relative a été définie par:

$$M_{moy} = \frac{\left|\frac{N(t+\Delta t)}{N(t)}\right| + \left|\frac{N(t)}{N(t+\Delta t)}\right|}{|N1 \cup N(t+\Delta t)|}$$

Avec N(t) : ensemble des nœuds voisins du nœud N à l'instant t

(Marin-Perianu et al., 2007) proposent un algorithme de clustering spontané pour les réseaux de capteurs mobiles faisant face au même contexte (comme le déplacement d'ensemble). Les auteurs supposent que chaque nœud exécute un algorithme de reconnaissance de partage de contexte, qui fournit un nombre à une échelle, représentant la valeur de confiance que deux nœuds sont ensemble. Chaque nœud calcule périodiquement la confiance de partager le même contexte avec ses voisins. Le nœud ayant le poids le plus grand parmi ses voisins, avec qui il partage un contexte commun, se déclare chef de cluster. Un nœud régulier souscrit au chef de cluster avec lequel il partage un contexte commun et qui a le poids le plus grand.

Les auteurs de (Marin-Perianu et al., 2008) présentent un algorithme de clustering généralisé (*A Generalized Clustering Algorithm for Dynamic Wireless Sensor Networks*). Cet algorithme prend des décisions localisées basées sur des informations de voisinage à 1-saut et produit des clusters disjoints. L'algorithme représente une généralisation d'algorithmes de clustering basés sur le poids conçus pour les réseaux *ad-hoc* et les réseaux de capteurs, en utilisant un ensemble de variables générales et de conditions.

3.3.4 Algorithmes d'élection utilisant d'autres métriques

D. J. Baker a présenté deux algorithmes qui utilisent l'identité comme métrique: l'algorithme (*Linked Cluster Algorithm* : LCA) (Baker, 1981) et sa variante (LCA2) (Ephremides et al., 1987) qui réduit le nombre de chefs de clusters. Dans le premier algorithme, un nœud (u) devient chef de cluster s'il a la plus grande identité parmi ses voisins ou s'il existe au moins un nœud voisin (v) tel que (u) est le nœud qui a la plus grande identité dans le voisinage du nœud (v). Tandis que dans le deuxième algorithme (LCA2), le nœud ayant la plus petite identité parmi les voisins non couverts sera élu chef

de cluster. Notons que LCA et LCA2 ont été développés pour s'adapter aux petits réseaux de moins de 100 nœuds.

Les auteurs de (Chatterjee et al., 2001) ont pris la durée de vie du chef de cluster et le nombre de changements d'état du chef de cluster comme paramètres d'évaluation de la stabilité des clusters. (Plus il y a restructuration du cluster plus il y a génération de messages de contrôle de maintenance. Ce qui réduit les performances du réseau).

(Baker, 1981) (Bandyopadhyay, 2003) (Younis, 2004) (Heinzelman et al., 2002) se sont intéressés à la taille moyenne des clusters et leur nombre. Ces paramètres caractérisent l'utilité du cluster. Plus le nombre de clusters est petit (donc clusters de grande taille), plus ils sont utiles. Mais plus les clusters sont de petite taille (donc en grand nombre), plus ils sont stables. Généralement, il faut faire un compromis entre stabilité et taille des clusters. Les auteurs de LMC (*Limiting Member node Clustering*) (Naruephiphat et Charnsripinyo, 2009) utilisent une valeur seuil pour limiter le nombre de nœuds par cluster ; la métrique utilisée est une fonction coût déterminée selon le niveau de batterie, la consommation d'énergie et la distance séparant le nœud au chef de cluster. Par contre, les auteurs de (Heinzelman et al., 2002) présentent un algorithme où un nœud devient chef de cluster selon une valeur probabiliste. La probabilité est définie telle que le nombre désiré de chefs de clusters est atteint. Elle peut dépendre du nombre de nœuds dans le réseau, de l'énergie globale ou résiduelle, du nombre de fois où le nœud a été chef de cluster, de la taille de cluster, etc..

Selon Mitton, (Mitton, 2006) la densité est définie par le ratio du nombre de liens par le nombre de nœuds dans le k-voisinage du nœud. Périodiquement, chaque nœud calcule sa densité et la diffuse à son (1-voisinage). Ainsi chaque nœud est en mesure de comparer sa propre densité avec celle de ses voisins et à partir de là, il décide de s'élire comme chef de cluster à condition de posséder la plus forte densité, soit de choisir comme parent son voisin de plus forte densité. Et en cas d'égalité et afin de privilégier la stabilité de la structure, le nœud choisit, s'il est en course, sera celui qui est déjà élu au tour précédent, sinon celui de plus petit identifiant. Cette méthode d'élection construit implicitement une forêt couvrante orientée.

Les auteurs de (Bouhafs et al., 2006) ont exploité la sémantique comme métrique du fait que les propriétés sémantiques se rapportent au rapport entre les paires de nœuds ou parmi des nœuds dans un groupe. Les propriétés sémantiques incluent la distance entre les nœuds, les chemins de disponibilité entre les nœuds, la mobilité absolue ou relative, l'emplacement ou le type d'événement détecté. Des clusters peuvent être formés selon les propriétés sémantiques semblables des nœuds.

3.3.5 Conclusion

Selon cette étude des plus importants algorithmes connus dans la littérature, nous pouvons isoler un certain nombre de paramètres de construction de la structure virtuelle, comme la densité, la mobilité et l'énergie. Ces paramètres peuvent être pris en compte de manière isolée ou combinée. De plus, chacun de ces paramètres peut être sujet à différentes définitions notamment la densité qui peut être envisagée à 1, 2 ,3 ou k-sauts. Enfin différents types de pondération ont été proposés. Nous constatons aussi que quelque soit l'influence de l'application considérée, l'algorithme de formation de clusters comporte, généralement, deux grandes phases : une phase d'élection des chefs de cluster et une phase de maintenance des clusters formés. Et la méthodologie la plus souvent appliquée, est composée des étapes suivantes :

- Identification d'une métrique de qualité ;
- Détermination de la fonction poids en fonction des métriques choisies selon les objectifs désirés par la répartition ;

- Choix de l'algorithme adapté ;
- Utilisation de la métrique et de l'algorithme pour le choix du chef de cluster.

Dans la section suivante, nous présentons notre première contribution qui consiste à proposer un algorithme de clustering distribué adapté à notre application particulière où la forte dynamique du réseau ne permet pas l'emploi d'une infrastructure repositionnée. Cet algorithme est basé sur le principe d'élection de chef de cluster qui utilise une métrique intégrant les paramètres énergie résiduelle, densité et mobilité. Nous montrerons via la modélisation de cet algorithme sous l'environnement de simulation réseau Opnet Modeler, qu'il est possible d'utiliser la simulation pour définir les coefficients de pondération, utilisés dans la métrique, appliqués à notre application de surveillance des nappes de pétrole en milieu marin.

3.4 Algorithme de partitionnement du réseau (notre 1ère contribution)

Cette section est consacrée à la présentation de notre algorithme de répartition du réseau. L'algorithme proposé a été conçu pour qu'il soit adapté à notre application. L'application visée repose sur l'utilisation d'un réseau de capteurs sans fil homogène dont la principale caractéristique réside dans sa dynamique. Cet algorithme se base sur le principe d'élection du chef de cluster parmi les nœuds du réseau. L'élection est faite selon une métrique composée de l'énergie résiduelle du nœud, de sa densité et de sa mobilité. Nous avons privilégié ce mécanisme d'élection car il est à la fois simple et particulièrement adapté à la structure du réseau que nous souhaitons mettre en place.

Comme il a été déduit suite à notre état de l'art sur les principaux algorithmes, il est essentiel de savoir définir un chef de cluster et de donner à chacun des autres nœuds capteurs les moyens de communiquer avec lui. Le chef de cluster est généralement choisi en fonction d'un critère de poids (métrique), et il est d'usage de choisir celui dont le poids est un extremum. Du fait que tous les nœuds capteurs du réseau possèdent des identités distinctes (adresse physique), le problème de l'élection se ramène à donner à chaque nœud capteur les moyens de reconnaître le chef de cluster. Une fois le choix du chef de cluster effectué, il est nécessaire d'une part de communiquer l'identité de ce dernier à tous les autres nœuds capteurs, et d'autre part, d'établir des chemins de communication de ces nœuds vers le chef de cluster.

La première méthode pour réaliser ces tâches consiste à effectuer le choix du chef de cluster et à établir des routes de manière statique. Cette solution manque de souplesse ce qui a conduit à la recherche d'autres solutions dites dynamiques. Ces solutions appliquent ce qui est convenu d'appeler un algorithme d'élection.

Le problème d'élection est un problème de calcul d'extremum selon diverses bornes (inférieures et supérieures), établi selon la topologie du réseau ou selon l'information sur la topologie du réseau qui est mise à la disposition locale des nœuds capteurs.

3.4.1 Hypothèses

Pour construire la structure virtuelle, nous considérons les hypothèses suivantes sur lesquelles se base l'algorithme proposé :

3.4.1.1 Hypothèses sur le réseau

Hypothèse 1. Les nœuds ne peuvent s'échanger de l'information que par messages (donc pas de mémoire partagée).

Hypothèse 2. Le réseau de communication est modélisé par un ensemble de sous-graphes (clusters) connexes, simples et symétriques.

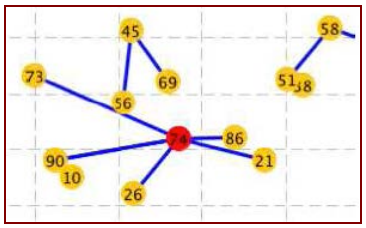


Figure 3-2. Exemple de sous-graphe connexe

Hypothèse 3. Pas de perte de messages ; tout message émis par un nœud vers un autre nœud finit par être reçu par ce dernier au bout d'un temps fini. Le temps dépend de l'état du réseau (fluide ou congestionné). De même, les messages ne sont pas altérés ou modifiés durant leur transmission.

3.4.1.2 Hypothèses sur les nœuds

Hypothèse 4. Tous les nœuds ont les mêmes capacités en termes de mémoire, puissance de calcul, d'énergie et portée radio. Chaque nœud est identifié par un identifiant unique qui n'est que son adresse physique et il connaît sa position. La position du nœud peut être déterminée par une méthode de géolocalisation adaptée à un réseau de capteurs sans fils dynamique. Dans la littérature, il existe plusieurs techniques utilisées telles :

- Techniques géométriques d'estimation de position :
  - la trilatération : la position estimée d'un nœud est calculée en fonction des positions des nœuds de référence et de leur distance à ce nœud.
  - La triangulation : le nœud estime sa position en fonction des positions des nœuds de référence et de l'angle d'arrivée du signal provenant des nœuds de référence.
- Techniques basées sur l'angle d'arrivée (Cheng, 1989)
- Techniques basées sur la distance :
  - Les mesures basées sur le temps de propagation à sens unique (Carter, 1981) ou aller-retour (Gezici, 2005).
  - Les mesures de différence de temps d'arrivée TDOA (Time Différence Of Arrival) (Chan, 1994), (Smith, 1987).
  - Mesures basées sur l'intensité du signal reçu (Chan, 1994): Les mesures basées sur l'intensité du signal reçu (RSS) estiment les distances entre les capteurs voisins à partir des mesures de l'intensité du signal reçu entre les deux capteurs (Correal et al., 2005).
- Techniques basées sur le profilage RSS : Les techniques de localisation basée sur le profilage RSS ont été principalement utilisées pour l'estimation de l'emplacement dans les réseaux locaux sans fil (WLAN's), mais ils semblent être adéquats aux WSNs. (Krishnakumar et al., 2004).

En règle générale, les algorithmes basés sur la localisation AOA et les mesures du temps de propagation sont en mesure de parvenir à une meilleure précision que les algorithmes des mesures de localisation basés sur RSS.

Hypothèse 5. Chaque nœud échange des messages avec ses voisins directs. Considérons deux nœuds capteurs (u) et (v). Un point sur le périmètre de nœud capteur (u) est couvert par le nœud capteur (v), si ce point est dans la zone de couverture du nœud capteur (v). De manière générale, nous disons que

le périmètre de la zone de détection du nœud (u) est k-couvert si tout point sur ce périmètre est couvert par au moins k capteurs en plus de (u).

Hypothèse 6. Un nœud n'aura jamais une vision globale de la structure globale du réseau.

Hypothèse 7. Initialement, les nœuds capteurs sont dispersés de manière aléatoire sur les nappes de pétrole.

3.4.2 Calcul de la métrique

La métrique que nous avons utilisé est une combinaison de l'énergie, de la densité et de la mobilité des nœuds. Elle est calculée par la relation suivante :

$$Poids_u(t) = \varepsilon_u(t)(\alpha\rho_{u,1}(t) + (1-\alpha)M_u(t)) \quad (1)$$

Avec

$\rho_{u,1}(t)$: Densité du nœud (u) à l'instant t. $\rho_{u,1}(t) = \frac{N_{u,1}(t-\Delta t)}{N_{u,2}(t-\Delta t)}$ (Mitton, 2006) (2)

$N_{u,k}(t)$: Nombre de voisins à k-sauts du nœud (u) à l'instant t.

$M_u(t)$: Mobilité de (u) à l'instant t. $M_u(t) = \frac{(|d_u(t) - d_u(t-\Delta t)|)}{\Delta t}$ (Lehsaini, 2009) (3)

Avec : $d_u(t) = \frac{1}{|N_{u,1}(t)|}\sum_{v\in N_{u,1}(t)} dist_{(u,v)}(t)$ (4)

Et : $dist_{(u,v)}(t) = \sqrt{(X_u(t) - X_v(t))^2 + (Y_u(t) - Y_v(t))^2}$ (5)

$\varepsilon_u(t)$: L'énergie résiduelle du nœud (u).

α : Paramètre qui dépend de l'application. Les valeurs choisies de ce paramètre, adaptées pour notre application, seront justifiées par simulation.

3.4.2.1 Analyse de la métrique choisie

Chaque cluster a un chef de cluster élu selon la valeur de son poids qui est une combinaison de la densité, l'énergie résiduelle et la mobilité.

Paramètre densité : Le choix de la densité comme valeur de mesure, dans la métrique composite, se justifie du fait que la densité limite la reconfiguration de la topologie et offre une bonne stabilité face à la mobilité des nœuds. Nous choisissons la 1-densité, c'est à dire calculée à 1-saut, au lieu de la k-densité (k>1) selon le travail de Mitton (Mitton et al., 2009) qui a démontré que la 1-densité fournit une structure plus robuste comparée à la K-densité. En plus, la 1-densité minimise l'échange des informations sur le réseau et donc la bande passante et la consommation d'énergie.

La Figure 3-3 illustre le calcul de la densité. Le nœud (H) a cinq voisins, donc il a un degré égal à cinq. Le nombre total de liens des voisins est égal à huit (cinq liens avec ses voisins directs plus les trois liens entre ses voisins). La densité du nœud (H) sera alors égale à (8/5).

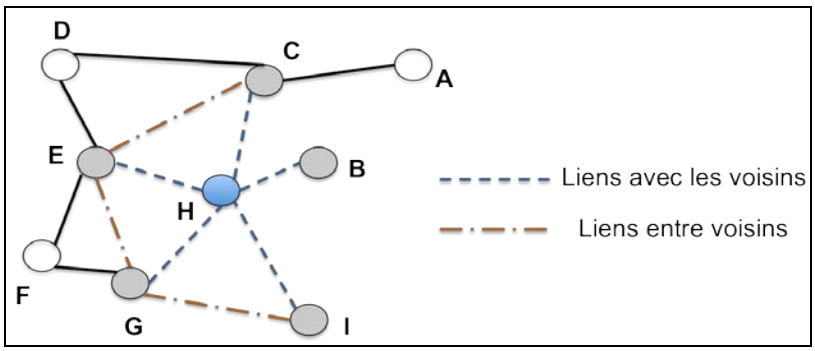


Figure 3-3. Exemple illustratif du calcul de la densité

Le choix de la 1-densité a pour conséquence que chaque cluster a un chef de cluster élu dans son 2-voisinage.

Paramètre mobilité : Le choix de la mobilité relative a été fait selon notre étude bibliographique des différents algorithmes utilisant la mobilité globale et ceux utilisant la mobilité relative. Cette étude a montré que la mobilité relative assure une bonne stabilité de l'architecture hiérarchique générée et évite par la suite les réactions en chaîne résultantes de l'instabilité de la structure.

Paramètre énergie : L'énergie est un paramètre incontournable à prendre en considération dans toute manipulation dans les réseaux de capteurs sans fil. Ce paramètre est favorisé par rapport aux autres paramètres constituant la métrique afin d'assurer une longue vie du réseau. En effet, un nœud qui possède une énergie résiduelle très faible a moins de chance de devenir un chef de cluster même s'il possède une grande densité ou une faible mobilité.

En conclusion, nous estimons que le choix d'exprimer la métrique selon la relation (1) en pondérant la densité et la mobilité par des facteurs α et $\beta = (1- \alpha)$, nous permet d'avoir le bon compromis entre la densité et la mobilité des nœuds du réseau.

Le processus d'élection implique que chaque nœud informe ses voisins de la valeur de sa métrique. Un changement de cette valeur peut induire un changement de chef dont il faudra informer l'ensemble des membres du cluster. Cette notification sera d'autant plus longue que le diamètre (nombre de sauts du chef à son plus lointain membre) sera grand. Notre application à forte dynamique suggère donc que nous choisissions d'élire le chef dans un k-voisinage le plus petit possible. La métrique étant basée sur une 1-densité, les membres du cluster seront au pire à 2 sauts du chef.

3.4.3 Description de l'algorithme

3.4.3.1 Notations

Soit C(i) le chef du nœud i, P(i) le parent du nœud i (son voisin à un saut de meilleure métrique), L(i) le nombre de sauts entre i et son chef, M(i) la valeur de la métrique du nœud i et V(i) = {V1,V2,…Vn} la liste des voisins de i.

Initialement, un nœud *i* est à l'état ordinaire, tel que :

$$C(i) \leftarrow -1 \; ; \; P(i) \leftarrow -1 \; ; \; L(i) \leftarrow -1 \; ; \; M(i) \leftarrow 0 \; ; \; V(i) \leftarrow \emptyset$$

3.4.3.2 Diagramme des états finis

Nous pouvons caractériser le comportement des processus sur les nœuds capteurs en les assimilant à des automates à états finis au sein d'un système de communication par acheminement de messages. Autrement dit, chaque processus à un instant donné est dans un état déterminé (parmi un ensemble fini

d'états possibles : ordinaire, membre à 1-saut, membre à 2-sauts ou Cluster-Head). A la réception d'un message (franchissement d'une transition), ce processus réalise de façon séquentielle un certain nombre d'opérations primitives dont la nature dépend à la fois de son état actuel et du contenu du message reçu. Les opérations primitives possibles sont : des calculs locaux, la transmission de messages et le changement d'état. Le comportement d'un processus sur un nœud capteur peut être alors décrit par un schéma de transition du type :

<état (K), message reçu> → *<état (K+1), message envoyé, calcul>*

Ainsi, un algorithme distribué sur un réseau n'est autre que la caractérisation des types d'opérations à réaliser séquentiellement par un processus d'un nœud capteur du réseau à la réception d'un message, ce processus étant dans un état donné.

L'automate à états finis sur lequel est basé notre algorithme est représenté dans la Figure 3-4.

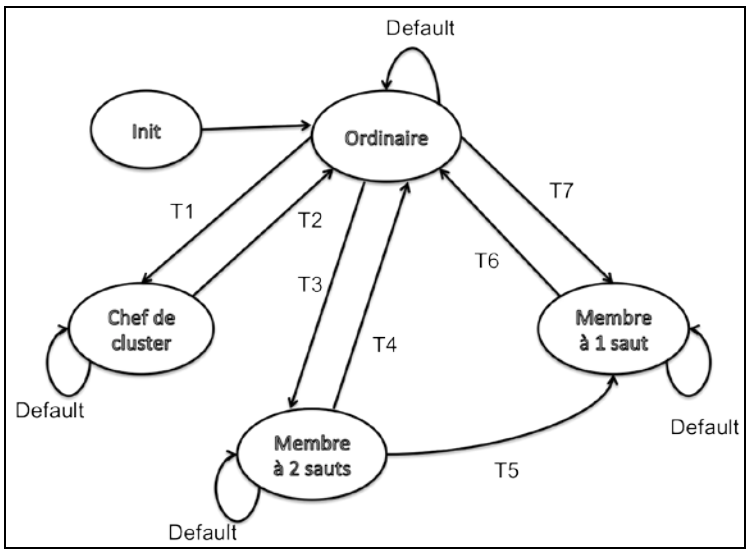


Figure 3-4 Automate à états finis

Le processus illustre les différents états que peut avoir un nœud du réseau (Tableau 3-1). Ainsi que les transitions qu'il doit franchir pour passer d'un état à un autre (Tableau 3-2).

Init	initialisation des états des variables
Ordinaire	le nœud n'adhère à aucun cluster
Chef de cluster	Le nœud se déclare chef de cluster
Membre à 1-saut	Nœud membre se trouvant à 1-saut de son chef de cluster.
Membre à 2-sauts	Nœud membre se trouvant à 2-sauts de son chef de cluster.

Tableau 3-1: Etats de l'automate à états finis

T1	Réception du message *HELLO* et $M(i) > M(V_v(i))$
T2	Réception du message *HELLO* et M(j) > M(i) et C(j) = j
T3	Réception du message *ADHESION* et C(j) ≠ i
T4	Réception du message *HELLO* et P(i) = j et L(j) ≠ 1
T5	Réception du message *HELLO* et C(i) = j
T6	Réception du message *HELLO* et C(i) = j et C(j) ≠ j
T7	Réception du message *INVITE*

Tableau 3-2 : Transitions de l'automate à états finis

L'algorithme de répartition proposé construit des arbres de clusters à *2*-sauts (la distance entre un nœud et son chef de cluster est d'au plus *2*-sauts). L'algorithme procède par itération à l'aide de messages spécifiques. Nous utilisons seulement trois types de messages de diffusion:

- *HELLO* : message envoyé périodiquement par chaque nœud du réseau. Ce message servira pour la détection du voisinage du nœud ;
- *INVITE* : message envoyé par le nœud chef de cluster pour inviter ses voisins à rejoindre son cluster ;
- *ADHÉSION* : message envoyé par un nœud pour notifier son adhésion au cluster et inviter aussi ses voisins à rejoindre le cluster ;

Le processus comporte 3 phases : une phase de découverte du voisinage, une phase de construction de la structure virtuelle et une phase de maintenance.

3.4.3.3 Découverte du voisinage

Pour prendre sa décision, chaque nœud du réseau doit connaître son voisinage à deux sauts. Le message *HELLO* est échangé périodiquement entre les voisins à un saut. Le paquet *HELLO* contient alors les informations suivantes :

- l'identifiant du nœud source ;
- les coordonnées (x, y) du nœud source ;
- le statut du nœud source ;
- l'identificateur du cluster auquel appartient le nœud source ;
- l'identificateur du nœud parent du nœud source ;
- l'adresse du cluster auquel appartient la source ;
- la densité calculée du nœud source ;
- la mobilité calculée du nœud source ;
- la métrique calculée du nœud source ;

- le nombre de sauts séparant le nœud source de son chef de cluster ;
- l'énergie consommée au niveau du nœud source ;
- nombre de voisins de la source ;
- la liste des adresses des voisins de la source ;
- la liste des adresses des clusters des voisins de la source.

En collectant les paquets *HELLO* reçus, chaque nœud maintient deux listes, une liste pour les voisins à un saut et une deuxième liste pour les voisins à deux sauts. Initialement, aucun nœud n'appartient à la structure des clusters et tous les nœuds sont à l'état ordinaire. Chaque nœud doit alors diffuser dans son voisinage à un saut le message *HELLO*. Ce message va permettre aux nœuds voisins à 1-saut de découvrir la présence du nœud émetteur et de l'ajouter dans leurs listes des voisins à 1-saut. Après le premier tour d'échange des messages *HELLO*, chaque nœud détecte tous ses liens directs. Un deuxième tour sera alors nécessaire pour permettre la découverte des voisins à 2-sauts.

Après échange du message *HELLO*, chaque nœud extrait les informations du message, calcule sa propre métrique et la compare à celle de ses voisins. Si sa métrique est la plus grande, alors il passe à l'état chef de cluster et lance la procédure de construction de son cluster. Sinon il attend un message *INVITE* provenant d'un chef de cluster voisin.

La Figure 3-5 illustre le processus de découverte de voisinage par message *HELLO*. Initialement, tous les deux nœuds (u) et (v) n'ont reçu aucun message *HELLO*. Les tables de voisinage Γ_u et Γ_v des nœuds (u) et (v) sont vides ($\Gamma_u = \phi$; $\Gamma_v = \phi$). Une fois que le nœud (v) reçoit le message *HELLO* diffusé par le nœud (u), il ajoute les informations concernant (u) dans sa table de voisinage ($\Gamma_u = \phi$; $\Gamma_v = \{u\}$). Quand le nœud (u) reçoit le message *HELLO* diffusé par le nœud (v), alors il met sa table à jour et ajoute le nœud (u) ($\Gamma_u = \{v\}$; $\Gamma_v = \{u\}$). De même, quand les nœuds (u) et (v) reçoivent un message *HELLO* du nœud (w), ils mettent leurs tables respectives à jour ($\Gamma_u = \{v, w\}$; $\Gamma_v = \{u, w\}$).

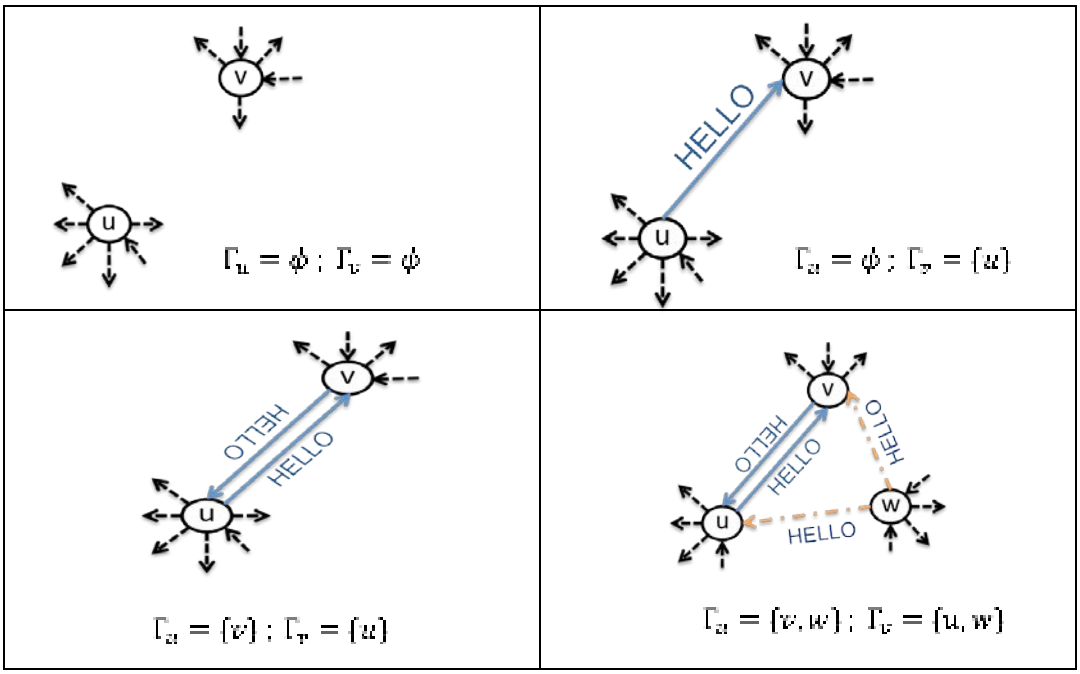


Figure 3-5. Schéma illustratif de découverte de voisinage par message *HELLO*

En fonction de la dynamique de notre système (conditions climatiques, vitesses des vents, mobilité des nœuds dans le pétrole,…etc.), et tout en souhaitant préserver la bande passante nécessaire à l'information mutuelle entre voisins, nous avons choisi de fixer à 7 secondes seconde la périodicité d'émission des messages *HELLO*.

L'algorithme d'émission du message *HELLO* est comme suit :

Algorithme 1 : Emission périodique du message hello

Pou*r chaque k seconde* ***faire***
Update (M(i)) ;
Diffusion (hello)

3.4.3.4 Maintenance de la liste du voisinage

L'échange périodique des messages *HELLO* permet aussi de détecter la perte de lien entre les voisins directs. En effet, chaque nœud associe un temporisateur à chaque voisin direct. Ce temporisateur est fixé à une valeur multiple de la période d'envoi d'un message *HELLO*. Si le temporisateur expire avant la réception d'un message *HELLO* du voisin auquel il est associé, le nœud peut alors considérer que le lien avec ce voisin est perdu. Alors, il l'enlève de sa liste des voisins et informe les autres voisins à travers l'envoi d'un message *HELLO*.

Algorithme 2: A l'expiration d'une entrée de voisinage

Pour chaque *k seconde* ***faire***
    Pour chaque $V_v \in V(i)$ ***faire***
        Si *dernière mise à jour de* V_v *ancienne de plus de 2 secondes* ***Alors***
            $V(i) \leftarrow V(i) - \{V_v\}$ ***;***
    Update (M(i)) ***;***
    Pour chaque *entrée de voisinage j retirée* ***faire***
        Suivant *état i* ***faire***
            Cas où *chef*
                Si $\exists v$ *tel que* $M(V_v(i)) > M(i)$ ***alors***
                $C(i) \leftarrow$ ***-1 ;*** $P(i) \leftarrow$ ***-1 ;*** $L(i) \leftarrow$ ***-1 ;***
            Cas où *membre* /* *à 1 ou à 2-sauts* */
Si *P(i) = j alors* $C(i) \leftarrow -1$; $P(i) \leftarrow -1$; $L(i) \leftarrow -1$;

3.4.3.5 Construction de la structure virtuelle

Chaque chef de cluster initie la construction de son cluster en diffusant à un saut un message *INVITE* servant à inviter ses voisins directs à adhérer à son cluster. Ensuite, il arme un temporisateur pour une retransmission de ce message si nécessaire. Si le temporisateur associé au message *INVITE* expire et que le chef ne reçoit aucune réponse d'acceptation de son entourage à 1-saut, il procède alors à la retransmission du message *INVITE*.

A la réception d'un message *HELLO*, le nœud met à jour sa table de voisinage et sa métrique. Selon l'état de nœud quatre cas se présentent :

- Si le nœud est à l'état ordinaire et si sa métrique est la meilleure par rapport à celles de ses voisins alors il se déclare chef de cluster, choisit le nœud émetteur du message comme parent, initialise le nombre de saut à zéro et diffuse le message *INVITE*.
- Si le nœud est à l'état chef de cluster et si sa métrique n'est pas la meilleure par rapport à celle du nœud émetteur alors il passe à l'état ordinaire. Sinon il diffuse le message *INVITE.* Si le nœud est à l'état membre et si l'émetteur est un autre chef que le sien alors il passe à l'état ordinaire et attend un nouveau message. Sinon il diffuse le message *ADHESION*.
- Si le nœud est à l'état membre à 2-sauts alors
    - Si l'émetteur est son chef de cluster alors il passe à l'état membre à 1-saut.
    - Si l'émetteur est son parent mais il n'est plus à 1-saut alors il passe à l'état ordinaire.
    - Si l'émetteur est son parent mais il a changé de cluster alors il change de cluster (le nœud suit son parent) et garde son état membre à 2-sauts.

Algorithme 3: A la réception d'un message (hello, invite ou adhésion)

```
A la réception sur un nœud i d'un message du voisin j;
si message type hello alors update (V(i)), udpate (M(i));
suivant état i faire
    cas où ordinaire
                /* C(i) = -1 */
      suivant type message faire
          cas où hello
              si M(i) >M(Vu(i)) ,∀v alors
              C(i) ← i; P (i) ← i; L(i) <-0; diffusion (invite);
          cas où invite
              C(i) ←j; P(i) ←j; L(i) ← 1; diffusion (adhésion);
          cas où adhésion
              si C(j) ≠ i alors C(i) ← C(j); P(i) ←j; L(i) ← 2;
    cas où chef                                                    /* C(i)
      = i et L(i) = 0 */
      suivant type message faire
          cas où hello
              si M(j) > M(i) et C(j)) = j alors
              C(i) ←-1; P(i) ← -1; L(i) ←-1;
              sinon si nouveau voisin alors diffusion (invite);
          cas où membre
                  /* P(i) = C(i) et L(i) =1 */
              suivant type message faire
                  cas où hello
                      si nouveau voisin alors diffusion (adhésion);
                      sinon si C(i) = j et C(j) ≠ j alors
                          C(i) ←-1 ; P(i) ←-1; L(i) ←-1
    cas où membre à 2-sauts                                        /* P(i)
      ≠ C(i) et L(i) =2 */
      suivant type message faire
          cas où hello
              si C(i) = j alors
                  C(i) ←j; P(i) ←j; L(i) ← 1; diffusion (adhésion);
              sinon si P(i) = j et L(j) ≠ 1 alors
                  C(i) ←-1; P(i) ←-1; L(i) ←-1;
              sinon si P(i) = j et C(j) ≠ C(i) alors
                  C(i) ←C(j); P(i) ←j; L(i) ← 2;
          cas où invite
              si C(i) = j alors
C(i) ←j; P(i) ←j; L(i) ← 1; diffusion (adhésion);
```

A la réception d'un message *INVITE*, deux cas se présentent selon l'état du nœud :

- Si le nœud est à l'état ordinaire alors il choisit l'émetteur du message comme parent, passe à l'état membre et diffuse un message *ADHESION* à un saut pour avertir d'une part le chef de

cluster de son adhésion au cluster et d'autre part pour inviter ses voisins à le rejoindre dans le cluster s'ils ne sont pas déjà intégrés.

- Si le nœud est à l'état membre ou chef de cluster alors il ignore le message reçu. (On privilégie la stabilité dans l'organisation à sa performance)
- Si le nœud est à l'état membre à 2-sauts et si l'émetteur est son chef de cluster alors il passe à l'état membre à un saut. Et diffuse le message *ADHESION*.

A la réception d'un message *ADHESION*, si le nœud est à l'état ordinaire et qu'il n'était pas le chef du nœud émetteur alors il passe à l'état membre à 2-sauts. Sinon il ignore le message reçu.

3.4.3.6 Maintenance de la structure de clusters

La phase de maintenance de la structure virtuelle formée est assez simple. Celle-ci n'est basée que sur l'élection de nouveaux parents ou d'un nouveau chef de cluster. Ce processus est déclenché pour les nœuds ayant perdu leur parent qui n'est plus attaché à un chef de cluster dans l'arbre de clustering à cause soit de l'épuisement de leur batterie soit de leur mobilité.

Nous avons opté pour cette stratégie de remise en cause de la structure virtuelle pour favoriser une maintenance rapide permettant ainsi de prévenir ou d'éviter les réactions de reconstruction en chaîne de la structure.

Le processus de maintenance ne s'appuie que sur le message *HELLO* utilisé pour la découverte de voisinage. Dans le processus de maintenance, chaque nœud vérifie ainsi périodiquement le poids de ses voisins toujours attachés à un chef de cluster. Ensuite, il s'attache à son voisin de plus fort poids.

3.4.4 Conclusion

Dans cette section, nous avons présenté un descriptif détaillé de notre algorithme de clustering adapté à la mobilité des nœuds capteurs. Cet algorithme exploite le mécanisme d'élection utilisant une métrique composée de l'énergie résiduelle, la densité et la mobilité. Ces derniers éléments sont pondérés par un facteur (α) que nous allons faire varier durant les simulations futures pour trouver un optimum assurant une stabilité de formation des clusters. Ces simulations sont réalisées à l'aide d'un modèle de nœud capteur que nous avons développé et qui fera l'objet de la section suivante.

3.5 Modèle du nœud capteur sous Opnet (notre 2ème contribution)

Après la phase de la spécification formelle de l'algorithme de clustering et sa représentation sous format algorithmique, vient la phase implémentation et de test de son bon fonctionnement et l'évaluation de ses performances. La section suivante décrira les différentes méthodes d'évaluation, souvent utilisées, et justifiera notre choix de la méthode par simulation, ainsi que le choix du simulateur sur lequel nous développerons notre modèle de nœud capteur.

3.5.1 Choix de la méthode d'évaluation.

D'une manière générale, pour tester le bon fonctionnement d'un processus et évaluer ses performances, on utilise trois méthodes d'évaluation : la méthode analytique, la méthode par simulation et la méthode par mesure sur un système réel existant. Ces trois méthodes diffèrent par le coût, la durée et l'exactitude des résultats. (Erard et Déguénon, 1996) :

- La méthode analytique consiste à représenter le système par un modèle mathématique et à faire son analyse numérique. Cette approche est certes rapide à réaliser une fois que le modèle mathématique est trouvé, mais présente le problème de la reproduction fidèle du système qui peut être très complexe. Il est très difficile de modéliser le comportement réel du système mathématiquement.

Souvent et par souci de simplification, on pose des hypothèses dans la phase de modélisation ce qui induit une mauvaise représentativité du système réel.

- Quant à la technique de mesure, celle-ci consiste à faire des mesures et à les analyser directement sur un système réel. Cette technique permet de bien comprendre le vrai comportement du système. Cependant, faire des mesures sur des systèmes réels n'est pas toujours possible, car cela peut perturber le fonctionnement du système. De même cette méthode n'est plus possible dans le cas où le système est non encore existant.

- En ce qui concerne la méthode de simulation, il s'agit d'implanter un modèle du système à l'aide d'un programme de simulation adéquat. C'est une technique largement utilisée pour l'évaluation des performances. Elle présente l'avantage par rapport aux méthodes analytiques de traduire d'une manière plus réaliste le comportement du système à évaluer. On recourt généralement à la simulation pour évaluer un nouveau système ou bien pour des raisons de coûts d'évaluation par des mesures réelles. En plus, la simulation permet de visualiser les résultats sous forme de graphes faciles à analyser et à interpréter. Et c'est pour ces raisons que nous avons décidé d'utiliser la simulation à événements discrets pour tester le bon fonctionnement et évaluer les performances de notre algorithme de clustering et par la suite notre protocole de session. Reste à choisir quel simulateur utiliser ? Pour répondre à cette question, nous avons fait une étude brève des différents simulateurs utilisés dans la communauté de recherche réseau.

3.5.2 Simulateur choisi

Pour développer, tester et valider ces algorithmes, les chercheurs se sont très rapidement tournés vers les simulateurs pour des raisons pratiques. Il est peu commode de déployer un protocole sur un réseau réel de grande taille, de faire une série de tests, puis les reprendre en changeant des paramètres ou des morceaux de code. De plus, peu de laboratoires disposent de plateformes matérielles nécessaire aux tests (problème de coût et de logistique).

Dans le domaine de la recherche, plusieurs logiciels sont utilisés pour simuler un réseau et évaluer ses performances. Ainsi, un certain nombre de simulateurs ont été développés conjointement. Ces simulateurs sont mis à contribution par les chercheurs désireux d'évaluer les performances des protocoles réseaux qu'ils développent.

Nous pouvons citer par exemple, NS2 (Network Simulator 2)[5], OPNET Modeler (*Optimum NETwork Performance*)[6] ou encore GloMoSim[7] / Qualnet[8]. NS2 est un simulateur à événements discrets développé dans un but de recherche. Donc, c'est un outil gratuit et son code source est disponible (open source). Mais sa librairie ne modélise pas tous les protocoles réseau et ne dispose pas d'interface graphique. Alors que OPNET Modeler dispose d'un environnement graphique.

OPNET Modeler est un simulateur à événements discrets très puissant pour la simulation et l'évaluation des performances de réseau. Sa bibliothèque est riche en protocoles de toutes les couches du modèle OSI (*Open System Interconnection*). OPNET Modeler possède trois niveaux d'abstraction pour construire les modèles. Les trois niveaux d'abstraction sont : le réseau, les nœuds et les processus. Le réseau est composé de plusieurs nœuds, le nœud est composé de plusieurs processus et le processus est décrit avec un automate et du code en C. Le plus grand avantage qu'offre OPNET est qu'il permet

[5] http://www.isi.edu/nsnam/ns/

[6] http://www.opnet.com/

[7] http://pcl.cs.ucla.edu/projects/glomosim/

[8] http://www.qualnet.com/

à l'utilisateur de construire ses propres modèles des plus simples aux plus complexes.

Notons que OPNET Modeler, destiné à ses débuts aux besoins militaire, fait actuellement partie de la famille des logiciels développés par la société MIL3 et est commercialisé depuis 1986.

En plus des avantages qu'offre OPNET Modeler, cités plus haut, nous avons choisi d'utiliser ce simulateur spécialement, pour développer notre modèle de nœud, pour les raisons suivantes :

- Opnet permet de modéliser le fonctionnement d'un réseau durant la phase de conception d'une manière modulaire.
- l'interface graphique et la modélisation orientée objet permettent de reproduire la structure réelle du réseau et de ses composants afin de coller à la réalité de façon intuitive.
- l'utilisation du simulateur nous permet de détecter les problèmes qui surviendront en exploitation dès la conception du réseau, et donc de pouvoir tester les différentes solutions permettant d'y remédier.
- le modèle de propagation radio utilisé par le simulateur OPNET Modeler est plus proche de la réalité car il est basé sur le calcul du Signal To Noise Ratio (SNR).
- le laboratoire CRAN dispose d'une licence académique et possède de très bonnes compétences accumulées sur plusieurs années.

3.5.3 Description du modèle du nœud capteur

Après avoir parcouru les différents types de nœuds existants dans la bibliothèque du simulateur OPNET Modeler et n'ayant pas trouvé de modèle de nœud répondant à nos attentes (code source verrouillé du modèle natif Opnet Zigbee), nous avons décidé de développer un modèle de nœud capteur. Ce modèle nous servira à évaluer notre algorithme de clustering et le protocole de session proposés. Comme le montre la Figure 3-6, le modèle du nœud proposé est constitué de quatre couches :

3.5.3.1 Couche ou processus physique

La couche physique implémente les modules émetteur radio (Tx) et récepteur (Rx) conformes aux spécifications du standard IEEE 802.15.4. Ces modules fonctionnent avec une bande de fréquence de 2,4 GHz et un taux de données de 250 Ko/s. La puissance de transmission est de 1mW avec une portée de 50m. La technique de modulation utilisée est le QPSK (*Quadrature Phase Shift Keying*).

Le modèle de propagation utilisé est le *Free Space Model*. Ce modèle considère qu'il n'existe qu'un seul chemin de propagation entre l'émetteur et le récepteur et qu'il est en vue directe. L'équation utilisée pour calculer la puissance du signal reçu en environnement libre à une distance d de l'émetteur est la suivante :

$$P_r(t) = \frac{P_t G_t G_r \lambda^2}{4\pi^2 d^2 L}$$

Où P_t est la puissance d'émission, G_t et G_r sont les gains respectifs de l'antenne émettrice et l'antenne réceptrice. L est la perte du système et λ la longueur d'onde.

Ce modèle de propagation représente les zones de communication comme un cercle de rayon *r* autour de l'émetteur. Ainsi un nœud ne peut recevoir les paquets que s'il se trouve au sein du cercle.

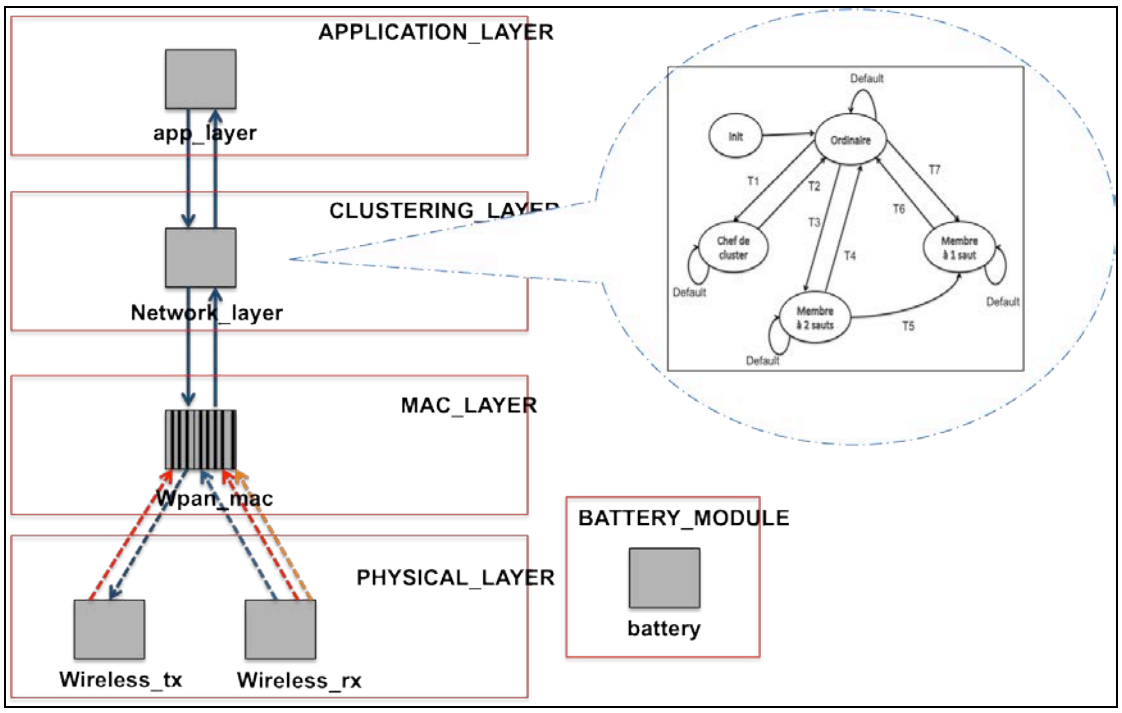


Figure 3-6. Modèle du nœud capteur sous Opnet

3.5.3.2 Couche MAC

Cette couche implémente le processus de la couche MAC [24] non-slottée, non-beaconée, reposant sur le standard IEEE802.15.4. Dans ce mode, le nœud voulant transmettre des données, teste si le canal est libre. Si c'est le cas, alors il transmet sinon il attend une période aléatoire définie dans le protocole IEEE 802.15.4-2003.

L'algorithme de CSMA/CA non-slotté (*Carrier Sense Multiple Access with Collision Detection*) est appliqué pour les envois de trames dans un réseau en mode non suivi de la trame beacon. Pour autant, il ne s'agit pas de CSMA/CA de la norme IEEE 802.11. Le paramètre CW (*Contention Window*) n'existe pas dans l'algorithme de CSMA/CA non-slotté : il suffit de détecter une seule fois que le canal est libre pour commencer une transmission. L'unité de temps reste la période de backoff, mais l'activité des entités n'est pas synchronisée sur ces périodes de backoff. L'algorithme commence par la phase d'initialisation en mettant BE = macMinBE (*Backoff Exponent*) et NB = 0 (*Number of Backoffs*). Puis, la machine tire aléatoirement un nombre entier de périodes de backoff. Après la consommation du backoff, la couche MAC demande à la couche physique d'effectuer un CCA (*Clear Channel Assessment*). Si la couche physique détecte que le canal est libre alors la couche MAC commence la transmission. Si ce n'est pas le cas, la couche MAC incrémente NB et BE (à condition que BE reste inférieur ou égal à macMaxBE).

Ce processus permet d'optimiser l'autonomie des batteries des capteurs (les nœuds capteurs dorment dans la majorité du temps et ne se réveillent que si un évènement survient dans la zone de captage). Ainsi, le canal n'est utilisé que durant la transmission des données utiles. Par contre, du fait de CSMA/CA, l'accès au canal n'est pas garanti dans une période donnée.

3.5.3.3 Couche clustering

Cette couche implémente notre algorithme de clustering proposé. Comme mentionné plus haut (voir 3.4), l'algorithme exploite trois types de messages :

Le message *HELLO* utilisé dans la découverte de voisinage, l'élection du chef de cluster et la maintenance du cluster.

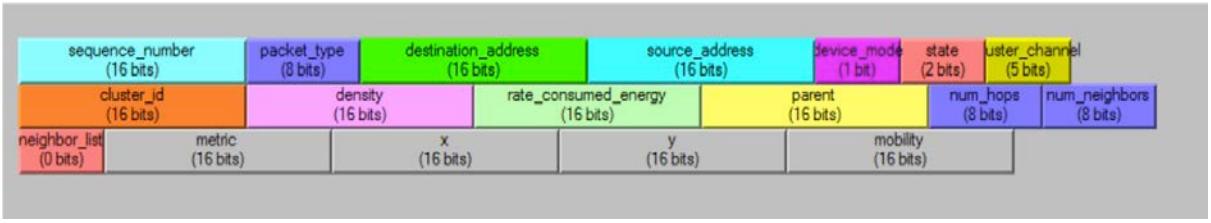


Figure 3-7. Format du paquet *HELLO*

Le message *INVITE* utilisé par le chef de cluster pour inviter les autres nœuds à le rejoindre. Ce message a une taille de 20 octets. Il contient des informations concernant le chef de cluster : son identifiant qui représente aussi l'identificateur du cluster, son état, sa métrique, ses cordonnées et sa mobilité.

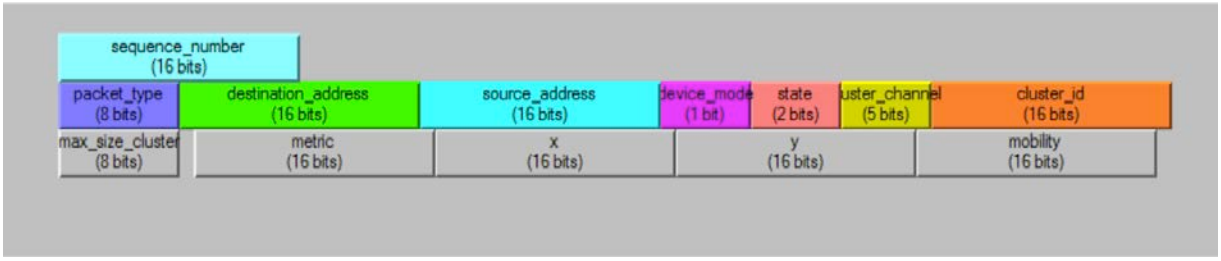


Figure 3-8. Format du paquet *INVITE*

Le message *ADHESION (*Figure 3-9*)* utilisé par les nœuds pour déclarer leur adhésion au cluster et inviter les autres nœuds non encore affiliés à le rejoindre dans son nouveau cluster. Ce message a une taille de 25 octets. Il contient des informations concernant le nœud membre émetteur : l'identifiant du nœud, l'identificateur du cluster auquel il est affilié, son état, sa métrique, ses cordonnées et sa mobilité.

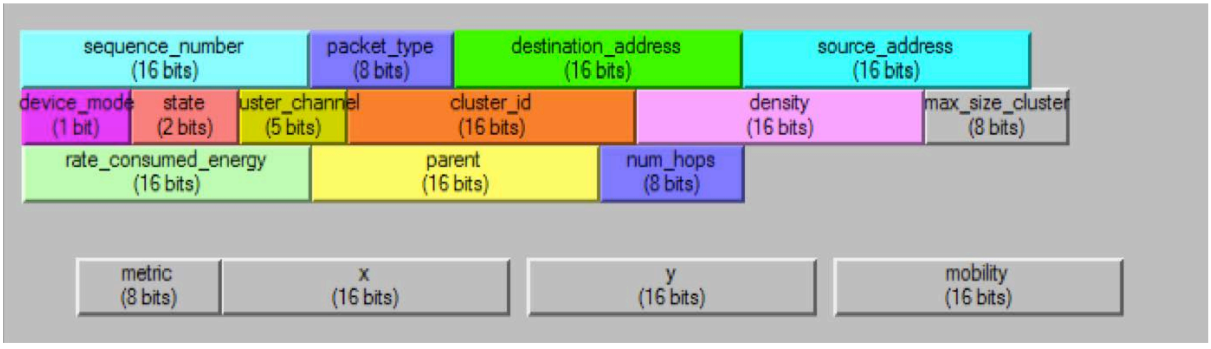


Figure 3-9. Format du paquet *ADHESION*

La mise en place de notre protocole a nécessité le développement d'un ensemble de fonctions codées en ProC (langage de programmation d'OPNET Modeler). Le tableau 1 décrivant l'ensemble de ces fonctions développées est placé à l'annexe 1.

3.5.3.4 Couche application

Cette couche implémente le processus qui génère le trafic applicatif dans le réseau. Selon son état, le nœud capteur exécutera l'un des processus suivants :

Processus 1 : Transmission périodique des données captées à son chef de cluster. Ce processus est exécuté par le nœud s'il est dans l'état membre à 1-saut ou à 2-sauts.

Processus 2 : Transmission des données reçues des nœuds membres du cluster au collecteur lors de sa visite. Ce processus n'est exécuté par le nœud que s'il est dans l'état chef de cluster.

Algorithme 4 : A la réception d'un message sur un nœud i au niveau de la couche application.

Si *état i = chef de cluster /* C(i) = i /* **alors** /* Exécution du processus 1 */*
Extraction des valeurs champs du message (type du message,@source,@destination) ;
Si *@source= @ du collecteur* **alors** ***/** message provenant du collecteur */*
#Création du message de réponse #
Encapsulation (type_pkt , @destination=@collecteur, donnée, …etc.);
Envoi du message au collecteur ;
Sinon / *message provenant d'un nœud membre */*
Extraction des valeurs des différents champs du message (type du message, @origine, data_cap) ;
Stockage de la donnée récupérée dans la liste des données List_data_cap
Destruction du message reçu ;
Finsi
Sinon ***/*** *état i ≠ chef de cluster, C(i) ≠ i */*
Création du message de données destiné au chef de cluster ;
Encapsulation (@destination=@chef du cluster, donnée,…etc.) ;
Envoi périodique du message de données vers le chef de cluster ;
Destruction du message envoyé.
Finsi

3.5.3.5 Module batterie

Le module batterie utilisé, calcule l'énergie consommée et les niveaux d'énergie résiduelle restants pendant les périodes actives et inactives des capteurs. Les valeurs par défaut du courant sont conformes à ceux de la spécification des capteurs MICAZ[9].

La consommation énergétique est une métrique très importante dans les réseaux de capteurs sans fil. Dans (Klein et Tran-Gia, 2007), les auteurs ont réalisé une analyse avancée de la consommation d'énergie. Le modèle de batterie est utilisé pour la collection de statistiques de la consommation d'énergie par le nœud de capteur sans fil. Ce modèle Open-ZB disponible dans (Jurčík et Koubâa, 2007) est pris comme modèle d'étude. Son modèle de processus d'états est plutôt simple (Figure 3-10). Après l'initialisation, la procédure de dissipation de puissance est appelée. Une interface ICI (*Interface Control Information*) est utilisée pour fournir les informations sur les paquets transmis et reçus. La quantité d'énergie utilisée est calculée en fonction de la longueur des paquets et la direction du flux.

[9] http://www.xbow.com/

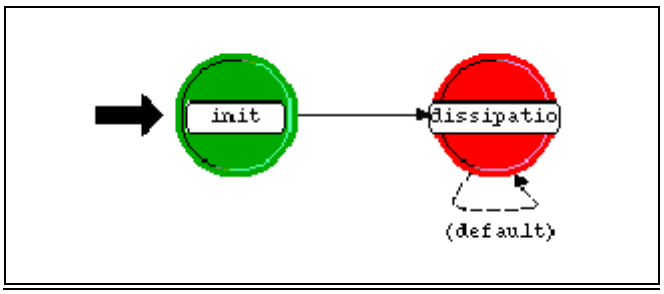


Figure 3-10. Processus du modèle de la batterie

Les paramètres de consommation énergétique sont choisis de la spécification de petite tâche MICAZ. Dans la Figure 3-11, on montre les paramètres de modèle de batterie. Il est possible de mettre les niveaux d'énergie exigés du mode réception, du mode transmission, l'état inoccupé et la consommation énergétique en mode sommeil. La quantité d'énergie initiale est mise en Joules ou correspondant à la capacité standard de batteries alcalines ou rechargeables.

Battery	
Current Draw	(...)
Receive Mode (mA)	MICAz default
Transmission Mode (mA)	MICAz (0 dBm)
Idle Mode (µA)	MICAz default
Sleep Mode (µA)	MICAz default
Initial Energy	200
Power Supply	2 AA Batteries (3V)

Figure 3-11. Paramètres du modèle de la batterie

3.5.4 Simulation

Cette section présente des résultats de simulation obtenus avec l'utilisation de notre modèle de nœud capteur détaillé dans la section précédente. L'objectif de la simulation est de souligner l'impact du paramètre (α) de la métrique sur la qualité de partitionnement du réseau. Cette qualité est évaluée en termes de nombre de clusters construits, de nombre de mises à jour des chefs de clusters pendant une simulation et enfin de durée de vie du statut de chef de cluster sur un nœud.

3.5.4.1 Environnement de simulation.

Les paramètres considérés durant la simulation sont le facteur de pondération (α) et la vitesse du déplacement des nœuds variant de 0,1m/s à 6m/s. Ces valeurs de vitesse choisies correspondent à des conditions climatiques réalistes. Par exemple, une vitesse d'un nœud égale à 0,2m/s correspond à une vitesse du vent de 40Km/h.

Le scénario simulé correspond au déploiement aléatoire de 100 nœuds capteurs sur une superficie de 1 km^2. Les capteurs ont été placés au début de la simulation sur un carré de 50 m^2. Les nœuds sont configurés à une portée radio maximale de 50 m.

Le modèle de mobilité que nous avons choisi est le *Random Waypoint*, utilisé pour la première fois dans (Johnson et Maltz, 1996) puis raffiné dans (Broch et al., 1998). Ce modèle est certainement le plus utilisé. Selon ce modèle, après son déplacement, le nœud mobile s'immobilise pour un temps fini, choisit aléatoirement une destination, puis se déplace à nouveau de la même manière que la première fois, et cela jusqu'à la fin de la simulation.

Notons que ce modèle a été sujet à quelques critiques comme c'est le cas dans (Yoon et al., 2003), où les auteurs mettent en lumière l'instabilité et la diminution progressive de la vitesse moyenne des mobiles en fonction du temps de simulation. Comme la destination et la vitesse sont choisies aléatoirement, des nœuds capteurs mobiles vont parfois choisir des destinations lointaines et des vitesses très lentes, et il leur faudra beaucoup de temps pour y arriver. Plus la durée de simulation est grande, plus le nombre de mobiles bloqués dans cette situation augmente, et la proportion de mobiles "lents" va croître de façon gênante.

Quoiqu'il en soit, il est clair que pour pouvoir construire des modèles de mobilité vraiment pertinents et réalistes, il faudrait idéalement pouvoir disposer d'informations sur le contexte dans lequel le réseau sera réellement déployé, donc disposer d'une plateforme réelle de test. Ce qui n'est pas le cas pour nous. (Disperser du pétrole en mer et faire des séries de mesure pour essayer de trouver un modèle de mobilité des nœuds réaliste !!!). Pour ces raisons, nous avons opté pour le choix de ce modèle de mobilité au niveau de nos nœuds capteurs mobiles.

Sur un nœud, le processus de découverte des voisins est périodiquement exécuté chaque seconde. La densité et la mobilité utilisée dans la métrique seront calculées sur durée de 7s. Pour faire face aux communications intermittentes, la métrique proposée considère la liste entière des messages reçus sur une fenêtre temporelle de $\Delta(t)$ (voir relation de la métrique) ce qui permet de limiter l'impact des messages perdus. En outre, un nœud (u) ne sera considéré comme voisin d'un autre nœud (v), que si le nœud (v) n'a pas reçu de message HELLO du nœud (u) après 7s.

3.5.5 Résultats

Un algorithme de clustering est considéré efficace si les chefs de clusters maintiennent leur statut de chef de cluster le plus longtemps possible. Du fait que l'occurrence des changements topologiques dus à la mobilité des nœuds cause une restructuration locale ou générale du réseau. Donc, pour évaluer la robustesse de notre algorithme face à la mobilité des nœuds, nous proposons d'estimer :

- le nombre de clusters formés ;
- le nombre moyen de changements des chefs de clusters ;
- la valeur moyenne du temps qu'un nœud passe dans l'état chef de cluster en fonction du facteur de pondération (α) et de la mobilité.

Des séries de simulations ont été exécutées avec des valeurs de *random number seed* «graine» différentes (900 simulations basées sur 30 seeds). Les valeurs du facteur de pondération (α) varient de 0 à 1 avec un pas de 0,1. Les valeurs de mobilité varient de 0,1m/s à 6m/s. Nous avons pu évaluer les performances de la métrique en terme de stabilité de clustering. Les valeurs moyennes sont calculées avec un intervalle de confiance de 95 %.

3.5.5.1 Nombre de clusters formés

En premier lieu, nous avons relevé l'évolution du nombre de clusters formés en fonction du temps, exprimé en secondes, durant la simulation. Cette information nous permet en effet d'analyser l'évolution de la dynamique du réseau en fonction du temps. Ainsi, la Figure 3-12 montre l'évolution du nombre de clusters formés pour une valeur donnée de (α) = 0.5 et une vitesse de nœuds constante de 2m/s. Cette évaluation est effectuée chaque seconde durant une simulation de 2 min.

Sur la Figure 3-12, nous pouvons voir qu'initialement, les capteurs sont proches les uns des autres et donc la densité des nœuds est forte. Le graphe est connexe. Et un ou deux clusters formés pourraient être suffisants. Ensuite, les capteurs commencent à se déplacer et s'éloigner les uns des autres : La densité du graphe diminue et des sous-réseaux apparaissent et donc plus de clusters sont formés.

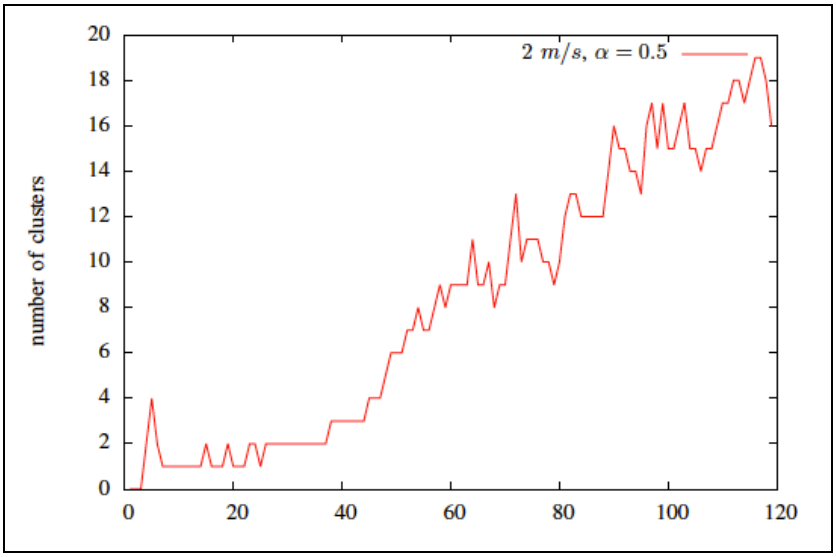


Figure 3-12. Nombre de clusters formés en fonction du temps (en secondes) durant la simulation

Sur la Figure 3-13, nous pouvons relever l'effet du facteur de pondération (α), pour différentes valeurs de vitesse des nœuds, sur le nombre moyen de clusters formés. Et ainsi, montrer l'intérêt d'adapter le facteur (α) à la mobilité pour minimiser le nombre de clusters formés.

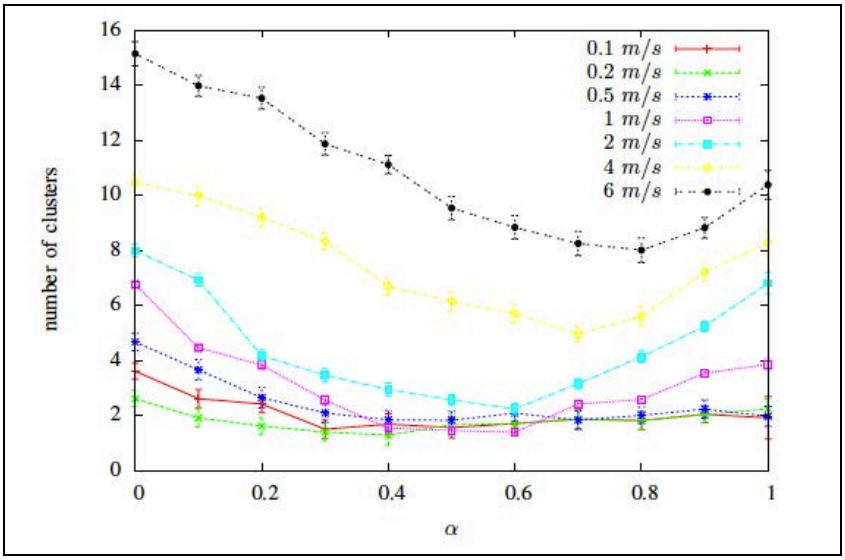


Figure 3-13. Nombre moyen de clusters formés en fonction de (α) et la mobilité

Les résultats mis en évidence sur la Figure 3-13, montrent que la forte mobilité peut être négligée par rapport à la densité. Mais toutefois, pour une faible et moyenne mobilité, le compromis est intéressant parce qu'il réduit le nombre des clusters formés sans affecter la stabilité (Figure 3-14 et Figure 3-15). En effet, pour des valeurs de mobilité inférieures à 2m/s, le nombre moyen de clusters formés est presque le même pour (α) = 0.3 à (α) = 1.

3.5.5.2 Nombre moyen de changements des chefs de clusters

La Figure 3-14 montre pour des vitesses différentes, la valeur moyenne du nombre de fois qu'un nœud passe à l'état chef de cluster en fonction du facteur de pondération (α). Cette valeur reflète la probabilité qu'un nœud puisse avoir la meilleure métrique par rapport à ses voisins directs. C'est à dire

la probabilité d'être élu chef de cluster.

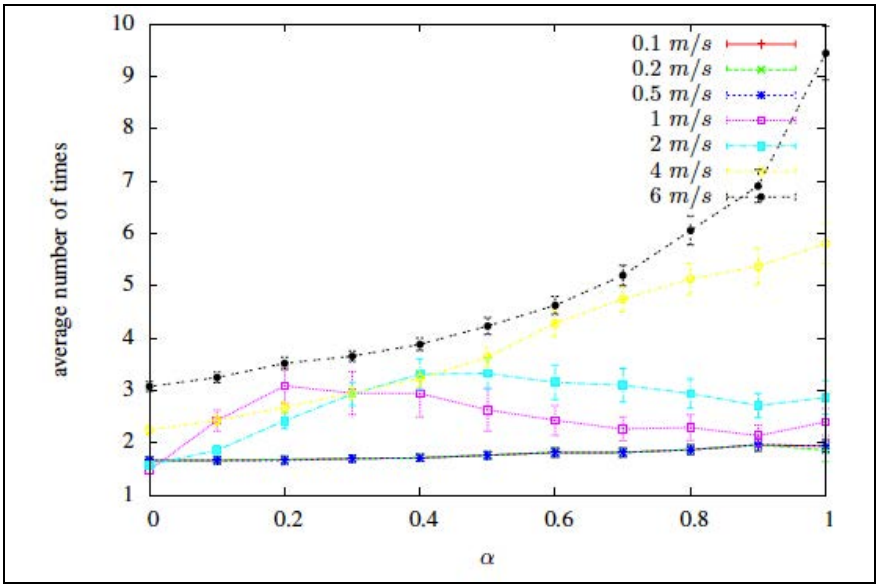


Figure 3-14. Nombre moyen de changement des chefs de clusters en fonction de (α) et de la mobilité

La Figure 3-14 montre qu'il y a une réduction du nombre moyen de changement des chefs de clusters pour (α) = 1 à des valeurs de mobilité inférieures à 2m/s. Tandis que pour les fortes valeurs de mobilité (supérieure à 4m/s), il y est plus intéressant de choisir de faibles valeurs de (α).

3.5.5.3 Valeur moyenne du temps qu'un nœud passe en tant que chef de cluster.

Pour avoir une idée sur la persistance des clusters formés, nous avons relevé la valeur moyenne du temps qu'un nœud passe en tant que chef de cluster (durée de vie d'un chef de cluster). Cette valeur nous donne une vue sur la stabilité qu'offre notre métrique et l'algorithme qui l'exploite.

La Figure 3-15 confirme les résultats obtenus sur la Figure 3-13. En effet, pour une mobilité faible, cette figure montre qu'il est meilleur d'avoir (α) = 1 pour qu'un nœud maintienne son statut de chef de cluster le plus longtemps possible. Cela signifie que plus la mobilité des nœuds est grande, plus la stabilité des clusters formés est affectée.

D'autres simulations sont ensuite lancées pour illustrer l'impact du poids (α). Des simulations plus longues ont été lancées pour visualiser la consommation d'énergie du nœud. Par exemple, sur une durée de simulation de 10mn, nous avons constaté que seulement 0.0016 % d'énergie ont été consommés.

Dans la section suivante, nous présentons un module que nous avons développé sous Opnet et en Java. Ce module nous permettra de prendre des « photos » de manière périodique. Ainsi grâce à son interface graphique, nous pouvons suivre l'évolution du clustering.

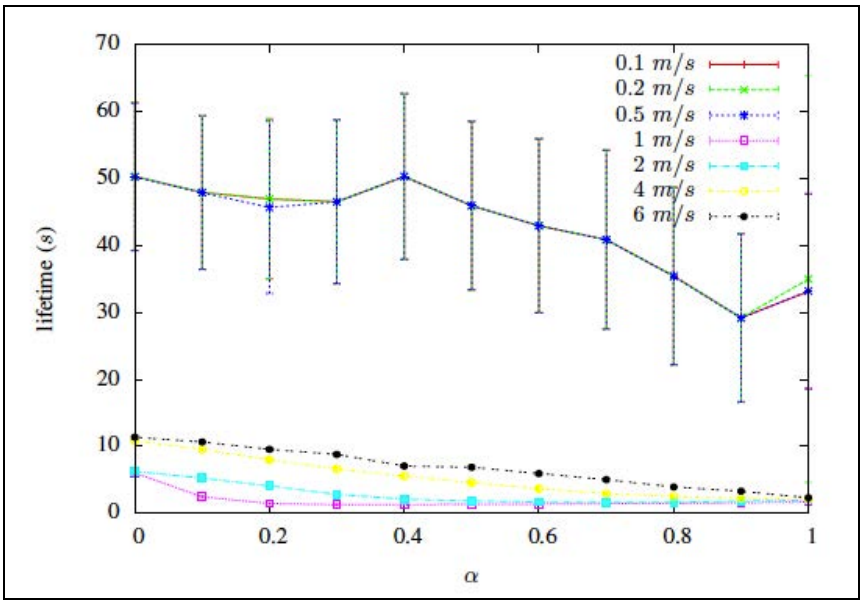
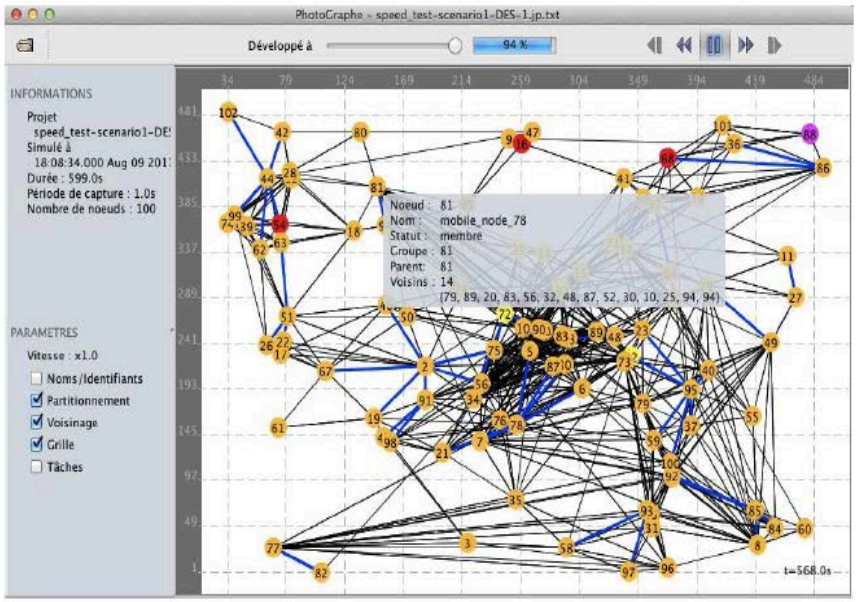


Figure 3-15. Valeur moyenne du temps qu'un nœud passe dans l'état chef de cluster en fonction de (α) et de la mobilité

3.5.6 Module Photographe

Cette section présente le module Photographe constitué du nœud « Photographe » développé pour Opnet, et un programme additionnel « PhotoGraphe » développé en Java. Le nœud « Photographe » permet de surveiller l'état des nœuds par rapport à l'algorithme de clustering. Le programme additionnel développé en Java permet de rejouer la simulation en observant les sous-graphes constitués par l'algorithme de clustering tout au long de la simulation.



Nœud« Photographe » Programme « PhotoGraphe »

Figure 3-16. Module Photographe

3.5.6.1 Nœud « Photographe »

Le fonctionnement de ce nœud consiste à l'enregistrement (vues du graphe du réseau) périodique d'un certain nombre de paramètres concernant l'algorithme de clustering. Parmi les informations stockées (à la même fréquence), nous trouvons les nœuds voisins de chaque nœud. L'exécution du processus implémenté dans le nœud est faite durant toute la simulation. Les informations récupérées sont enregistrées dans un fichier texte.

Pour bénéficier de cet additif de surveillance de l'état de clustering, nous ajoutons au projet Opnet le nœud Photographe.

3.5.6.2 Eléments du fonctionnement

Le processus implémenté dans le nœud Photographe fonctionne suivant deux états comme indiqués à la Figure 3-17 :

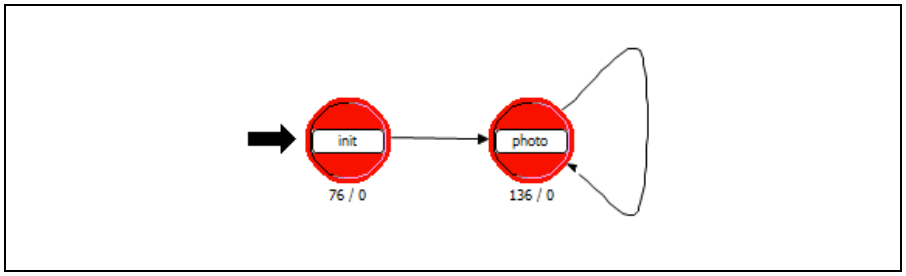


Figure 3-17. Automate à états finis du processus Photographe

- Etat init : permet de récupérer l'ensemble des attributs, de préparer le fichier de sortie et de traiter l'éventuelle inhibition de la fonction photographie.

- Etat photo : permet de sonder tous les nœuds du réseau (automatiquement découverts) et enregistre les informations associées. L'ensemble des paramètres est capturé au même instant de la simulation. L'interruption de fin de simulation y est également capturée afin de fermer proprement le fichier de sauvegarde des informations.

L'annexe A donne les éléments de programmation correspondant au développement du nœud photographe.

3.5.6.3 Programme « PhotoGraphe » développé en Java

« PhotoGraphe » est un programme Java (JDK 1.6+). Ce programme permet de rejouer la simulation en observant les sous-graphes constitués par l'algorithme de clustering tout au long de la simulation. L'interface graphique de ce programme est représentée dans la Figure 3-16.

Dans la zone de représentation du graphe (et sous graphes) sont représentés :

- Les nœuds par un cercle de couleur dépendante de son état (Jaune pour l'état ordinaire, Orange pour l'état membre, violet pour l'état membre à 2-sauts et Rouge pour l'état chef de cluster). Au centre de chaque cercle figure l'identifiant (l'adresse MAC) du nœud. (Figure 3-18 (a))

- Les relations de parenté, colorées en bleu, (Figure 3-18 (b))

- Les relations de voisinage, colorées en noir, (Figure 3-18 (c))

- Une représentation estimative de la nappe de pétrole. (paramètre d'ordre gadget). Comme l'illustre la Figure 3-18 (d). Cette estimation correspond à des ellipses d'une certaine dimension autour de chaque nœud.

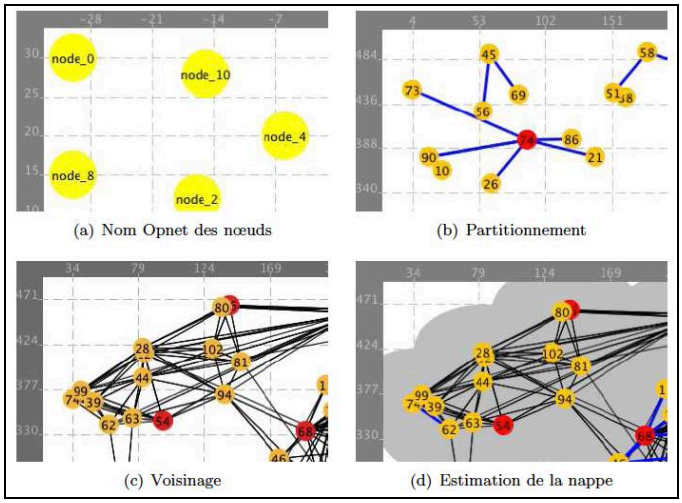


Figure 3-18. Paramètres d'affichage

3.6 Conclusion

Dans ce chapitre, après avoir fait une revue de la bibliographie des algorithmes de clustering et plus précisément de ceux qui traitent la mobilité dans les réseaux de capteurs, nous avons présenté nos deux propositions : l'algorithme de clustering et le modèle de nœud capteur. L'algorithme de clustering est basé sur un mécanisme d'élection. L'élection est réalisée selon notre métrique composée de l'énergie résiduelle, la densité et la mobilité. Nous avons montré, à travers des simulations sur OPNET Modeler, que notre algorithme assure une stabilité des clusters formés. Durant ces simulations, nous avons essayé de trouver les valeurs optimales du facteur de pondération (α) figurant dans notre métrique. Des valeurs qui minimisent le nombre de clusters formés et augmentent le temps qu'un nœud passe dans l'état chef de cluster, ce qui assure une stabilité satisfaisante des clusters. En outre, la mobilité des nœuds capteurs permet d'éviter l'élection des chefs de clusters pour une longue durée afin de ne pas les pénaliser en consommation d'énergie. Ces simulations ont été faites à l'aide du modèle de nœud capteur que nous avons développé sous l'environnement de simulation OPNET. En cas de désastre, le modèle de nœud proposé permet de concevoir le système de monitoring à déployer en termes de nombre de nœuds, de période de mesure, de dynamique du clustering et de poids de la métrique en prenant en compte la surface de zone à surveiller et les conditions climatiques. Notons que nous projetons de soumettre le modèle de nœud capteur développé à *OPNET Technologies, Inc*. Ainsi, le nœud sera mis à la disposition de la communauté scientifique.

A ce stade, notre réseau est auto-organisé en un nombre de clusters stables. La phase suivante est de mettre en place notre protocole de session que nous proposons dans le chapitre suivant. Nous allons nous concentrer sur le niveau 2 de l'architecture de notre système de supervision dédié à la gestion du collecteur mobile pour lequel nous décrirons notre protocole de session qui permet de récupérer le contexte des visites précédentes du collecteur mobile. Le but est de récupérer une connectivité discrète pour reconstruire périodiquement un système d'information cohérent.

Chapitre 4 Protocole de session

Ce chapitre est divisé en trois parties :

- *La première partie est consacrée à la définition de la notion de session. Nous essayons dans ce chapitre de définir les attributs de session et de dénombrer les indicateurs de sessions que nous utilisons dans la mise en place de notre protocole. Nous présentons également les notations utilisées.*

- *Les détails de notre proposition font l'objet de la deuxième partie de ce chapitre, où nous présentons notre troisième contribution dans cette thèse à savoir un algorithme de session sous sa version centralisée. Cet algorithme, placé au niveau de notre nœud photographe, permet de donner des vues (photos) à des instants donnés des nappes de pétrole dispersées en mer. Nous décrivons les outils de détection de fusion/scission des nappes de pétrole sur lesquels repose l'algorithme.*

- *Dans la troisième partie, nous décrivons le protocole de communication en implémentant notre algorithme mais cette fois-ci, de manière distribuée au niveau du collecteur mobile et au niveau du chef de cluster. Une évaluation de l'algorithme par simulation sous Opnet ainsi qu'une discussion termine cette partie.*

Enfin, nous clôturons ce chapitre par une conclusion qui résume les résultats obtenus et leur évaluation.

4.1 Introduction

Après avoir auto-organisé le niveau 1 de notre architecture de système de surveillance des nappes de pétrole en milieu marin en un nombre de clusters stables, nous allons nous focaliser dans ce qui suit sur le niveau 2. Ce niveau est constitué d'éléments mobiles ayant pour tâche de relever les informations du niveau 1 et les transmettre au système d'information (niveau 3). Mais la dynamique de l'application a pour conséquence qu'entre deux passages successifs dans une même région, ils vont trouver une situation qui aura potentiellement énormément évolué. Ce changement de contexte est un frein à la restauration d'un système d'information cohérent qui doit s'affranchir des ruptures de connectivité du niveau 1.

Nous illustrons cette problématique à travers l'exemple suivant :

Considérons deux clusters $C1$ et $C2$ et *CH($C1$)* et *CH($C2$)* leurs chefs de clusters respectifs. Entre deux visites du collecteur, les deux clusters $C1$ et $C2$ se regroupent et forment un seul cluster $C3$ (avec ou sans autres nœuds n'appartenant préalablement ni à $C1$ ni à $C2$). Plusieurs cas peuvent se présenter selon le résultat de l'exécution de l'algorithme de clustering :

- **Cas1** : CH($C1$) devient chef de cluster du nouveau cluster $C3$ → CH($C3$) = CH($C1$)
- **Cas2** : CH($C2$) devient chef de cluster du nouveau cluster $C3$ → CH($C3$) = CH($C2$)
- **Cas3** : un ancien nœud membre (v) du cluster $C1$ ou $C2$ devient chef de cluster du nouveau cluster $C3$ → CH($C3$) = (v).
- **Cas4** : Un nœud (u), qui n'appartenait ni à $C1$ ni à $C2$, se déclare chef de cluster du nouveau cluster $C3$ → CH($C3$) = (u).

Dans le cas1 ou le cas2, le collecteur lors de sa prochaine visite dans la région trouvera CH(C3)=CH(C1) ou CH(C3) = CH(C2), et relèvera les informations relatives à la zone couverte par le cluster C3. Il faut pouvoir l'informer de l'évolution du contexte. Cela peut paraître assez simple.

Mais dans le cas3 et le cas4, le collecteur lors de sa prochaine visite au chef de cluster CH(C3) = (v) ou CH(C3) = (w) relèvera des informations relatives à la zone couverte par le cluster C3 pour lesquelles l'association avec le précédent contexte parait moins évidente…

Dans les quatre cas, cités ci-dessus, le problème posé au niveau du collecteur est un problème de représentativité d'informations collectées par zone. Autrement dit, comment le collecteur peut savoir que les informations, qu'il vient de récupérer au niveau du nouveau CH(C3), représentent celles des zones C1 et C2 qu'il connaissait sachant qu'il n'a aucune information géographique sur l'emplacement de ces zones, et que ses correspondants ont potentiellement changé ?

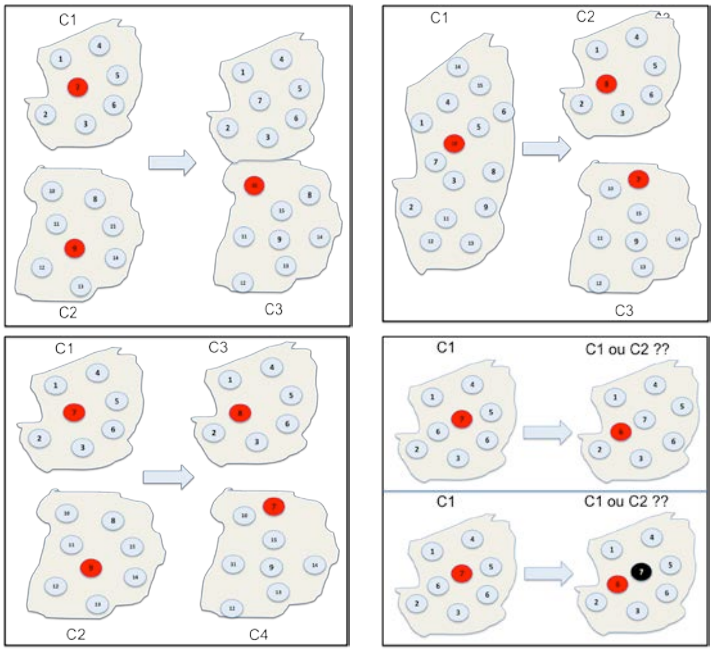


Figure 4-1 Quelques exemples d'évolution des clusters

La Figure 4.1 montre d'autres exemples de fusions ou scissions de clusters avec changements de chefs, et illustre la nécessité d'implémenter au niveau du collecteur l'algorithme de session que nous allons présenter, qui lui permettra de récupérer le contexte de ses précédentes visites. Il pourra ainsi reprendre le dialogue avec l'ancien ou éventuellement le nouveau chef de cluster élu, regrouper les informations spécifiques à chaque zone et les classer dans le bon ordre, et pourra donc recréer le contexte global de la collecte des données.

Autrement dit, même avec des connexions discrètes aux instants de visite du collecteur aux chefs de clusters, le collecteur peut reconstruire un système d'information cohérent sur l'ensemble des connexions déjà établies. Par exemple, dans le cas d'une fusion de clusters où il y a changement de chef, le collecteur attribue à ce dernier un identifiant de session déduit des anciens identifiants de session attribués aux anciens chefs des clusters fusionnés.

Nous privilégions la reconstitution du contexte au niveau du collecteur mobile plutôt qu'au niveau du système d'information pour les raisons suivantes :

- les problèmes de rupture de connexions dues aux facteurs internes du réseau (reconstruction des clusters) ou aux facteurs externes au réseau (interférences, mort des nœuds) seront transparents pour le système d'information.

- diminuer le temps de collecte de bout en bout. En effet, en échappant à l'obligation de créer de nouvelles connexions avec le système d'information nous réalisons une économie non négligeable en terme de temps. Ce gain en temps de collecte permet de relever avec précision l'évolution des nappes de pétrole dans le temps.

- la trajectoire du collecteur peut être définie localement en fonction des paramètres de session proposés au lieu qu'elle soit définie par le système d'information.

La mise en place d'un protocole de session assurera alors le maintien de la cohésion entre le collecteur et les chefs de clusters pour le relevé de l'information concernant l'évolution des nappes de pétrole. Cette cohésion se caractérise par :

- la reconnaissance des requêtes provenant d'un même cluster même si le chef de cluster change ou la représentativité du cluster (les membres qui le composent) change.

- l'association d'un profil (ensemble de paramètres sessions) au cluster.

- la connaissance des paramètres de l'application (dérive des nappes de pétrole, densité de l'hydrocarbure, viscosité, etc.)

4.2 Notations

Les notations adoptées au cours de ce chapitre sont rassemblées dans le tableau suivant :

S(k)	Sous ensemble informationnel du système d'information global à l'instant k
Gi	Groupe de nœuds
|Gi|	Nombre de nœuds au sein du groupe Gi
$CH(Gi)$	Chef du cluster Gi.
$Ci = \{i, \{j, k, m\}\}$	Cluster Ci avec i = CH(i)) est le chef du cluster i) et $\{j, k, m\} = V(Ci)$
$V(Gi)$	Ensemble des membres du cluster Gi dont le chef est CH(Gi).
$Pos(n)$	Position (x, y) du nœud n.
$D_{Moy}(Gi)$	La valeur moyenne des données mesurées au sein d'un groupe de nœuds Gi.
$Info(Gi)$	Informations produites par le chef de cluster pour la session. $Info(Gi) = \{CH(Gi), Pos(CH(Gi)), V(Ci), D_{Moy}, Pos(n)\}$
$LS^{-} = \{CH(G1), CH(G2), \ldots CH(Gn)\}$	Ensemble des chefs de clusters visités par le collecteur et qui ont été associés à des identifiants de sessions.
$Id_{session}(Gi)$	L'identifiant d'une session Gi est la ou les propriétés permettant de déterminer de façon unique chacune des sessions.

4.3 Définitions

La topologie du réseau change constamment suivant l'évolution de la dispersion des nappes de pétrole en mer. Ainsi, des groupes de nœuds capteurs peuvent fusionner et former un plus grand groupe. A

l'inverse, un groupe de nœuds capteurs peut se diviser en deux ou plusieurs sous-groupes de nœuds capteurs. Ces groupes dynamiques sont de taille variable, et donc informationellement significatifs ou non (y a-t-il intérêt à visiter un groupe de deux nœuds représentant une mappe de 20 cm de diamètre ?).

Nous définissons donc la session, et les relations d'équivalence totale ou partielle, sur lesquelles se basent les fusions ou les scissions qui représentent ces différents cas de figures de la dynamique du réseau, et qui vont gouverner la stratégie de visite du (des) collecteur(s).

4.3.1 Session

Une session représente l'échange d'un sous-ensemble informationnel du système d'information global S(k), entre un collecteur et un groupe de nœuds G à l'occasion de sa visite. Elle a donc une dimension spatiale et temporelle. k évolue à chaque nouvel échange du collecteur avec un groupe de nœud, et représente donc la discrétisation de la connectivité du réseau de capteurs global.

Dans un souci d'efficacité de la collecte, l'échange doit être informationellement significatif. On peut donc l'associer à un groupe de nœuds (cluster) de taille suffisante ($|G| \geq n$). Un cluster trop petit ne fera donc pas l'objet d'une session. Il ne sera pas visité car informationellement négligeable.

Une session i représente donc l'échange d'information avec un cluster Gi composé d'au moins un nombre (n) de nœuds. La valeur du nombre (n) dépend du nombre total de nœuds, mais surtout de la finesse avec laquelle on veut observer le phénomène physique. Une grande valeur de (n) fournira une vision macroscopique de la réalité. Une petite valeur de (n) donnera une vision plus précise, mais compliquera considérablement le programme de visite du (des) collecteur(s).

L'association d'une session à un cluster permet de réduire la trajectoire du collecteur aux chefs des différents clusters. Nous considérons qu'à chaque visite du collecteur au chef de cluster représentant un groupe de nœuds Gi, ce dernier peut lui communiquer les paramètres suivants : l'identifiant $Id_{session}(Gi)$ de session qu'il lui avait été assigné à la dernière visite ($G(j) \sqsubset S(k)$), sa position $Pos(CH(Gi))$ et du nœud membre le plus éloigné par rapport au chef de cluster $Pos(n)$, l'ensemble des identifiants des membres de son cluster Gi ainsi que la valeur moyenne des valeurs mesurées au niveau de son cluster $D_{Moy}(Gi)$. Le but de cette agrégation est de réduire la quantité d'informations à récupérer, et réduire ainsi le temps de parcours du collecteur.

4.3.2 Relations d'équivalence

Ces relations sont essentielles, car elles vont permettre le suivi du contexte entre deux visites du (des) collecteur(s), même si les clusters et/ou leurs chefs ont changé. En effet, un cluster peut avoir perdu ou gagné des nœuds, peut avoir élu un nouveau chef, peut s'être fondu dans un autre, ou avoir donné lieu à plusieurs autres clusters. Le collecteur doit pouvoir associer le nouveau contexte qu'il découvre à la situation qu'il avait observée lors de sa précédente visite.

4.3.2.1 Relation d'équivalence totale ou héritage total

La relation d'équivalence totale ou héritage total représente le cas de figure où dans un groupe de nœuds capteurs visité Gi, il existe au moins un pourcentage **Γ** de nœuds capteurs de ce groupe qui appartenaient à un groupe Gj visité précédemment.

$$Gi \in S(k+1) \text{ hérite totalement de } Gj \in S(k) <=> \exists\, g \subset Gi \;/\; g \subset Gj \text{ et } |g| \geq \Gamma |Gj|$$

Ainsi, nous considérons qu'une session A équivaut (totalement) à une session B si A contient un groupe de nœuds représentatifs d'au moins **Γ**% de B.

4.3.2.2 Relation d'équivalence partielle ou héritage partiel

La relation d'équivalence partielle ou héritage partiel représente le cas de figure où dans un groupe de nœuds capteurs visité Gi, il existe au moins un pourcentage γ ($\gamma < \Gamma$) de nœuds capteurs de ce groupe qui appartenaient à un groupe Gj visité précédemment.

$$Gi \in S(k+1) \text{ hérite partiellement de } Gj \in S(k) <=> \exists g \subset Gi \ / \ g \subset Gj \text{ et } |g| \geq \gamma |Gj|$$

Une session A hérite donc (partiellement) d'une session B si A contient un groupe de nœuds représentatifs d'au moins γ% de B.

4.3.3 Fusion et scission

4.3.3.1 Fusion

Une session est une fusion d'autres sessions si elle hérite totalement (équivalence) ou partiellement d'au moins deux sessions. La fusion symbolise le cas où une partie significative de deux ou plusieurs groupes fusionnent pour créer un nouveau groupe.

4.3.3.2 Scission

Une session se scinde en deux ou plusieurs autres sessions si chacune hérite au moins partiellement de la session d'origine à l'occasion de la division du groupe auquel elle était associée en groupes de taille suffisante.

4.3.3.3 Création / suppression de session

Une session est considérée nouvellement créée si elle est associée à un groupe de taille suffisante et si elle n'hérite d'aucune session existante.

Une session disparait si aucune session n'en hérite ou si le groupe auquel elle était associée ne contient plus au moins (n) nœuds.

4.4 Protocole de session sous sa vision globalisée

Dans cette section, nous décrivons l'algorithme sous sa version globalisée. Cet algorithme est placé au niveau de notre nœud Photographe (présenté dans le chapitre précédent). L'objectif de cet algorithme est d'identifier l'occurrence des fusions ou des scissions des clusters formés au niveau du réseau. Ainsi, grâce au nœud *Photographe* implémentant cet algorithme il est possible à tout instant k discret de connaître le système d'information global S(k). Attention, il s'agit d'une étape dans laquelle on imagine que le collecteur est capable d'échanger instantanément et simultanément avec l'ensemble des groupes de nœuds. Plus loin, nous présenterons ce protocole de session sous sa vision localisée tel qu'il fonctionnera réellement, c'est-à-dire lorsque le collecteur ne pourra dialoguer qu'avec un seul groupe de nœuds à la fois.

Cet algorithme s'exécute séquentiellement en trois phases distinctes :

- **Identification ou recherche des sessions** : cette phase représente la reconnaissance du réseau. L'algorithme commence par une scrutation du réseau à la recherche des groupes de nœuds.

- **Recherche des héritages** : durant cette phase, l'algorithme cherche l'existence ou non de liens entre l'état discret actuel du réseau (photo du réseau à l'instant $k+1$) et son état précèdent (photo du réseau à l'instant k).
- **Identification des attributs de sessions** : suite aux résultats obtenus à la phase 2, l'algorithme attribue des identifiants de sessions selon les types de la relation trouvée.
  - S'il n'y a pas de liens alors il y a création d'identifiant pour une nouvelle session.
  - Si le lien existe et est de type relation héritage total alors il y a transfert des paramètres de l'ancienne session à la nouvelle.
  - Si le lien existe et est de type héritage partiel alors l'identifiant de la session actuelle hérite des anciennes sessions vérifiant la relation d'héritage.
    Le collecteur créera un nouvel identifiant de session déduit des anciens identifiants de sessions.

$$Id_{session}((Id_{c1}),(Id_{c2}),\dots\dots,(Id_{cn})) \rightarrow Id_{session}(Id_{cx})$$

D'un point de vue informationnel, les mesures réalisées par les nœuds d'un groupe seront agrégées par une valeur moyenne. En cas d'héritage (et potentiellement de fusion de groupe de nœuds), ces valeurs seront-elles mêmes agrégées si de nouvelles mesures ne sont pas encore disponibles à l'instant de la visite du collecteur.

$$(D_{Moy}(G1),(D_{Moy}(G2),\dots\dots\dots(D_{Moy}(Gn)) \xrightarrow{c} (D_{Moy}(Gx))$$

Dès que les différents nœuds auront transmis leur mesure au chef de cluster, celui-ci mettre à jour la nouvelle valeur moyenne pour le groupe.

Au final, nous proposons d'implanter au niveau du nœud photographe (pour cette version global), l'algorithme suivant.

Algorithme 5 /* Algorithme placé au niveau du nœud photographe*/

Pour *tout groupe Gi à l'instant (k+1)*

$Info(Gi) = \{CH(Gi), Pos(CH(Gi)), V(Gi), D_{Moy}, Pos(n)\}$

$\varnothing \rightarrow Id_{session}(Gi)$

Pour *tout groupe Gj présent à l'instant k*

Si $\exists\ g \subset Gi\ /\ g \subset Gj$ *et* $|g| \geq \Gamma |Gj|$

alors */* $Gi \in S(k+1)$ hérite totalement de $Gj \in S(k)$ */*

$Id_{session}(Gj)\ U\ Id_{session}(Gi) \rightarrow Id_{session}(Gi)$

Finsi

Sinon Si $\exists\ g \subset Gi\ /\ g \subset Gj$ *et* $|g| \geq \gamma |Gj|$

Alors */* $Gi \in S(k+1)$ hérite partiellement de $Gj \in S(k)$ */*

$Id_{session}(Gj)\ U\ Id_{session}(Gi) \rightarrow Id_{session}(Gi)$

Finsi

Finpour

Si $Id_{session}(Gi) = \varnothing$

alors */* perte de contexte */*

Create_new ($Id_{session}(Gi)$*)* */*création d'un nouvel identificateur de session* $Id_{session}(Gi)$

Finsi

Finpour

Update($S(k+1)$*)* */* Mise à jour de l'ensemble des identificateurs de sessions */*

Dans cet algorithme, on note qu'il y a perte de contexte si aucune relation d'héritage n'a pu être identifiée et que par ailleurs, en cas de fusion, le contexte sera enrichi des contextes de chaque parent, qu'il s'agisse d'un héritage total ou partiel.

4.4.1 Exemples

Afin de donner une vision plus concrète tout en sachant que nous ne pouvons pas citer tous les cas possibles et imaginables résultants de la dynamique du réseau, nous utilisons quelques cas de figure pour illustrer les relations d'héritage total, d'héritage partiel et la fusion ou scission de clusters.

Dans ce qui suit, nous considérons deux vues du réseau à deux instants différents (k et $k+1$).

La vue du réseau à un instant k est supposée comme suit :

$C1$ = {**1** ;{2, 3, 4}} ; $C2$ = {**5** ;{6, 7, 8}} et $C3$ = {**9** ;{10, 11, 12}} et LS* = {**1, 5, 9**}

4.4.1.1 Exemple 1 : Relation d'équivalence total ou héritage total

La Figure 4-2 illustre le cas d'un changement de chef de cluster au sein du cluster C1 entre les instants $(k \text{ et } k+1)$.

Vue du réseau à l'instant $(k+1)$:

$C'1$= {**2** ;{1, 3, 4}} ; $C'2$ = {**5** ;{6, 7, 8}} et $C'3$ = {**9** ;{10, 11, 12}} et LS = {**2, 5, 9**}

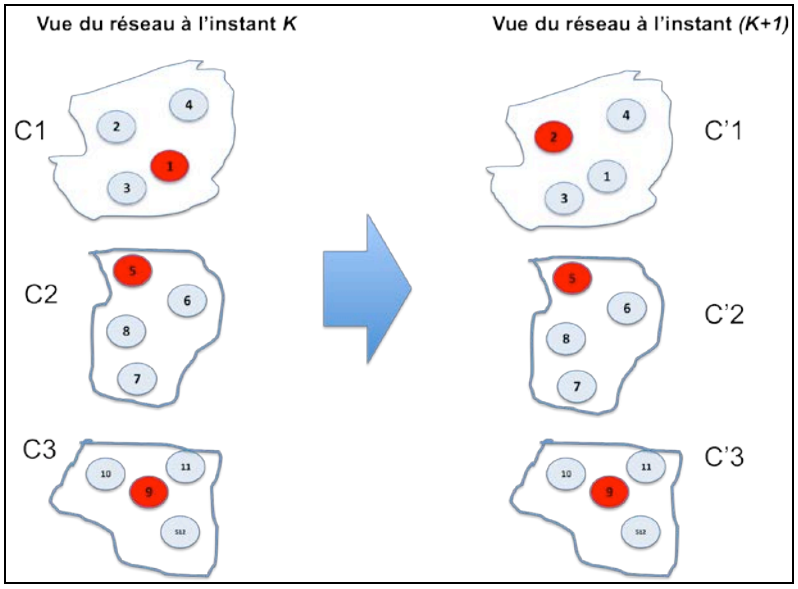


Figure 4-2 : Changement du chef de cluster au niveau du même cluster

Selon les deux vues du réseau aux instants $k \text{ et } k-1$ le nœud (**2**) se déclare chef de cluster et le nœud (**1**) devient un nœud membre. Ce cas de figure illustre la présence de trois équivalences totales. Et par conséquent, le nouveau chef de cluster (**2**) du cluster $C'1$ hérite totalement des paramètres de session du chef de cluster (**1**) du cluster $C1$. De même pour les chefs de clusters (**5**) et (**9**) des clusters $C'2$ et $C'3$ qui maintiennent leurs paramètres de session.

De même, le cas illustré sur la Figure 4-3 représente aussi un cas d'équivalence totale. Ce cas décrit la migration d'un ancien chef de cluster vers un nouveau cluster où il sera élu comme chef de cluster.

Vue du réseau à l'instant $(k+1)$:

$C'1$ = {**5** ;{1, 3, 4}} ; $C'2$ = {**1** ;{6, 7, 8}} et $C'3$ = {**9** ;{10, 11, 12}} et LS= {**1, 5, 9**}

D'après les deux vues du réseau aux instants $k \text{ et } k+1$, nous remarquons que malgré ce changement de clusters des chefs (**1**) et (**5**) la relation d'équivalence totale est toujours satisfaite et donc il y a héritage total des paramètres de session.

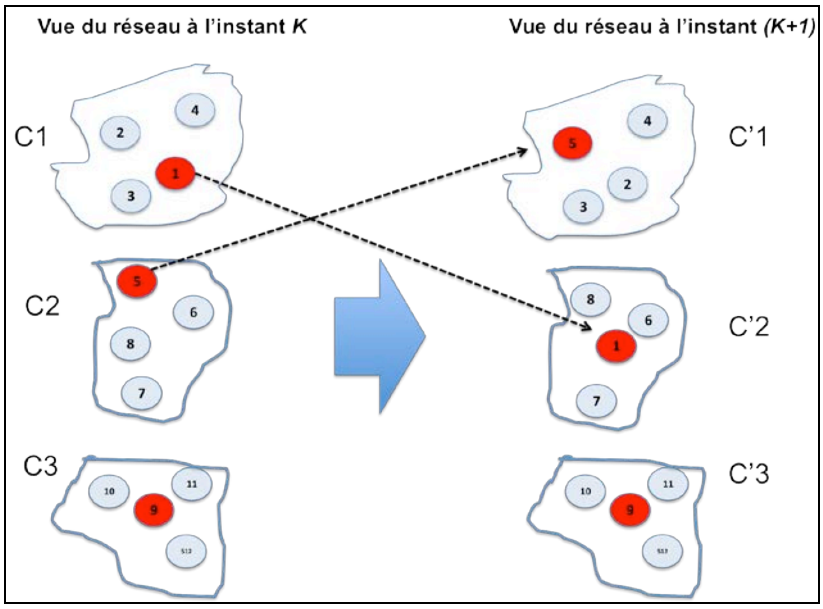


Figure 4-3 : Migration d'un ancien chef de cluster vers un nouveau cluster avec maintien de son statut

4.4.1.2 Exemple 2 : Relation d'équivalence partielle ou héritage partiel

La Figure 4-4 présente un cas de scission de clusters. Le cluster $C1$ se divise en deux clusters $C'4$ et $C'5$.

Vue du réseau à l'instant $(k+1)$:

$C'4$ = {**1** ;{2}} ; $C'5$ = {**3** ;{4}} et $C'3$ = {**9** ;{10, 11, 12}} et LS= {**2, 3, 5, 9**}

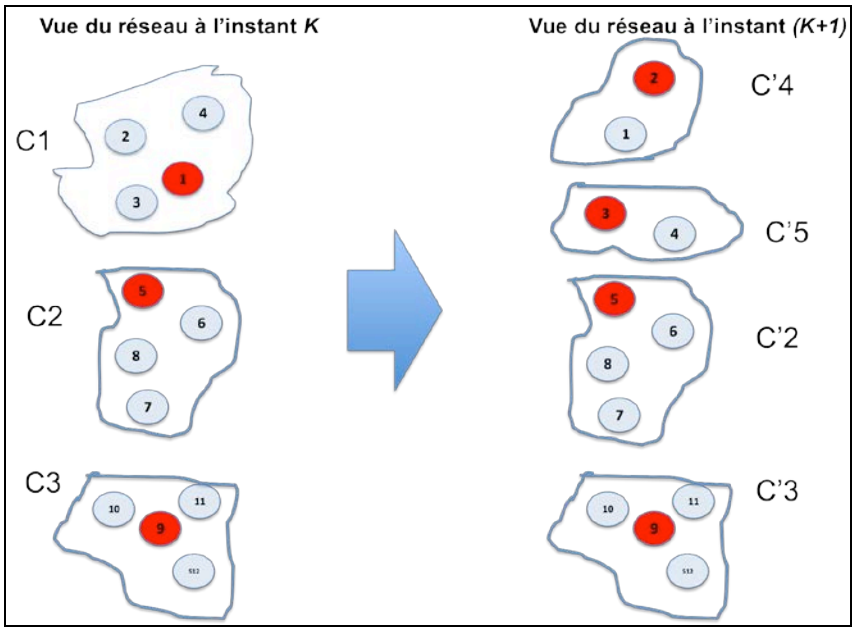


Figure 4-4 : Scission de cluster

Dans ce cas, le collecteur crée de nouvelles sessions pour les nouveaux clusters formés et supprime l'ancienne session attribuée à l'ancien cluster.

4.4.1.3 Exemple 3 : Fusion

Dans l'exemple de la Figure 4-5, nous retrouvons un cas de fusion de deux clusters C1 et C2 en un nouveau cluster C'4 dont le chef de cluster **(5)** est celui d'un des deux clusters fusionnés.

Vue du réseau à l'instant (k+1) :

C'4 = {**5** ;{1, 3, 4, 2, 6, 7, 8}} et C'3 = {**9** ;{10, 11, 12}} et LS= {**5, 9**}

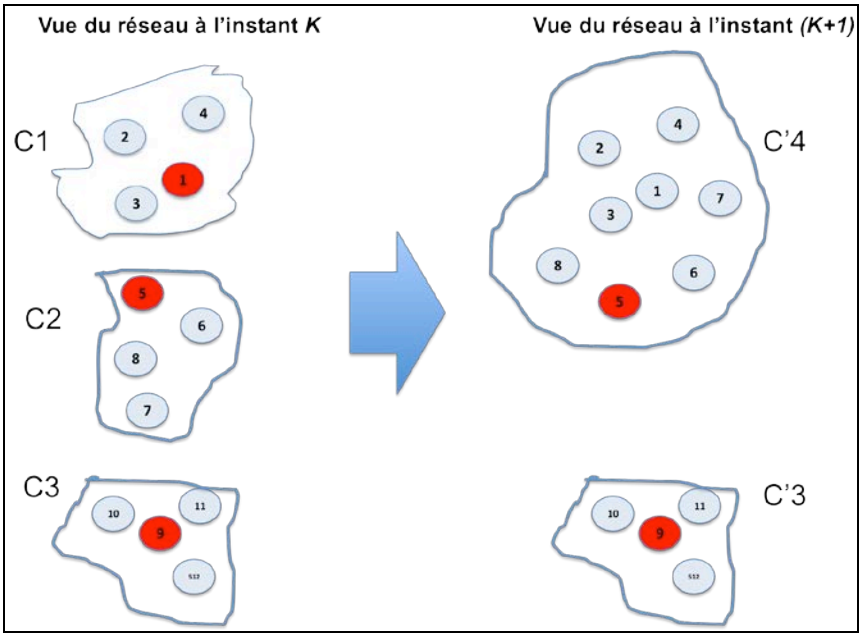


Figure 4-5 : Fusion de clusters

Dans ce cas l'algorithme détecte la fusion et attribue au nouveau chef de cluster un identifiant de session déduit des anciens identifiants de session attribués aux anciens chefs des clusters fusionnés.

4.5 Protocole de session sous sa vision localisée.

Une fois assurés que l'algorithme de session sous sa vision globalisée fonctionne correctement, nous avons procédé à son implémentation au niveau du collecteur mobile (et non plus du photographe), et à l'insertion d'autres fonctionnalités comme la collecte des données (échange des messages entre le chef de clusters et le collecteur mobile, et gestion de la mobilité du collecteur mobile). A chaque instant, le collecteur n'a donc plus qu'une vue rafraichie seulement par morceaux du système d'information, puisqu'il ne communique simultanément qu'avec un chef de cluster.

4.5.1 Hypothèses

Nous souhaitons tout d'abord étayer certaines hypothèses avant de décrire le fonctionnement du protocole de session proposé.

4.5.1.1 Mode opératoire de collecte des données

Le relevé des informations peut être fait de deux façons : soit en phase de mouvement (à la volée), ou en s'arrêtant pour communiquer. Nous utilisons le premier mode que nous considérons mieux adapté à notre cas vu la dynamique forte du réseau qui risque d'induire des volumes d'information à échanger importants.

Ainsi, en se déplaçant, le collecteur émet périodiquement un message de découverte de chef de cluster potentiel (*message de type Hello_CH*). Ce message est émis périodiquement sur un canal de communication dédié activé sur un nœud lorsque celui-ci passe à l'état chef de cluster.

Dès que le collecteur mobile entre dans la zone de couverture d'un chef de cluster, ce dernier accuse réception du message envoyé par le collecteur mobile par un message contenant les informations à transmettre.

La découverte de la présence d'un chef de cluster dans l'aire de captage des informations est une phase importante dans le mécanisme de collecte d'information. Cette phase doit non seulement permettre de correctement détecter la présence du point de contact, mais devrait aussi être opportune, pour que le temps de contact puisse être entièrement exploité. Ceci nous amène à considérer un intervalle de confiance du rapport signal sur bruit S/N (Signal/Noise). Ainsi, le nœud/chef de cluster pourra détecter la présence du collecteur (réception du message du collecteur) lorsqu'il est dans sa zone de couverture. Puisque la communication n'est possible qu'entre les contacts (DC : Début du Contact et FC : Fin du Contact) (Figure 4-6.). Le rayon de couverture utilisé comme référence est celui du chef de cluster et non celui du collecteur qui possède lui, un rayon de couverture plus grand.

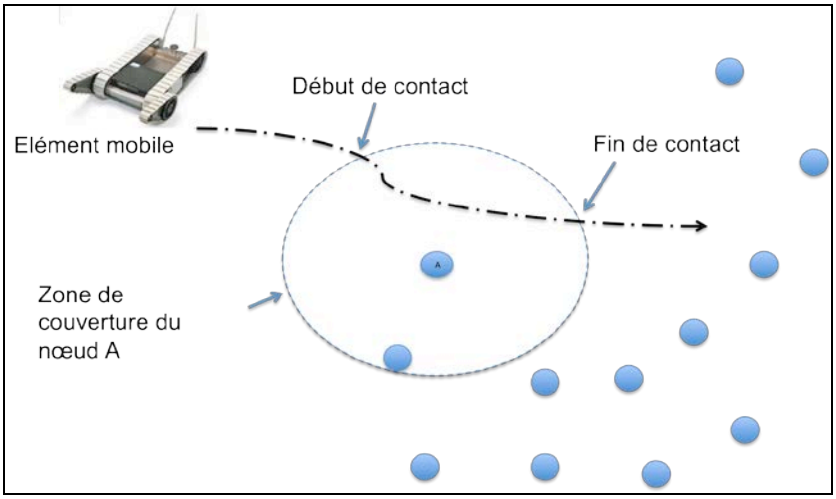


Figure 4-6 : Zone d'accrochage et d'échange

Le calcul de la position du chef de cluster trouvé par rapport à la position actuelle du collecteur est basé sur le rapport Signal sur Bruit S/N. Ainsi, la mesure de la qualité du rapport S/N permet un ajustement de la position du chef de cluster visité. Pour notre application, les paramètres considérés sont un seuil de puissance de réceptaion de -80 dB et une puissance d'émission de 0,0006253 V, soit une portée maximale de 50m (en atténuation en espace libre).

4.5.1.2 Modélisation de la vision du réseau par le collecteur

La théorie des graphes est largement utilisée dans la modélisation des réseaux de capteurs sans fil. Elle offre un cadre de choix pour la modélisation des réseaux dynamiques. Le réseau peut être regardé comme un graphe, où les différents nœuds du réseau sont généralement représentés comme étant les sommets d'un graphe dont les arrêtes représentent l'existence d'une communication, typiquement un canal de radiofréquence, entre les nœuds associés.

Plusieurs modèles de représentation des réseaux de capteurs sans fil utilisant la notion de graphe ont été proposés selon différentes approches. Parmi ces différentes approches nous pouvons citer l'approche aléatoire et l'approche communication.

4.5.1.2.1 Approche aléatoire

Souvent, la couche physique du réseau donne lieu à un graphe géométrique aléatoire. En raison de ses ressources limitées, un nœud capteur ne peut communiquer qu'avec les autres nœuds se trouvant à sa portée.

On définit un graphe comme étant connexe, s'il existe au moins une chaîne entre une paire quelconque de somment G. La relation est définie par :

$$x_i R\, y_i \Leftrightarrow \begin{cases} soit\ x_i = y_i \\ soit\ il\ existe\ une\ chaîne\ joignant\ x_i à\ y_i \end{cases}$$

Ainsi, le modèle de Barabasi-Albert et ses variations (Barabási et Albert, 1999) (Albert et Barabàsi, 2002), repose sur des arrêtes qui suivent une loi globale et prédéterminée. Ce modèle intègre la notion de croissance et celle de l'attachement préférentiel. La procédure proposée consiste à générer un graphe aléatoire de degré moyen d avec un petit nombre de nœuds. On augmente ensuite le graphe nœud par nœud. Ainsi le nombre de nœuds dans le réseau augmente avec le temps. Pour chaque nœud qu'on rajoute on le connecte à d voisins. La probabilité de choisir un nœud préexistant est proportionnelle au degré de ce nœud. Généralement, ce type de graphe est caractérisé par une forte hétérogénéité dans la distribution du degré de ses nœuds. Dans ce même type de graphes aléatoires, les auteurs de (Erdős et Rényi, 1960) présentent un graphe où les liaisons entre les nœuds obéissent à une loi probabiliste. C'est à dire qu'entre toute paire de nœuds (*u,v*) on place une liaison de probabilité *p.*

Un cas particulier des graphes aléatoires est présenté dans (Ferreira et Viennot, 2002). Les auteurs présentent un modèle de graphe géométrique aléatoire qui permet de fournir une exacte représentation de la structure topologique des réseaux de capteurs. Un graphe géométrique aléatoire G(n,r) est obtenu en distribuant (n) nœuds aléatoirement dans un cube unité de d-dimensions $[0,1]^d$.

4.5.1.2.2 Approche communication

Dans l'approche communication nous trouvons comme exemple, les graphes de connectivité et les graphes d'interférences.

Graphes de connectivité : (Li et al., 2009) représentent un réseau de capteurs sans fil par un graphe non orienté noté $G_c = (V, E)$ où V représente l'ensemble des nœuds du réseau dans un plan bidimensionnel et E l'ensemble des liens de communication sans fil entre les différents nœuds du réseau. Chaque nœud est supposé avoir un identifiant unique et dispose d'une antenne omnidirectionnelle. Soit *(u)* et *(v)* deux nœuds, *(u)* et *(v)* sont dits adjacents si et seulement si *(u)* est dans la zone de couverture de *(v)* ou selon le modèle de connectivité UDG *(Unit Disk graph*) (Clark et al., 1991), leur distance euclidienne est au plus égale à 1.

$$\forall u, v \in V\ tel\ que\ [u, v] \in E \Leftrightarrow |u, v| \leq 1$$

Le nœud *(u)* peut donc communiquer directement avec le nœud *(v)* d'une manière symétrique en cas d'absence d'interférences. Dans le cas contraire où les problèmes d'imperfections de liens radios doivent être pris en compte, le modèle QUDG (*Quasi Unit Disk Graph*) paraît le mieux adapté. Ce modèle définit la distance euclidienne entre deux nœuds (*u*) et (*v*) comme suit :

$|u, v| \leq \rho$, avec $\rho \in [0, 1]$, les nœuds sont dits adjacents ;
$|u, v| > 1$, les nœuds ne sont jamais dans les mêmes zones de couverture ;
$\rho \leq |u, v| < 1$, les nœuds peuvent être adjacents comme ils peuvent ne pas l'être.

En effet un nœud *(u)* peut communiquer avec un nœud *(v)* positionné à une douzaine de mètres de lui et ne pas pouvoir le faire avec un autre qui est juste à côté de lui. Des obstacles fixes ou temporaires peuvent changer la topologie du réseau au point de vue connectivité.

Graphes d'interférences : (Madan et al., 2006) vont permettre de prendre en compte les accès simultanés au médium par plusieurs stations en utilisant le modèle des *graphes d'interférences* G_i. Un récepteur *(u)* peut recevoir des messages d'un émetteur *(v)* avec succès si et seulement si *(u)* et *(v)* sont voisins dans le graphe de connectivité G_i et que *(u)* n'a pas d'émetteur concurrent dans le graphe d'interférences G_i.

4.5.1.2.3 Modèle de graphe utilisé pour le collecteur

La majorité des modèles cités ci-dessus ne prennent pas en considération la dynamique du réseau. Or notre réseau possède une dynamique plus au moins importante dépendante des courants marins, du vent, et de la mobilité des nœuds sur les nappes. Les paramètres propres du réseau évoluent donc avec le temps. Il est alors judicieux de choisir un modèle représentatif de notre réseau prenant en considération le facteur temps soit d'une manière discrète ou continue. Lorsque le temps est discret, on peut se ramener sans perte de généralité à considérer le réseau aux dates t = 0, 1, 2 . . . T, et on peut supposer que toutes les grandeurs relatives au temps sont des entiers. Le pas de discrétisation représente la période de visite du collecteur aux chefs de clusters qui doit garantir une bonne cohérence du système d'information, malgré les problèmes induit par la dynamique du réseau tels les changement de chefs de clusters, la destruction/construction de nouveaux clusters de tailles variables, etc.

Suite à toutes ces exigences, nous proposons d'utiliser un modèle exploitant la théorie des graphes dynamiques évolutifs mais considérée d'une manière distribuée. Dans ce modèle, le collecteur construit son propre graphe du réseau à 1 ou K-sauts, afin :

- de conserver au maximum les sessions en cours,
- de rétablir la connectivité du réseau (qui n'est que discrète).
- d'optimiser sa trajectoire ;
- de trouver les nouvelles positions des chefs de clusters, connaissant les anciennes positions et les valeurs des paramètres de sessions déjà établies. Il s'agit donc d'une vue agrégée où les clusters sont simplement représentés par leur chef.

Ainsi, nous représentons le réseau, vu par le collecteur, par un graphe dynamique évolutif (Monteiro et al., 2006) dirigé $S'(V, E)$. Ce graphe représente tous les nœuds et les connexions possibles, passées, présentes et futures du réseau. : V représente l'ensemble des nœuds (chefs de cluster) du réseau, E modélise l'ensemble des connections existantes entre ces nœuds et E_t l'ensemble des connexions entre les nœuds à l'instant t. Ainsi, si e = (*u*, *v*) ∈ E_t, veut dire que les nœuds *u* et *v* sont en mesure de communiquer directement à l'instant *t*.

Le réseau vu par le collecteur au temps (t = k) est modélisé par un sous-graphe statique $S'(k) = (V_k, E_k)$ du graphe global S'. Le graphe $S_G = S'(0), S'(1), \ldots\ldots, S'(T)$; $T \in N$ est une suite ordonnée de sous graphes partiels de S' et le graphe $S' = G(S', S_G)$ est appelé graphe évolutif.

Avec :

$$E_G = \bigcup E_k \text{ et } V_G = \bigcup V_k$$

$M = |E_G| \leq |E|$ et $N = |V_G| \leq |V|$

Ainsi à chaque instant (k) :

- Le réseau est représenté par un sous-graphe (statique) de $S'(k)$ dénoté $S'(k) = (V_k, E_k)$.

- Les liaisons entre deux nœuds (u) et (v) sont représentées par la matrice de liaison E_K [(u, v), k].
- La présence d'un nœud (u) dans le graphe est représentée par la matrice V_G [u, k].
- Les matrices E_K [(u, v), k] et V_K [u, k] représentent le graphe S'(k).

Sachant que le processus de clustering qui consiste à un découpage virtuel de V en un ensemble de groupes est représenté par :

$$V = [C_1, C_2, \dots C_k] \text{ Tel que : } V = \bigcup_{i=1}^{k}(C_i)$$

Le modèle des graphes évolutifs est un modèle de réseau dynamique à durée finie, où tous les paramètres du réseau évoluent de manière discrète. Il existe un ensemble de dates, appelées événements, qui correspondent à un changement dans l'état du réseau. Entre deux de ces dates, l'état du réseau est constant. Ainsi, un réseau dynamique à N nœuds est considéré comme un simple système dynamique discret dépendant du temps évoluant durant $[1, T]$.

Cette définition correspond à un modèle de graphe permettant l'agrégation d'une liste de sous-graphes statiques. Chaque sous-graphe représente l'état du réseau étudié à un moment donné. Les graphes ne peuvent pas être considérés invariants dans le temps. Aussi avec ce type de graphes la dynamique est totale (toutes les combinaisons d'ajouts, de suppression, de modification de liaisons et de nœuds sont envisageables). La topologie sera représentée sous forme de graphe dynamique évolutif.

La Figure 4-7 montre un exemple de sous graphes statiques représentant la vision du collecteur. On retrouve les différentes étapes de l'évolution d'un graphe dynamique dans le temps. Le graphe évolutif construit à partir de cette liste de sous-graphes $S'(k)$ contient donc tous les chefs de clusters et toutes les liaisons des sous-graphes considérés.

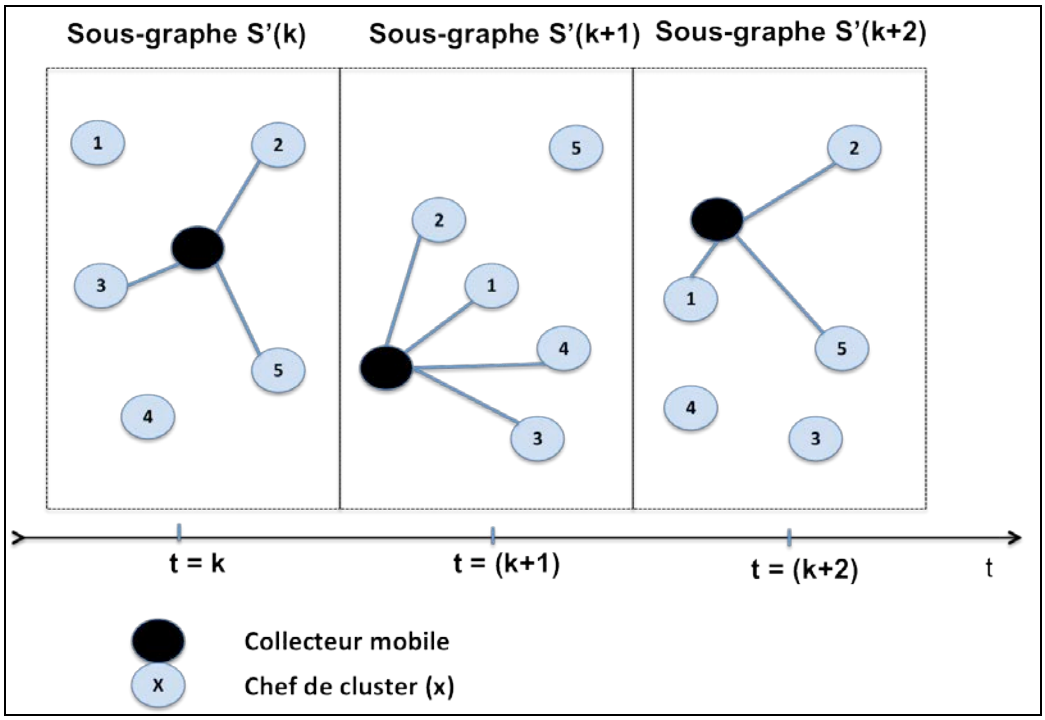


Figure 4-7 : Schéma illustratif de l'évolution des sous-graphes

Ainsi, à une fréquence d'échantillonnage bien définie (dépendante de la dynamique du réseau), notre collecteur mobile peut construire le graphe S'(k) contenant les chefs de clusters présents à cet instant. Il peut donc extraire à chaque instant (k) un sous graphe de chefs de clusters présentant une connexité.

L'intervalle de temps $[0, T]$, pendant lequel les positions des collecteurs sont mises à jour, est principalement lié au changement topologique du réseau. Un temps court correspond à une topologie très variable, où le déploiement doit s'adapter et correspondre à la morphologie du réseau.

4.5.2 Collecte des données

La collecte de données est une tâche très onéreuse en temps. Elle est souvent liée à la connectivité et à la couverture du réseau. Plusieurs travaux se sont concentrés sur cette tâche et ont proposé plusieurs techniques et mécanismes de collecte. Ainsi, les auteurs de (Xing et al., 2008) proposent une approche de collecte de données à base de rendez-vous qui explore la mobilité contrôlée de la station de base et la capacité de sauvegarde des données dans un réseau de capteurs sans fil. Un sous-ensemble de nœuds statiques dans le réseau servira de points de rendez-vous (RPs : *Rendez-vouz Points*) et agrégera des données produites par les nœuds capteurs. La station de base visite périodiquement chaque RP et prend les données sauvegardées. La collecte de données basée sur la technique de rendez-vous permet de réaliser un gain en consommation d'énergie et une diminution du retard dans les transmissions de données. Cependant, il faut avoir une bonne optimisation dans le choix des nœuds (RPs,) le trajet et la vitesse de déplacement de la station de base. Dans (Caillouet et al., 2011), les auteurs donnent une vision multicritère du problème de collecte de données des réseaux de capteurs sans fil. Le problème de placement optimal des nœuds de collecte (passerelles) de données dans les réseaux de capteurs économes en énergie. C'est à dire, déterminer où placer un ensemble de passerelles pour rassembler les données d'une région dans le réseau et calculer le parcours d'un puits mobile se déplaçant entre les passerelles pour rassembler des données des capteurs.

La solution d'introduire des collecteurs mobiles entre le réseau physique de captage et le système de traitement des données a permis de faire une optimisation de temps de collecte. Cependant, l'utilisation d'un collecteur mobile au sein d'un réseau dynamique génère des problèmes non ressentis dans les réseaux statiques (perte d'informations, modifications topologiques, …).

4.5.2.1 Perte d'informations

La perte d'informations est due à la rupture de communication entre le collecteur et le chef de cluster. Elle peut être causée soit par :

Imperfections de communication et épuisement d'énergie : Si les liaisons de communication sont imparfaites et le temps de transfert entre le collecteur et un chef de cluster est trop long, le chef de cluster peut perdre son statut de chef de cluster ou s'éloigner du collecteur pendant cet échange. Donc, le temps d'échange entre le collecteur mobile et le chef de cluster doit être le plus faible possible.

Modes de fonctionnement des protocoles utilisés : À cause des contraintes énergie et bande passante, la majorité des protocoles de communication utilisés dans les réseaux de capteurs sans fil sont de type non connecté. Dans ce genre de protocoles, chaque requête est traitée indépendamment des autres et aucun historique des différentes requêtes n'est conservé. Ainsi le collecteur ne peut pas se souvenir des requêtes précédentes échangées avec le chef de cluster. Sachant que ce dernier change à chaque fluctuation des paramètres de sa métrique.

4.5.2.2 Modifications topologiques

La métrique utilisée dans l'algorithme de clustering, reflète bien la dynamique du réseau à travers les trois paramètres qui la composent à savoir :

- l'énergie (un nœud qui meurt par épuisement d'énergie déclenche une restructuration de la topologie du réseau)

- la densité (fortement liée à la couche MAC du réseau et à la qualité de communication entre les nœuds)
- la mobilité du nœud par rapport à ses voisins.

Ainsi, chaque variation de la métrique d'un nœud déclenche le mécanisme d'élection qui conduira à une nouvelle carte du réseau en termes de clusters (en nombre et en taille). La collecte d'informations captées dans un réseau en mouvement permanent où les points de collecte (chefs clusters) changent de position et d'identité durant le temps est donc une tâche très difficile à réaliser.

Cependant, connaissant les matrices E et V, le meilleur chemin que peut emprunter le collecteur mobile pour explorer tout le graphe $S'(k)$ peut être calculé. Notons que le meilleur chemin correspond à celui qui offre une bonne connectivité et qui n'est pas forcément le plus court chemin.

Cette solution est certes très simple et ne nécessite aucune modification dans ce type de modèle. Cependant elle ne prend pas en considération l'historique du réseau (la topologie précédente du réseau). Autrement dit, si on dispose d'un déploiement à l'instant (k), à l'instant (k+1) le nouveau déploiement des chefs de clusters peut engendrer de longs déplacements pour le collecteur relativement à son ancienne position ou causer d'importantes interruptions de communications déjà en cours. Ce problème est résolu grâce à l'implémentation du protocole de session proposé décrit dans ce chapitre.

4.5.3 Descriptif de l'algorithme

Après déploiement et une fois la phase d'auto-organisation terminée, chaque nœud transmet, à travers son module de communication, les informations captées au niveau de sa zone de déploiement vers son chef de cluster d'une manière périodique. Nous avons opté pour le mode de transmission périodique au lieu du mode événementiel pour avoir accès à l'ensemble des données captées par le réseau physique de manière presque continue dans le temps. Ce qui après analyse, permettra d'assurer un monitoring permanent des nappes de pétrole. Le chef de cluster enregistre la valeur reçue dans une table des données de longueur N (N : nombre des nœuds membres du cluster). Le chef de cluster calcule la moyenne des valeurs reçues pour chaque valeur, ensuite calcule et stocke la moyenne des moyennes de l'ensemble des valeurs mesurées par tous les capteurs dans une table en attendant la visite prochaine du collecteur.

Le calcul de la moyenne des valeurs mesurées reçues se fait sur une période de temps égale à (T = 7s * N) (7s : période d'envoi des messages *HELLO*, N : nombre des nœuds membres du cluster). Période suffisante pour recevoir la totalité des mesures envoyées par tous les membres de son cluster. Le déclenchement du temporisateur associé à cette période est conditionné par la réception du premier message envoyé par un nœud membre.

Ensuite, chaque chef de cluster calcule la moyenne des moyennes calculées sur une période estimée à (4 * T). (Durée dépendante du nombre des chefs de clusters).

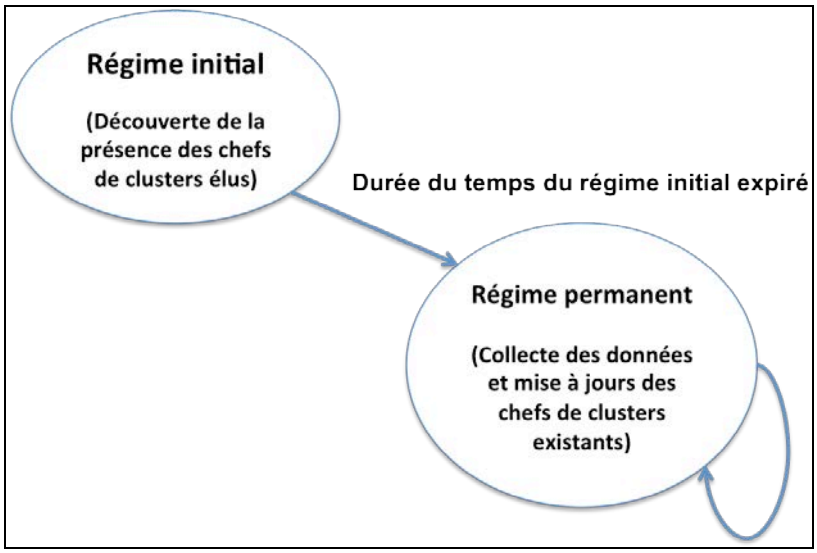


Figure 4-8 : Régime du comportement du collecteur mobile

Ainsi, comme le montre la Figure 4-8, le scénario de collecte des données comporte deux régimes de comportement du collecteur mobile : un régime initial pendant lequel le collecteur détecte les chefs de cluster élus et récupère leur position. Et un régime permanent durant lequel le collecteur suit une trajectoire calculée selon les informations obtenues dans le régime initial et les informations de la mise à jour des positions des chefs de clusters élus. Durant cette phase, le collecteur récupère les données captées et stockées au niveau des chefs de clusters et procède à la mise à jour de la liste des chefs de clusters existants.

4.5.3.1 Régime initial :

Cette étape consiste à former le premier graphe sur lequel nous allons définir la 1ère trajectoire dynamique que va emprunter le collecteur durant le premier tour du régime permanent une fois que la période allouée au régime initial sera écoulée. La racine du graphe est le collecteur et les chefs de clusters les points du graphe.

Durant le régime initial ou phase d'initialisation, le collecteur se déplace à une vitesse constante selon une trajectoire de type prédéfini. Nous suggérons l'utilisation de la fonction d'Archimède comme trajectoire du collecteur durant le régime initial : Initialement le collecteur est déployé au centre de la zone de déploiement. Ensuite et comme le montre la Figure 4-9, le collecteur se déplace suivant une seule trajectoire de type spirale de rayon de courbure égale à R (R : rayon de couverture d'un nœud).

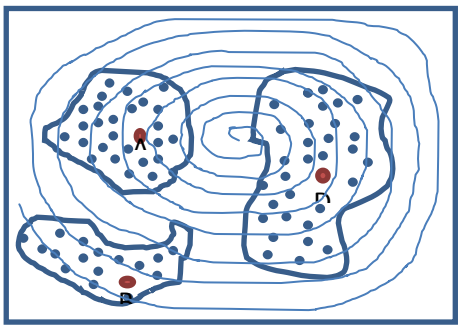


Figure 4-9: Trajectoire d'initialisation de récupération des chefs de clusters.

Notons que la fonction spirale d'Archimède se définit sous forme linéaire par l'équation :

$$X(t) = at\cos(t) \text{ et } Y(t) = at\sin(t)$$

Avec son équation centrale :

$$X^2 + Y^2 = a^2 Arctg(X/Y)^2$$

Et en coordonnées polaires (r,ϑ) est r = a+bϑ.

Ainsi, l'aire balayée par le collecteur sur l'intervalle [0,θ] est $a^2\theta^3/6$ et le pas de la spirale est e=2πa

4.5.3.2 Régime permanent de collecte des données

Durant le régime permanent où nous considérons que le collecteur a déjà fait un tour de reconnaissance du réseau et a pu associer des identifiants de sessions aux identifiants des chefs de clusters existants à l'instant k, la trajectoire du collecteur est de type dynamique. Le contrôle de cette trajectoire se réfère à la définition d'une politique qui change la trajectoire du collecteur en mouvement suivant les paramètres de sessions déjà établies durant le tour précédent. Ce contrôle doit satisfaire les contraintes spécifiques au réseau comme son changement topologique. Le nombre de sessions existantes donne aussi le nombre de points à visiter. Et avec un système de prédiction appuyé sur les paramètres de sessions, le collecteur mobile peut optimiser son parcours de visite des chefs de clusters.

Une fois le régime initial terminé, le collecteur mobile dispose du sous-graphe $S'(k) = (V_k, E_k)$ sur lequel il se base pour calculer sa trajectoire et définit un circuit de visite des chefs de clusters. Nous suggérons l'utilisation de la méthode d'Ambrosini pour la construction du sous-graphe du réseau $S'(k) = (V_k, E_k)$ (collecteur mobile, chefs de clusters) au niveau du collecteur mobile. Cette méthode suit le séquencement suivant :

- On part du collecteur mobile comme sommet du graphe ;
- On marque tous les sommets adjacents à ce sommet de départ ;
- Puis tous les sommets adjacents aux précédents et ainsi de suite.
- On s'arrête quand une passe ne permet pas de marquer de nouveaux sommets.

On considère une matrice M(n,n) (n : nombre de nœuds)

$$\begin{cases} Aij = 1 \text{ s'il y a une arrête entre les noeuds i et j} \\ Aij = 0 \quad \text{Sinon} \end{cases}$$

M(n,n)*M(n,n) donne les sommets reliés par au moins un chemin au plus de deux arrêtes de long

La puissance $p^{ème}$ de M(n,n) donne les sommets reliés par au moins un chemin au plus de *p* arêtes de long.

La puissance (n-1) de M(n,n) permet de connaitre la fermeture transitive du graphe ; si cette fermeture ne contient aucune clique (ie pas de zéro dans la matrice M(n,n)) alors le graphe est connexe.

La puissance (n-1) de M(n,n) permet aussi de connaitre les composantes connexes du graphe.

4.5.3.3 Diagramme des états finis au niveau du chef de cluster (Serveur)

La Figure 4-10 montre le diagramme des états finis du processus implémenté au niveau du chef de cluster (serveur). Il est constitué des quatre états suivants :

- ***Idle*** : initialisation et attente des messages entrants.

- ***Connect*** : envoie au client (collecteur) ses propres paramètres de session.
- ***Opened session*** : Connexion de session établie avec le collecteur.
- ***Resumed session*** : Connexion de reprise de session avec le collecteur.

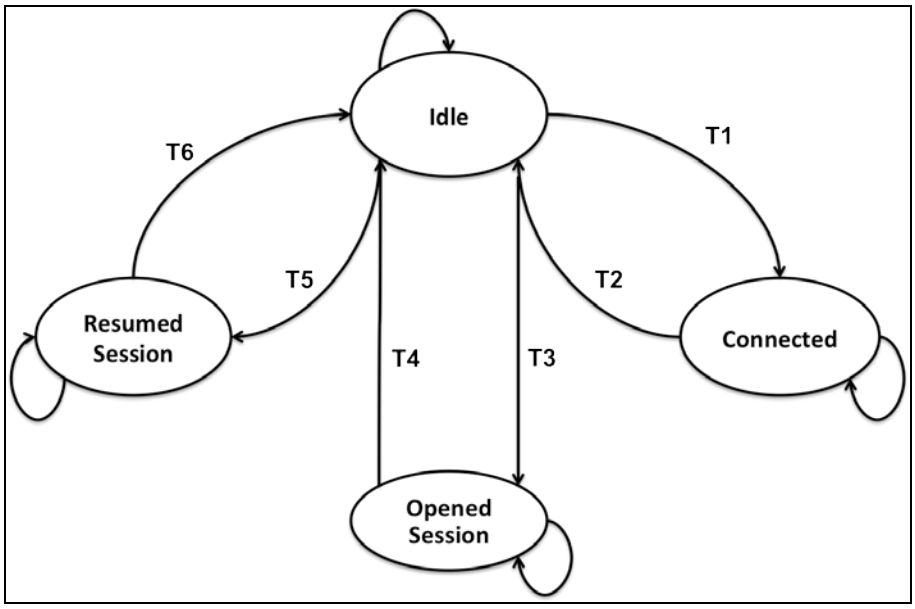


Figure 4-10 : Diagramme des états finis du serveur (chef de cluster)

Initialement le nœud est à l'état *Idle.* A la réception du message *INVITE* et si il est « chef de cluster », il passe à l'état *Connect*, envoie au collecteur ses propres paramètres et passe à l'état *Idle*. Si le nœud reçoit du collecteur le message *SESSION_PARAM_Client* de type ouverture de session*,* il passe à l'état *Opened session* et envoie au collecteur le message *SENT_DATA* contenant la valeur moyenne des valeurs mesurées et bascule à l'état *Idle*. Si le nœud reçoit du collecteur le message *SESSION_PARAM_Client* de type reprise de session, il passe à l'état *Resumed session* et envoie au collecteur le message *SENT_DATA* et bascule à l'état *Idle***.** Et si le nœud reçoit du collecteur le message *CLOSE_SESSION*, il ferme la session et reste dans l'état *Idle* pour attendre un nouveau message *INVITE.*

De même, le Tableau 4-1 présente les six transitions à sauter pour passer d'un état à un autre.

T1	Réception du message *INVITE* et l'état actuel du nœud est chef de cluster.
T2	Envoi du message *SESSION_PARAM_Server* ou notification de fermeture de connexion de session après temporisation.
T3	Réception du message *SESSION_PARAM_Client* de type ouverture de session.
T4	Envoi du message *Sent_Data* ou expiration de la durée de session ou notification de fermeture de connexion de session après temporisation.
T5	Réception du message *SESSION_PARAM_Client* de type reprise de session.
T6	Envoi du message *SENT_DATA* ou expiration de la durée de session ou notification de fermeture de connexion de session après temporisation.

Tableau 4-1: Transitions au niveau du serveur (Chef de cluster)

4.5.3.4 Diagramme des états finis au niveau du collecteur mobile (Client)

La Figure 4-11 montre le diagramme des états finis du processus implémenté au niveau du collecteur mobile (client). Ce diagramme est composé des cinq états suivants:

- ***Init*** : Initialisation, envoi du message *INVITE* au chef de cluster et attente du message *SESSION_PARAM_Server*.
- ***Connect*** : Connexion établie avec le chef de cluster détecté et test de la validité des relations équivalence totale ou partielle (invariance, fusion ou scission).
- ***Opened session*** : Connexion de session établie avec le chef de cluster.
- ***Resumed session*** : Reprise de connexion de session avec le chef de cluster.
- ***Close session*** : Fermeture de session

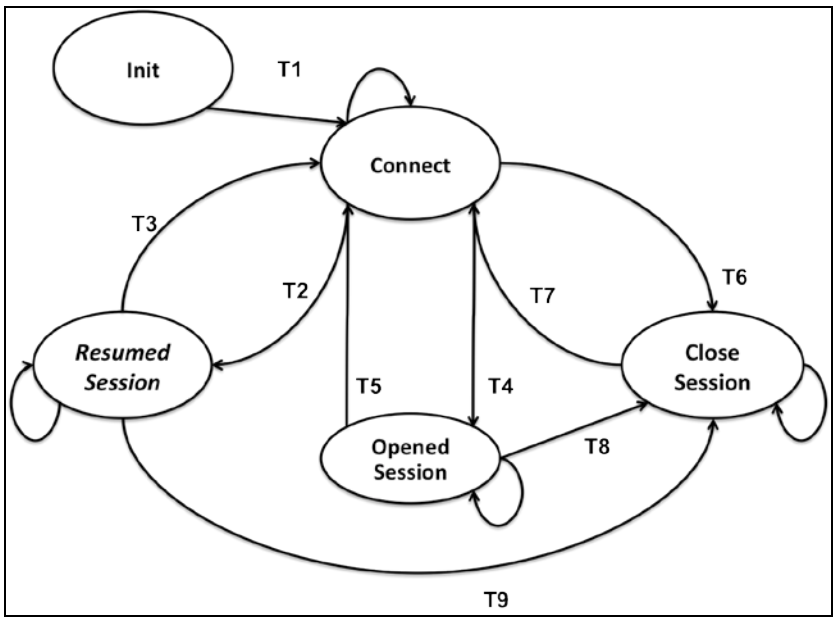


Figure 4-11: Diagramme des états finis du client (Collecteur mobile)

Initialement, le collecteur est à l'état *Init*. Dans cet état, le collecteur se déplace en émettant le message *INVITE* servant à inviter les chefs de clusters à initier une connexion de session. Et dès réception d'un message *SESSION_PARAM_Server*, le collecteur passe à l'état *Connect,* établit une connexion avec le chef émetteur du message et lance la procédure de test de la validité des relations d'héritage. Si la transition T2 est franchissable c'est à dire une des relations d'héritage est trouvée, le collecteur envoie au chef de cluster le message *SESSION_PARAM_Client* et passe à l'état *Resumed session*. Si aucune des relations d'héritage n'est détectée alors le collecteur passe à l'état Opened Session et ouvre une nouvelle session avec le chef de cluster visité. Si le temporisateur associé à la session a expiré ou il y a erreur de transmission alors le collecteur passe à l'état *Close Session*. La valeur du temporisateur est choisie intuitivement, il faudrait étudier en perspective comment optimiser le choix de cette valeur soit à travers des tests sur le terrain ou suite au degré de finesse de la collecte des données.

Les transitions à franchir pour transiter entre les états sont décrites dans le Tableau 4-2.

T1	Réception du message *SESSION_PARAM_Server*.
T2	Une des relations équivalence, fusion ou scission est vérifiée.
T3	Envoi du message *SESSION_PARAM_Client* de type reprise de session.
T4	Aucune des relations d'équivalence ou fusion ou scission n'est vérifiée.
T5	Envoi du message *SESSION_PARAM_Client* de type ouverture de session.
T6, T8 et T9	Temporisateur associé à la session a expiré ou cas d'erreur.
T7	Envoi du message *CLOSE_SESSION.*

Tableau 4-2 : Transitions au niveau du client (Collecteur mobile)

Notons que les six messages utilisés par le protocole ne sont traités que par le collecteur mobile et le nœud capteur lorsqu'il est dans l'état chef de cluster.

- *INVITE* : Ce message est utilisé par le collecteur pour inviter un chef de cluster présent dans son voisinage à initier une connexion de session. Ce message est de type diffusion et est constitué des champs suivants : (champ type, identifiant du collecteur).
- *SESSION_PARAM_Client* : Ce message est envoyé par le collecteur au chef de cluster. Ce message contient les paramètres de session. Ce message est de type unicast et il est composé des champs : (Champ type, identifiant du collecteur, identifiant du chef de cluster, durée de la session, etc). Le champ type prend la valeur 0 lorsque le message envoyé est de type ouverture de session. Et prend la valeur un lorsqu'il est de type reprise e session.
- *SESSION_PARAM_Server* : Ce message envoyé par le chef de cluster au collecteur, correspond à un accusé de réception du message *INVITE* et par conséquent une acceptation d'ouverture de session avec le collecteur mobile. Ce message est de type unicast et il contient les paramètres de session. Les champs le constituant sont : (Champ type, identifiant du collecteur, identifiant du chef de cluster, position du chef de cluster, la liste des nœuds membres du cluster, la position du nœud le plus éloigné du chef de cluster).
- *SENT_DATA* : Ce message est envoyé par le chef de cluster. Ce message est de type unicast. Et il est constitué des champs suivants : (Champ type, identifiant du collecteur, identifiant du chef de cluster, valeur moyenne des valeurs mesurées au niveau du cluster).
- *Error_Notification :* Ce message de notification est envoyé en cas erreur.
- *CLOSE_SESSION* : Ce message est envoyé par le chef de cluster pour fermer une session ouverte.

4.5.3.5 Diagramme de séquence et diagramme Use case

Le diagramme de séquence qui traduit l'échange des messages entre le collecteur mobile et le chef de cluster, présentés dans un ordre chronologique, est décrit sur la Figure 4-12.

- *INVITE* : Ce message est utilisé par le collecteur pour inviter un chef de cluster présent dans son voisinage à initier une connexion de session. Ce message est de type diffusion et est constitué des champs suivants : (champ type, identifiant du collecteur).
- *SESSION_PARAM_Client* : Ce message est envoyé par le collecteur au chef de cluster. Ce message contient les paramètres de session. Ce message est de type unicast et il est composé des champs : (Champ type, identifiant du collecteur, identifiant du chef de cluster, durée de la session, etc). Le champ type prend la valeur 0 lorsque le message envoyé est de type ouverture de session. Et prend la valeur un lorsqu'il est de type reprise e session.
- *SESSION_PARAM_Server* : Ce message envoyé par le chef de cluster au collecteur, correspond à un accusé de réception du message *INVITE* et par conséquent une acceptation d'ouverture de session avec le collecteur mobile. Ce message est de type unicast et il contient les paramètres de session. Les champs le constituant sont : (Champ type, identifiant du collecteur, identifiant du chef de cluster, position du chef de cluster, la liste des nœuds membres du cluster, la position du nœud le plus éloigné du chef de cluster).
- *SENT_DATA* : Ce message est envoyé par le chef de cluster. Ce message est de type unicast. Et il est constitué des champs suivants : (Champ type, identifiant du collecteur, identifiant du chef de cluster, valeur moyenne des valeurs mesurées au niveau du cluster).
- *Error_Notification :* Ce message de notification est envoyé en cas erreur.
- *CLOSE_SESSION* : Ce message est envoyé par le chef de cluster pour fermer une session ouverte.

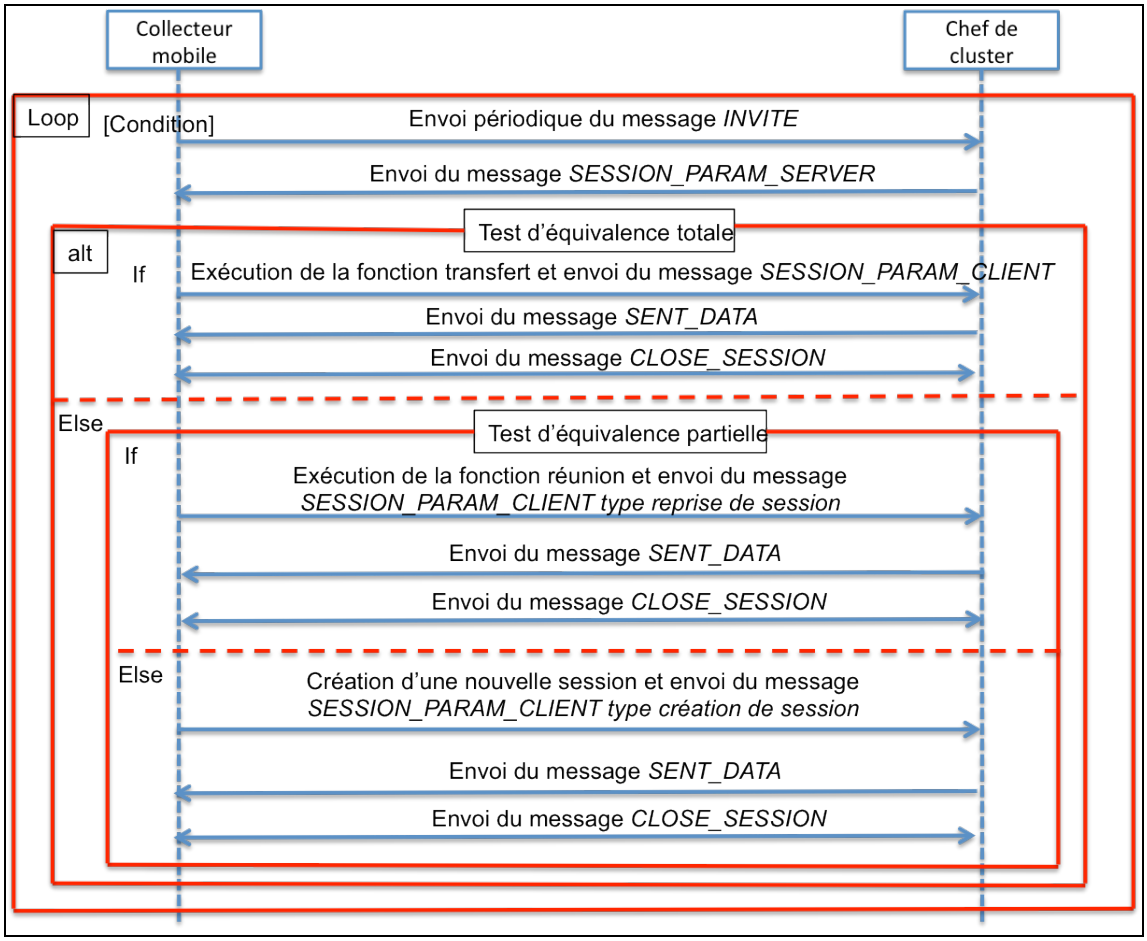


Figure 4-12 : Diagramme de séquence

Les deux acteurs principaux sont le collecteur mobile et le chef de cluster visité. Le collecteur en se déplaçant envoie un message *INVITE* permettant d'inviter un chef de cluster à initier une session. Et dès réception d'un message de type *SESSION_PARAM_Server* provenant d'un chef de cluster, le collecteur mobile lance la procédure de test des relations d'héritage.
La procédure commence par tester l'existence d'une relation de type équivalence totale. Si elle existe alors le collecteur envoie au chef de cluster un message de type *SESSION_PARAM_Client*.. Ce message contient les paramètres de session. Le chef de cluster accuse la réception de ce message par un message *SENT_DATA* contenant la moyenne des valeurs mesurées au niveau de son cluster. Et ferme la session avec le collecteur mobile par l'envoi du message *CLOSE_SESSION*.

Sinon, Le processus de détection de présence des relations d'héritage teste l'existence d'une relation d'équivalence partielle. Si oui alors le collecteur envoie le message *SESSION_PARAM_Client*.de type reprise de session. A la réception de ce message, le chef de cluster répond avec le message *SENT_DATA* contenant la donnée et ferme la session par l'envoi du message *CLOSE_SESSION*.

Si aucun type de relation n'est trouvé, le collecteur envoi au chef de cluster visité un message *SESSION_PARAM_Client* de type création de session. Le chef de cluster après réception de ce message lui envoie le message de données *SENT_DATA* contenant la donnée et ferme la session par l'envoi du message *CLOSE_SESSION*.

Sur la Figure 4-13, nous présentons le diagramme de cas d'utilisation. Ce diagramme modélise les fonctionnalités et services offerts de notre système. Il décrit les fonctions effectuées par les deux acteurs à savoir le collecteur mobile et le chef de cluster.

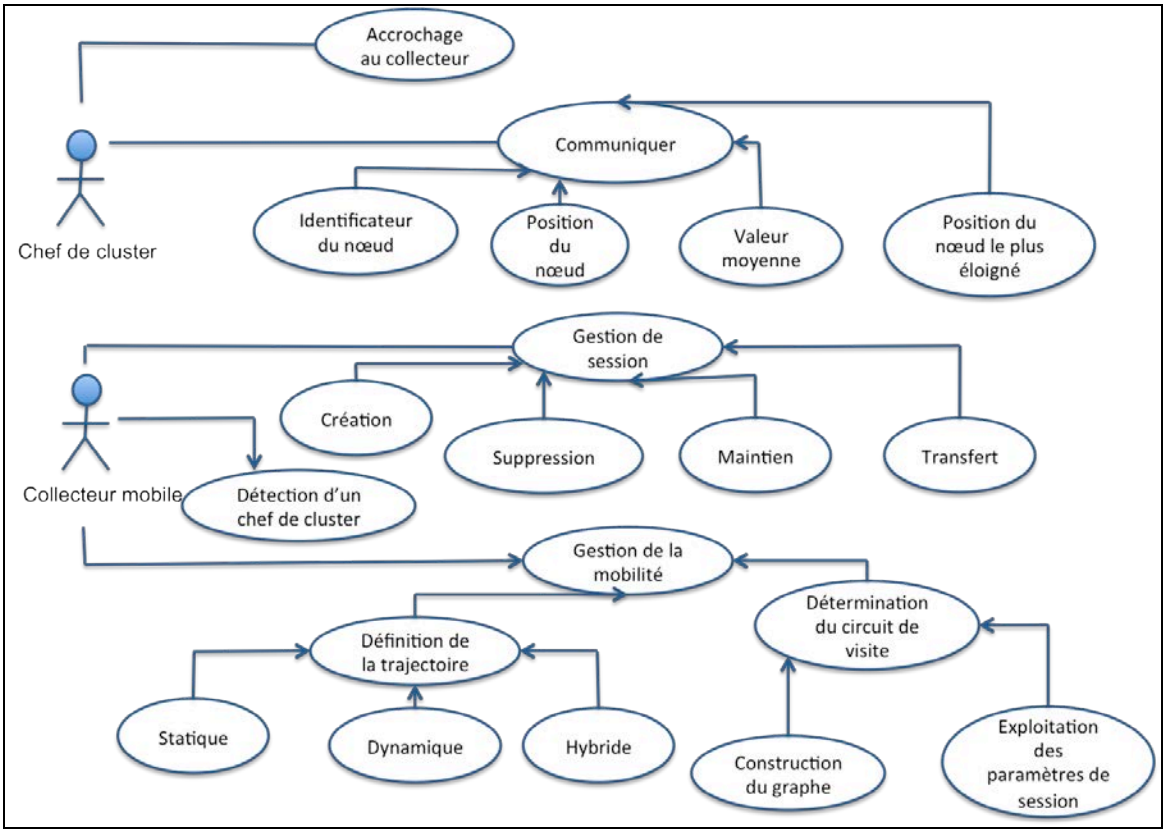


Figure 4-13 : Diagramme Use Case

Ainsi, le chef de cluster participe à la fonction d'accrochage au collecteur et à la fonction de communication des données de session. Alors que le collecteur mobile réalise les taches de gestion de sessions qui englobe la création, la suppression, le maintien et le transfert de sessions. Un service de gestion de mobilité est offert au collecteur mobile. Ce service permet la définition de type de la trajectoire (statique, dynamique ou hybride) et la détermination du circuit de visite que le collecteur mobile peut emprunter. Ce circuit de visite est déterminé sur la base des informations de sessions et le graphe construit par le collecteur mobile.

4.5.3.6 Algorithmes:

Dans cette section nous présentons les algorithmes correspondants aux deux régimes initial et permanent de la collecte des données décrits plus haut.

Algorithme 6 /* Algorithme du régime initial */

Tant que *fin parcours régime initial =faux* ***faire*** *# parcours défini selon la trajectoire de type spirale d'Archimède#*

Envoi du message INVITE
Réception du message SESSION_PARAM_SERVER
Mesure du SNR
Si *SNR est dans l'intervalle de confiance* ***alors***
Enregistrement des informations obtenues dans la table_position_CH
Création de la table des CHs<->Id_sessions<-> données
Récupération / Création de l'Id_session
Mise à jour de la table des CHs<->Id_sessions<-> données
Fin Si

Fin tant que

Algorithme 7 /* Algorithme du régime permanent de collecte des données */

Tant que *fin parcours du régime permanent =vrai* ***faire*** *# parcours défini selon la trajectoire calculée selon le graphe obtenu dans le régime initial #*

Envoi du message INVITE
Réception du message SESSION_PARAM_SERVER
Mesure du SNR
Si *SNR est dans l'intervalle de confiance* ***alors***
Enregistrement des informations obtenues dans la table_position_CH
Création de la table des CHs<->Id_sessions<-> données
Récupération / Création de l'Id_session
Mise à jour de la table des CHs<->Id_sessions<-> données
Calcul de la trajectoire suivant le graphe obtenu
Fin Si

Fin tant que

Algorithme 8 /* Algorithme exécuté par un nœud capteur à la réception du message INVITE

Si *état du nœud = chef de cluster* ***alors***
Encapsulation des données (identfiant du cluster, position) dans message d'envoi au collecteur.
Envoi du message SESSION_PARAM_SERVER au collecteur
Sinon *détruire message reçu.*

4.6 Mécanisme de suivi de session

Ce mécanisme concerne la mise en place de la session, de son suivi, et de sa terminaison. Il assure le stockage des informations concernant le chef de cluster sur le collecteur mobile, et les relie à chaque requête grâce à un identifiant unique de session présent dans la requête et sur le collecteur mobile.

Le maintien de l'identification d'un chef de cluster par le collecteur peut se faire soit :

- en échangeant des données d'identification du chef de cluster entre chaque requête à travers des variables d'identification ou en passant en paramètre un champ de formulaire caché contenant l'identifiant de session. Cette méthode nécessite une bonne capacité mémoire (gestion du formulaire) et une bonne bande passante (nombre de requêtes échangées est assez grand).
- en demandant l'identification du chef de cluster à chaque fois.

Notons que nous avons opté pour l'utilisation de la dernière méthode. Du fait qu'elle est simple et ne nécessite que l'ajout d'un champ réservé à l'identification du chef de cluster dans chaque message échangé entre le collecteur mobile et le chef de cluster.

De même, notons que les informations concernant les chefs de clusters sont stockées sur le collecteur dans une table persistante, nommée *TAB_SESSION*. Cette table stocke toutes les informations de session concernant les chefs de clusters qui sont indexées par l'identifiant du chef de cluster passé en paramètre dans chaque requête.

4.7 Gestion des sessions

Les sessions permettent de conserver des informations pendant toute la durée d'un cycle de visite du collecteur mobile. La question de la gestion des sessions mérite une attention particulière lorsque l'application a besoin de retrouver en mémoire des informations de contexte, relatives aux échanges précédents avec le chef de cluster de la même session.

La gestion des sessions se préoccupe des taches suivantes, et se fait comme suit :

Génération des identifiants de sessions : La génération des identifiants de session associé à l'identifiant du chef de cluster est retourné par la fonction *getidsession(Random(y) + id (CH)), x)*.

Extraction de l'identifiant d'une session existante : nous faisons appel à la fonction *getsession()* qui retourne l'identifiant de session associé au chef de cluster visité.

Obtention des informations associées à une session existante : pour obtenir une information stockée dans la table *TAB_SESSION*, nous utilisons la fonction *getattribut()* de la table *TAB_SESSION*. La fonction retourne un entier. Si l'attribut passé en paramètre n'existe pas, la fonction retourne la valeur *null*.

Stockage des informations de session : le stockage des informations dans la session est réalisé par la fonction *setattribute()* en lui fournissant comme attribut l'identifiant de session et la valeur associée ainsi que le temps passé.

La suppression d'une session : Lorsqu'une session n'est plus valide, elle est détruite en faisant appel à la fonction *supp_session()*.

Les procédures sont définies en termes:

- d'interactions entre entités de session homologues, par échange d'unités de données de protocole de session;
- d'interactions entre une entité de session et l'utilisateur du service de session, par échange

de primitives du service de session;

- d'interactions entre une entité de couche session et la couche réseau/clustering, par échange de primitives du service réseau.
- adresse de session d'appelant: Identifie l'utilisateur du service de session qui constitue la source des données pendant une instance déterminée de transmission de session
- adresse de session d'appelé: Identifie l'utilisateur du service de session qui constitue le collecteur des données pendant une instance déterminée de transmission de session.

4.8 Simulations

Pour montrer le bon comportement du protocole proposé, nous avons effectué des séries de simulations. Dans cette section, nous présentons les premiers résultats obtenus. L'objectif des simulations faites est d'évaluer le protocole selon trois métriques : le nombre de sessions, le nombre de nœuds hors session et le nombre de relation entre deux vues des sessions. Les métriques sont définies en fonction des paramètres (n, v, Γ% et γ %).

- le « *nombre de session* » définit le nombre de clusters existants. «Tout cluster d'au moins (**n)** membres (puits compris) est une session».
- le « *nombre de nœuds hors sessions* » nous permet d'estimer l'aspect statique de notre réseau à un instant donné, et donc de la capacité de couverture de toute la zone de captage. Plus le nombre de nœuds hors session est faible, plus la zone de captage est bien couverte.
- le « *nombre de relations entre deux vues des sessions* », nous permet d'estimer l'aspect dynamique du réseau. Plus le nombre de relations est important, plus le suivi de l'évolution des nappes est précis et donc la prédiction des futures positions des chefs de clusters est meilleure.

Ces métriques sont évaluées selon les facteurs suivants :

- Nombre de nœuds représentatifs d'une session ;
- Taille d'une session ;
- Les taux seuils Γ et γ ;
- La densité des nœuds ;
- Le rapport (nombre de nœuds représentatifs/densité) ;
- La vitesse des nœuds v.

4.8.1 Environnement de simulation

Les paramètres considérés durant les simulations sont :

Paramètres	Valeurs
Nombre de nœuds	52
Zone de déploiement	625 m^2
Facteur de pondération pour le clustering ($\propto$)	0,5
Durée de la simulation	300s
Nombre de seeds	30
Période de photographie du clustering	1s
Période de photographie des sessions (**p**)	5, 7 et 10s
Taille minimale d'une session (n)	3,4 et 5 nœuds
Seuil d'équivalence ou héritage total (**Γ**)	70, 80, 90 et 100%
Seuil d'héritage partiel (**γ**)	30, 40 et 50%
Vitesse des nœuds	1 et 2 m/s
Taille des données des résultats obtenus	*3x4x3x2x3x30=6480 simulations*

Tableau 4-3: Valeurs des paramètres de simulation

4.8.2 Résultats et discussion

4.8.2.1 Premiers résultats obtenus

Dans cette section nous présenterons les résultats obtenus suite à une première série de simulations. Leur analyse nous conduiront à mener une seconde campagne de simulation.

4.8.2.1.1 Nombre de sessions

La Figure 4-14 représente la variation du nombre de sessions présentes sur le réseau sur une période de 300 secondes. Les relevés sont obtenus à des prises de vues de l'état du réseau toutes les 5 secondes pour les taux seuils Γ et γ respectivement égaux à 70% et à 30%, une vitesse des nœuds de l'ordre de 1m/s et un nombre minimal (n) des nœuds représentatifs égal à 5.

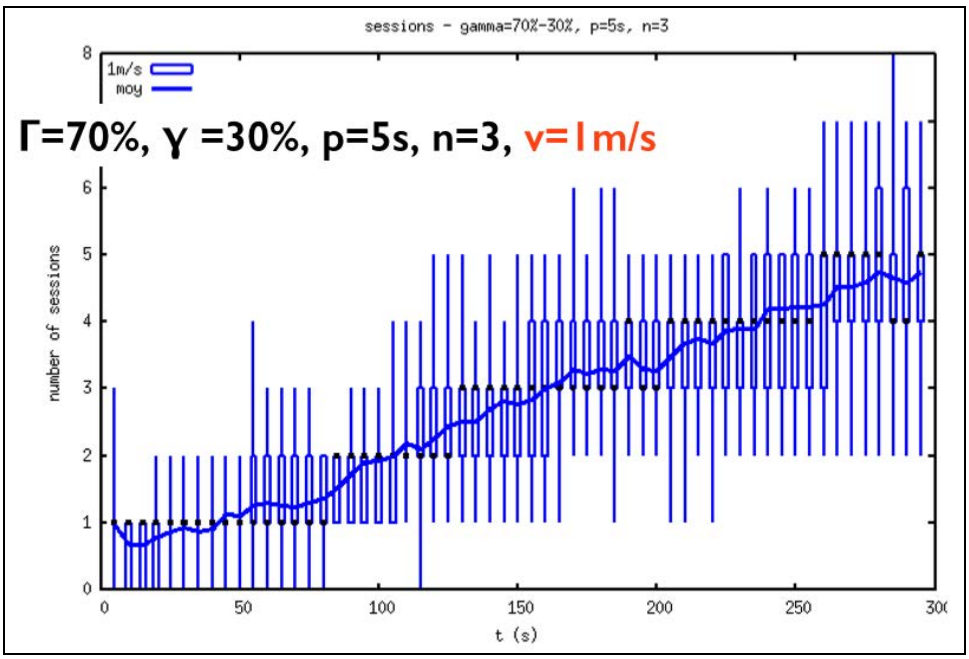


Figure 4-14 : Nombre de sessions en fonction de Γ, γ et n.

La Figure 4-15 montre l'évolution dans le temps du nombre de nouvelles sessions pour un taux seuil γ égal à 80%, une vitesse de déplacement des nœuds de l'ordre de 1m/s. Les relevés sont effectués toutes les 5 secondes.

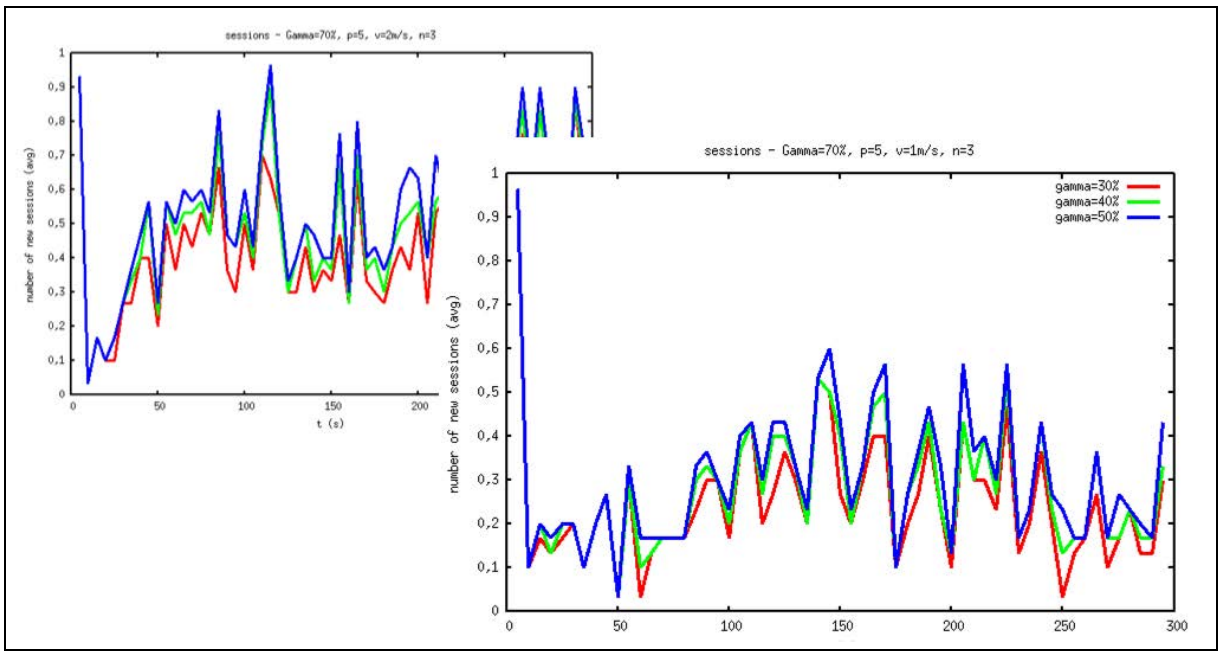


Figure 4-15: Nombre de nouvelles sessions en fonction du temps.

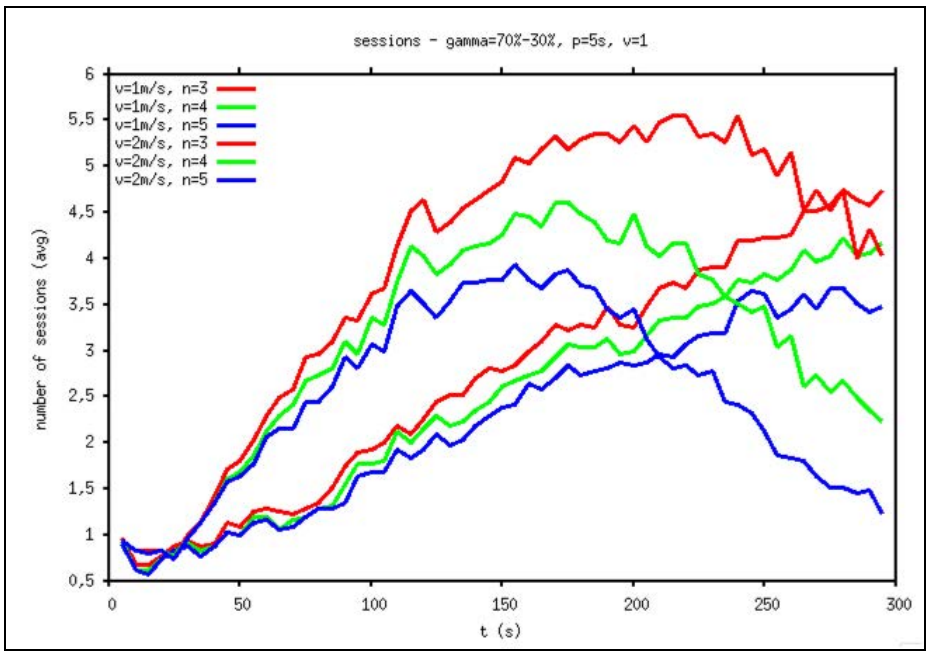


Figure 4-16 : Nombre de sessions en fonction de la vitesse et n.

A travers ces courbes nous constatons que les seuils Γ et γ influent sur le type d'héritage mais pas sur l'existence d'une session. Et que la valeur du seuil γ affecte le nombre des nouvelles sessions crées (Figure 4-15).

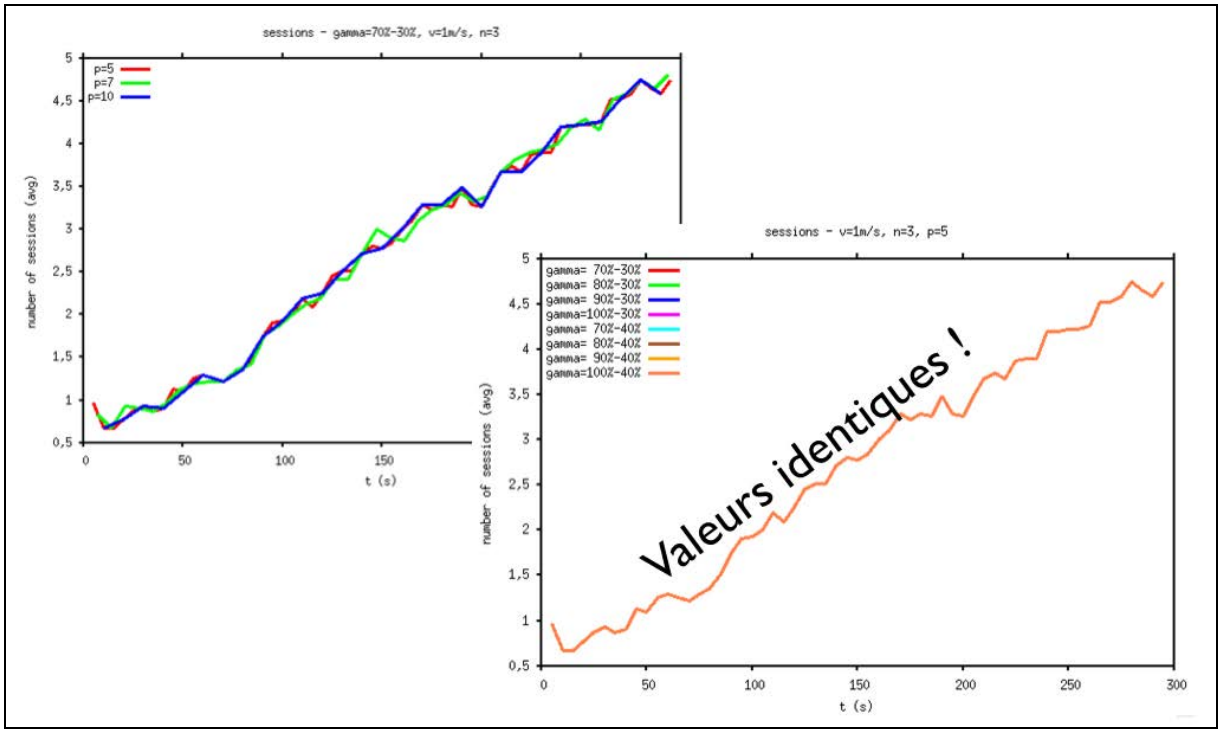


Figure 4-17 : Nombre de sessions en fonction des taux seuils Γ et γ.

La variation du nombre de session est linéaire et croissante dans le temps. Ce qui est logique vu l'évolution du réseau qui est de type dispersion. C'est à dire que les nappes de pétrole ont tendance à devenir des gouttelettes.

4.8.2.1.2 Nombre de nœuds hors sessions

La Figure 4-18 montre la variation du nombre de nœuds hors sessions en fonction de la vitesse de déplacement des nœuds et le nombre minimal (n) des nœuds représentatifs d'une session. Les valeurs des taux seuils Γ et γ sont fixées respectivement à 70% et 30%. Pour chaque valeur de vitesse égale à 1m/s et 2m/s nous avons fait varier le nombre minimal (n) des nœuds représentatifs d'une session.

Nous constatons que pour une vitesse donnée de 1m/s ou de 2m/s, il n'y a presque pas d'influence du nombre (n) sur le nombre de nœuds hors sessions. Alors que l'influence de la vitesse est conséquente au delà des 250 secondes, nous remarquons que le nombre de nœuds hors sessions a presque triplé.

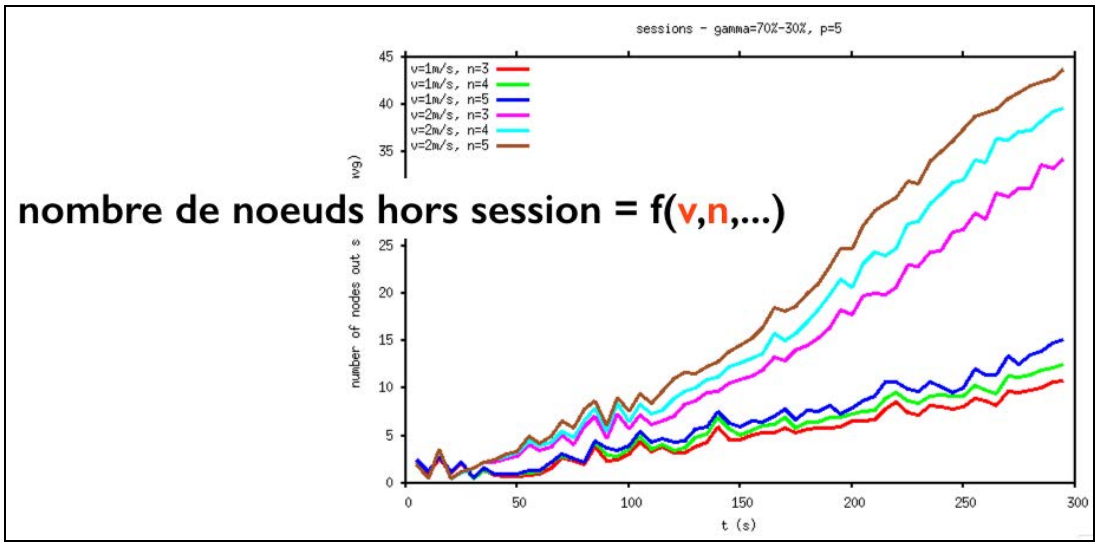


Figure 4-18: Nombre de nœuds hors sessions en fonction de la vitesse et n.

Sur la Figure 4-19 nous présentons la variation du nombre de nœuds hors session en fonction des taux seuils Γ et γ pour une vitesse de déplacement des nœuds de l'ordre de 1m/s et un nombre minimal de nœuds représentatifs d'une session n= 3. Les taux seuils Γ et γ varient respectivement de {100%, 90%, 80%,70%} et {50%, 40%, 30%}.

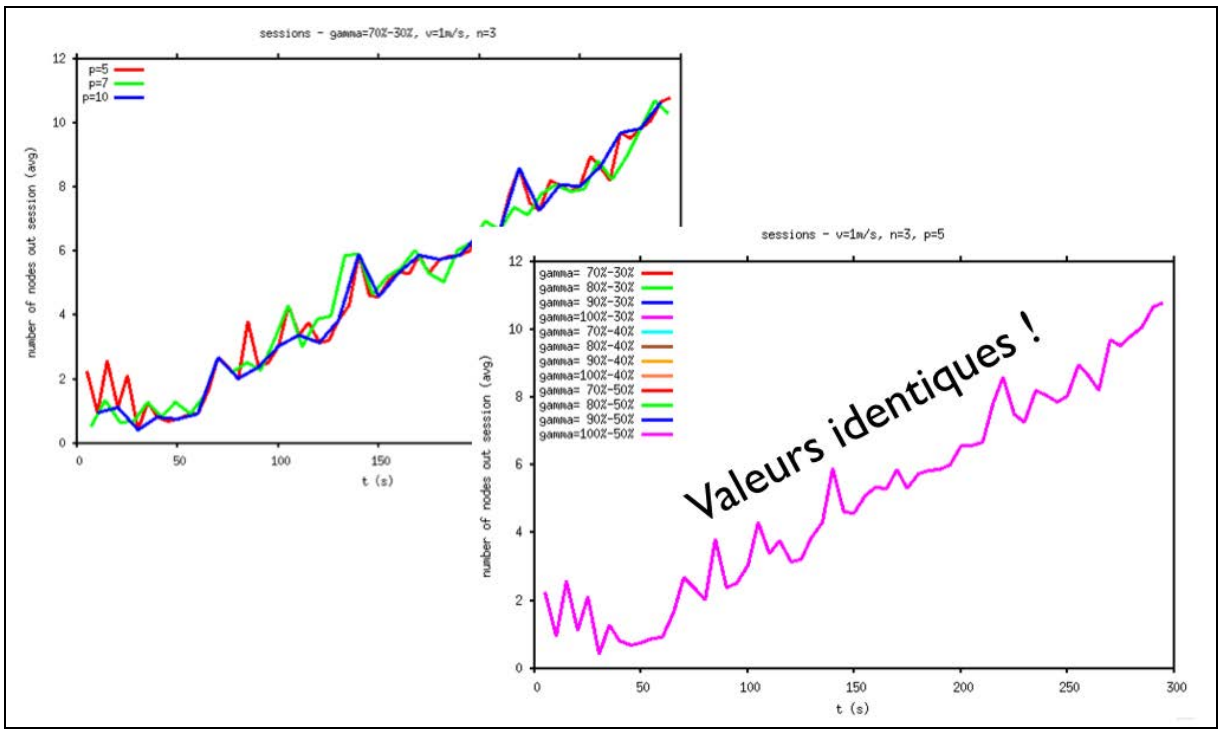


Figure 4-19 : Nombre de nœuds hors sessions en fonction de Γ et γ.

D'après les courbes obtenues, nous observons que les taux seuils Γ et γ n'influent pas sur le nombre de nœuds hors sessions et donc pas sur l'existence d'une session. Par contre, nous constatons qu'ils influent bien sur le type d'héritage.

4.8.2.1.3 Nombre de relations

Pour des valeurs de vitesse de déplacement des nœuds de 1m/s et 2m/s et comme le montre la Figure 4-20, nous avons relevé la variation du nombre des relations d'héritage partiel et total en fonction du nombre minimal des nœuds représentatifs d'une session. Ce nombre (n) prend les valeurs 3, 4 et 5 pour chaque valeur de vitesse.

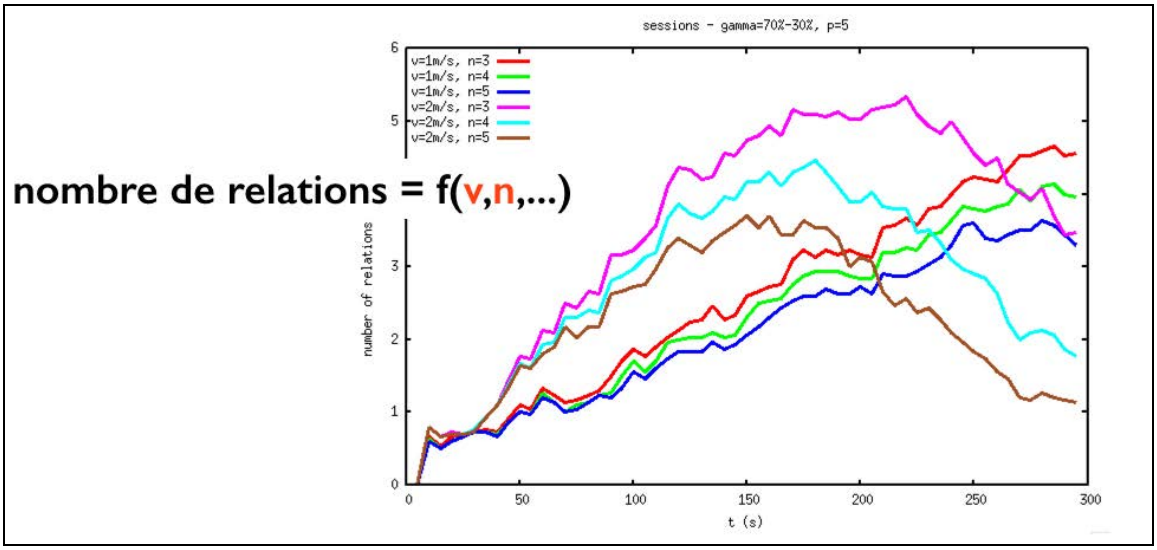


Figure 4-20 : Nombre de relations en fonction de la vitesse et n.

Ensuite, nous nous sommes intéressés à relever l'impact des taux seuils Γ et γ sur le nombre de relations pour un nombre (n) égale à 3 et une vitesse de 1m/s (Figure 4-21) et sur la Figure 4-22 pour les valeurs de vitesse 1m/s et 2m/s.

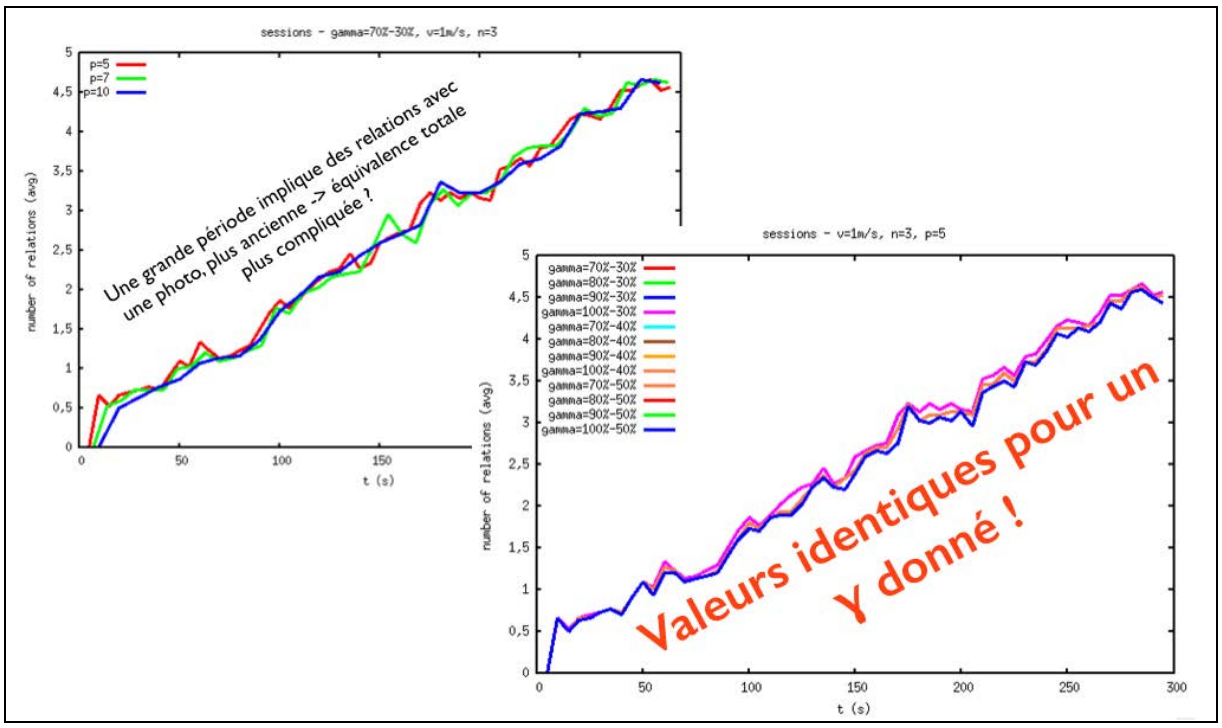


Figure 4-21 : Nombre de relations en fonction des taux seuils Γ et γ.

D'après ces relevés, nous observons qu'il y a peu d'impact des taux seuils Γ et γ sur le nombre de relations.

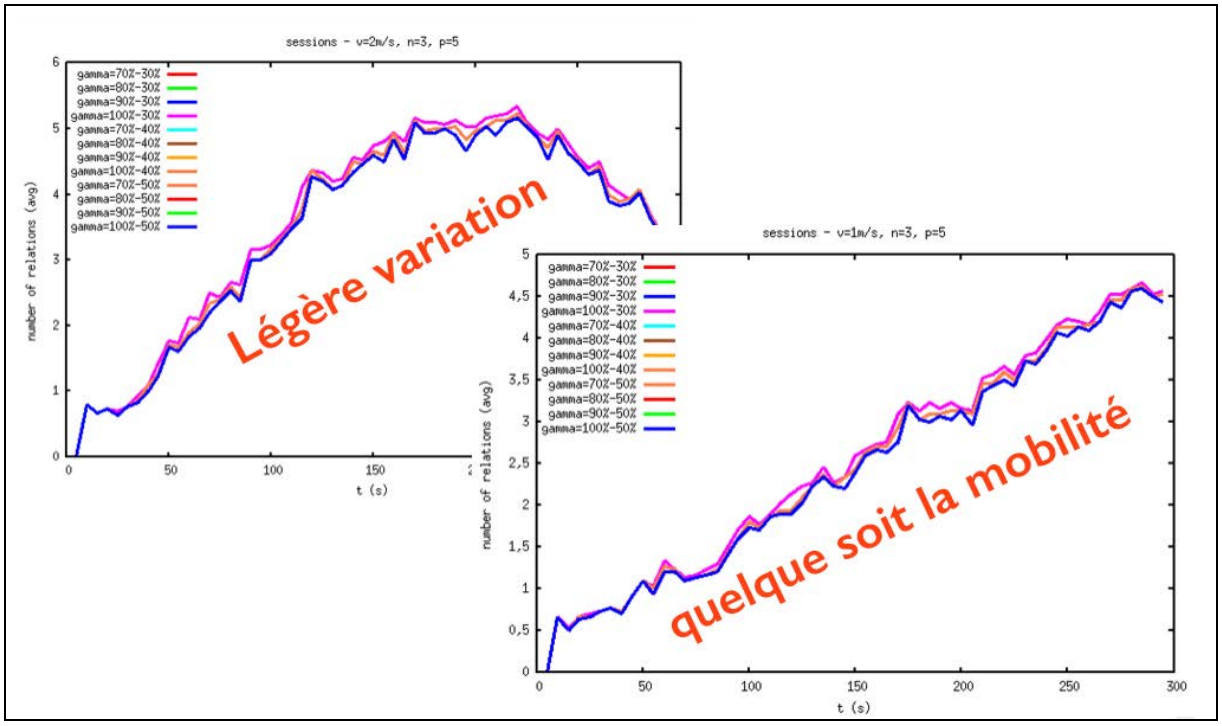


Figure 4-22 : Nombre de relations en fonction de la vitesse et les taux seuils Γ et γ.

En effet et comme l'illustre la Figure 4-23 et la Figure 4-24 pour une équivalence (Γ-ε << γ), nous avons forcément un héritage partiel et donc le nombre de relations reste identique. De même, plus nous avons de sessions, plus il peut y avoir de relations et vice-versa.

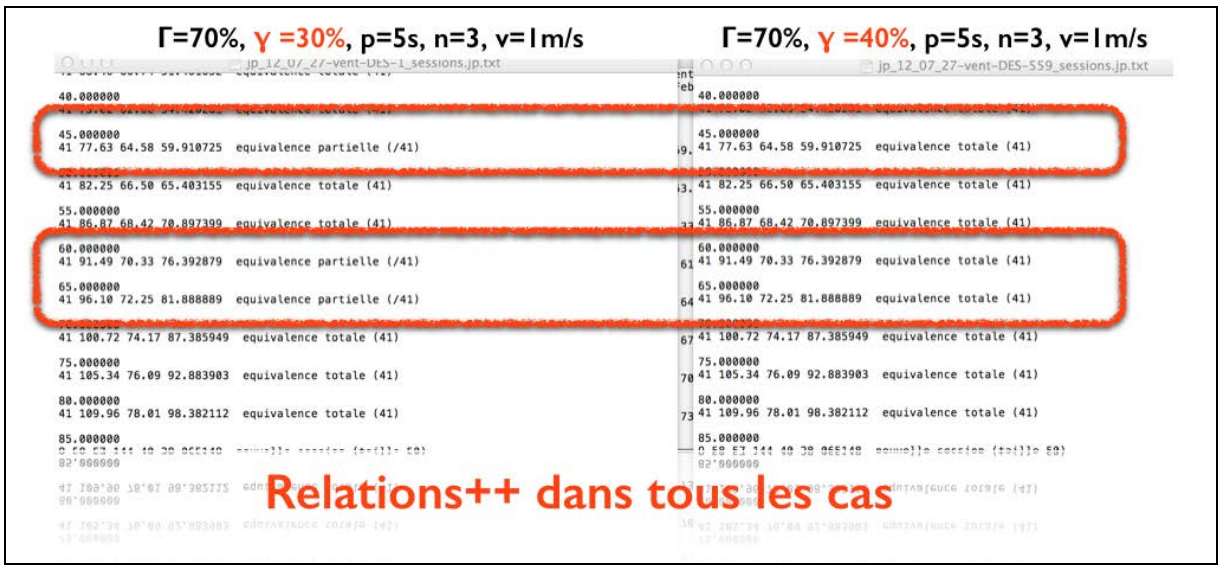


Figure 4-23 : Cas de relations d'équivalences partielles et totales

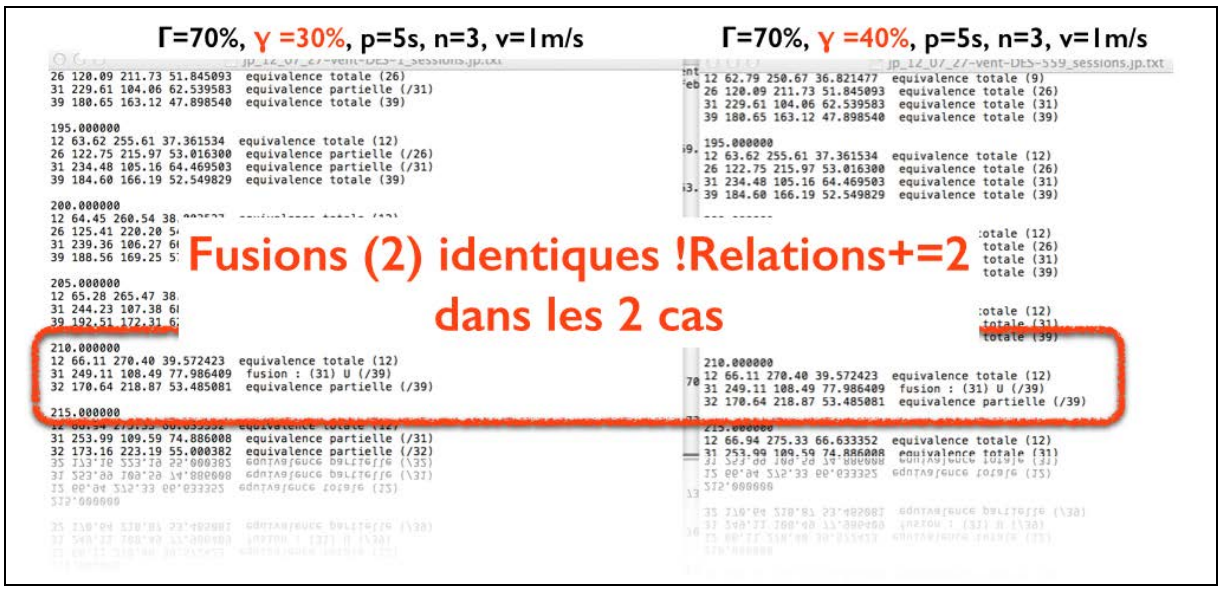


Figure 4-24 : Cas de relation totale type fusion

En ce qui concerne l'influence des taux seuils Γ et γ sur les métriques relevées, nous concluons que :

Le taux seuil γ influe sur le nombre de nouvelles sessions. Si γ est trop grand il y a risque qu'une relation ne soit pas détectée et donc que l'héritage ne soit pas enregistré et que la session soit qualifiée de nouvelle. Par conséquent nous aurons une perte de contexte !

Nous avons remarqué aussi qu'il y a une sensibilité du taux seuil γ par rapport à la taille d'une session et par conséquent au rapport (nombre de nœuds représentatifs d'une session/ densité des nœuds) (30% de 3 équivaut à 1 nœud et 40% ou 50% de 3 équivaut à 2 nœuds, soit une différence d'un seul nœud).

Nous remarquons également que le nombre de nouvelles sessions peut donner un indice de la qualité du suivi de la dynamique de notre réseau. En effet, plus il y a de nouvelles sessions, c'est à dire « si l'on ne retrouve pas au moins un ensemble de nœuds représentatifs d'au moins γ% d'une autre session », plus il y a perte de contexte.

De même, nous pensons :

- que Γ n'a pas ou peu d'influence sur les différentes métriques évoquées précédemment, car il modifie la nature des relations d'héritage (partiel ou total) sans en modifier le nombre.
- que le seuil γ considéré jusqu'à maintenant constitue un seuil relatif à la taille de l'ancien groupe. En effet, dans le cas d'un gros groupe, un nombre important de nœuds peut ne pas hériter (même s'il dépasse largement la taille minimale requise pour former un groupe). On créée alors de nouvelles sessions donc on perd du contexte.

Nous allons donc procéder à une reformulation de l'héritage qui ne se limite donc plus qu'à une et une seule expression prenant en compte les deux remarques précédentes :

$$Gi \in S(k-1) \text{ hérite de } Gj \in S(k) <=> \exists\, g \subset Gi \,/\, g \subset Gj \text{ et } |g| \geq \xi.n$$

ξ représente la variation du nombre de transfuges autour de la taille minimale pour former un groupe. Il est positif et peut être inférieur à 1 pour éviter de créer une nouvelle session lorsque des transfuges de différents groupes fusionnent pour en former un nouveau (double héritage).

De nouvelles simulations vont maintenant être conduites afin de voir l'intérêt de cette réforme de l'héritage et l'effet de ξ.

4.8.2.2 Nouveaux résultats

Suite à cette nouvelle reconsidération des relations d'héritage, nous avons réalisé des séries de simulations pour mieux ressortir l'effet du facteur ξ sur la qualité du suivi du phénomène observé.

Durant les simulations, nous avons maintenu les mêmes paramètres de simulations utilisés auparavant (Voir paragraphe 4.8.1)

Nous avons réalisé des simulations pour deux valeurs de vitesse de déplacement des nœuds : 1m/s et 2m/s. Le but est de voir l'influence de la dynamique du réseau sur les métriques considérées. Les résultats obtenus montre que l'allure des courbes est presque la même sauf que la pente des courbes à 2m/s est plus importante. Ceci s'explique par le fait que plus la dynamique du réseau est importante, plus les nœuds ont tendance à s'isoler et donc à se retrouver hors session.

4.8.2.2.1 Nombre de nœuds hors sessions

Nous pouvons observer à travers les figures Figure 4-25 et Figure 4-26 que le facteur ξ a peu d'effet sur le nombre moyen de nœuds hors sessions.

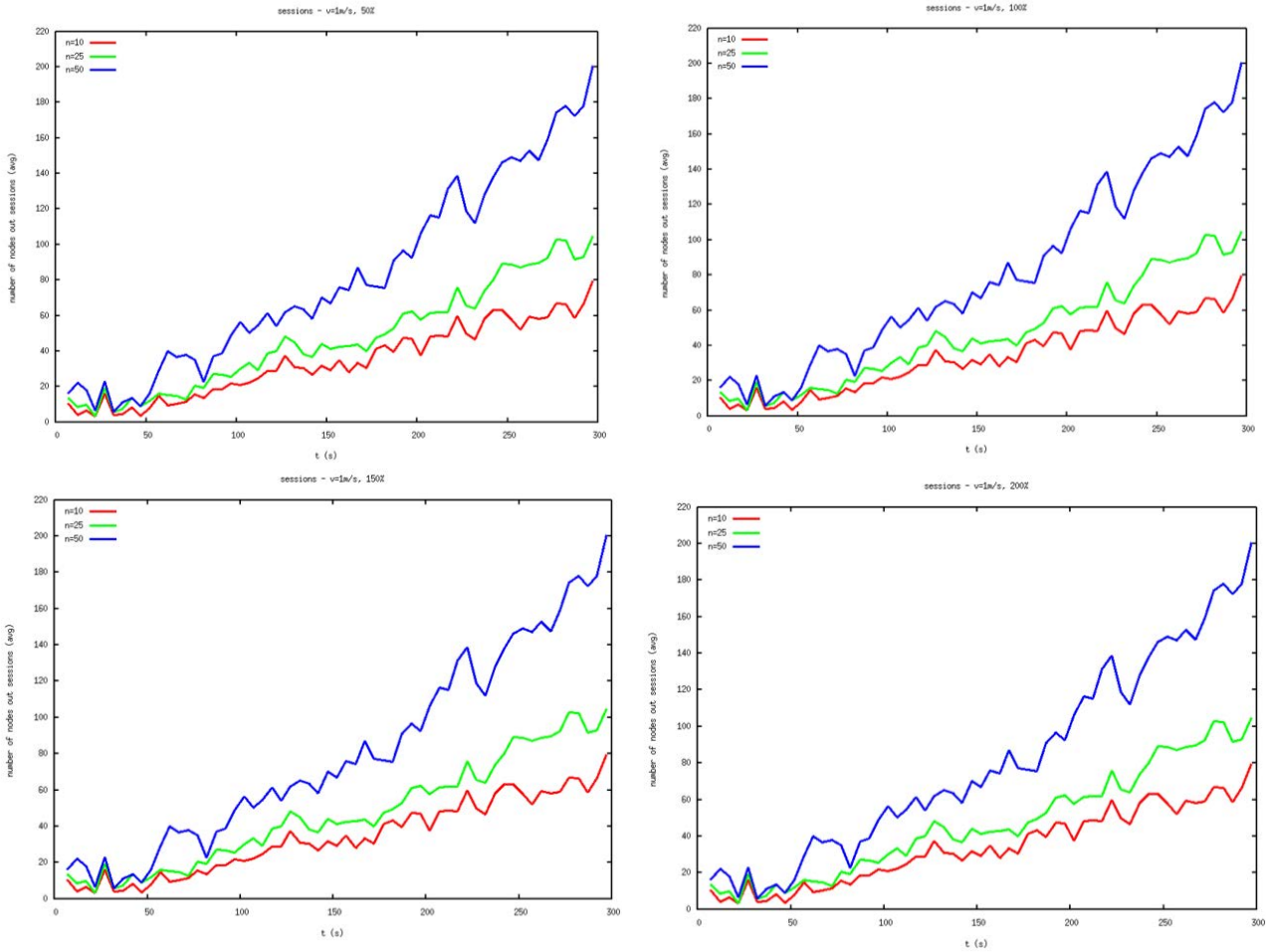


Figure 4-25 : Nombre moyen de nœuds hors sessions pour $n \in \{10, 25, 50\}$; V= 1m/s et $\xi \in \{50\%, 100\%, 150\%, 200\%\}$

Seul le nombre (n) qui affecte ce nombre de nœuds hors sessions, nous remarquons toujours que plus le nombre (n) est grand, plus le nombre de nœuds hors sessions est grand.

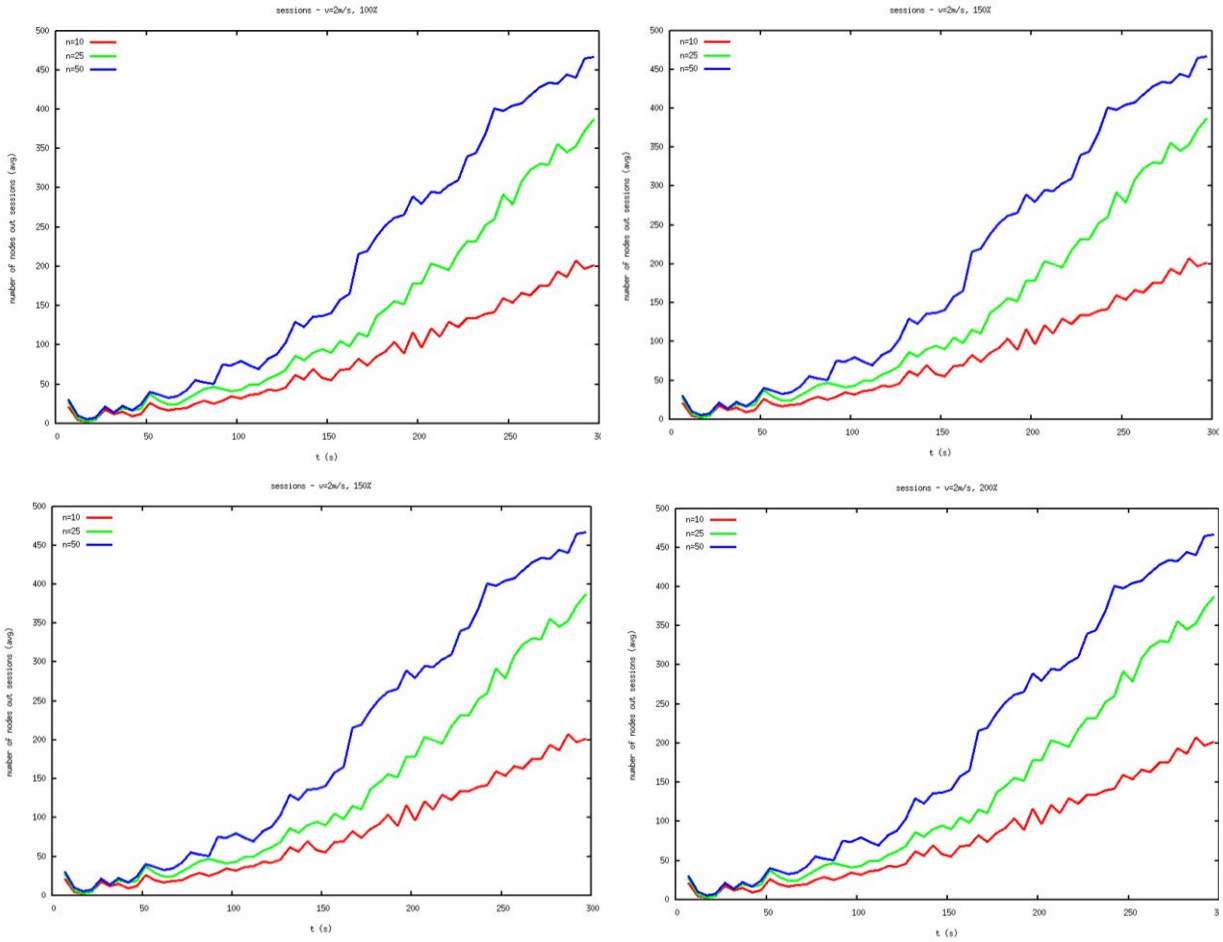


Figure 4-26 : Nombre moyen de nœuds hors sessions pour $n \in \{10, 25, 50\}$; V= 2m/s et $\xi \in \{50\%, 100\%, 150\%, 200\%\}$

Pour avoir un nombre moyen de nœuds hors session acceptable (valeur déterminé par l'utilisateur), nous pouvons avoir un nombre (n) qui ne satisfait pas la taille minimale d'un groupe et par conséquent toutes les informations captées par ces nœuds seront perdues. Ce qui porte préjudice à la finesse de reproduction du contexte.

4.8.2.2.2 Nombre de sessions

La Figure 4-27 présente, à une vitesse de déplacement V= 1m/s, l'évolution du nombre de sessions en fonction du temps pour différentes valeurs du nombre (n) ($n \in \{10, 25, 50\}$) et différents pourcentages du facteur (ξ) $\xi \in \{50\%, 100\%, 150\%, 200\%\}$.

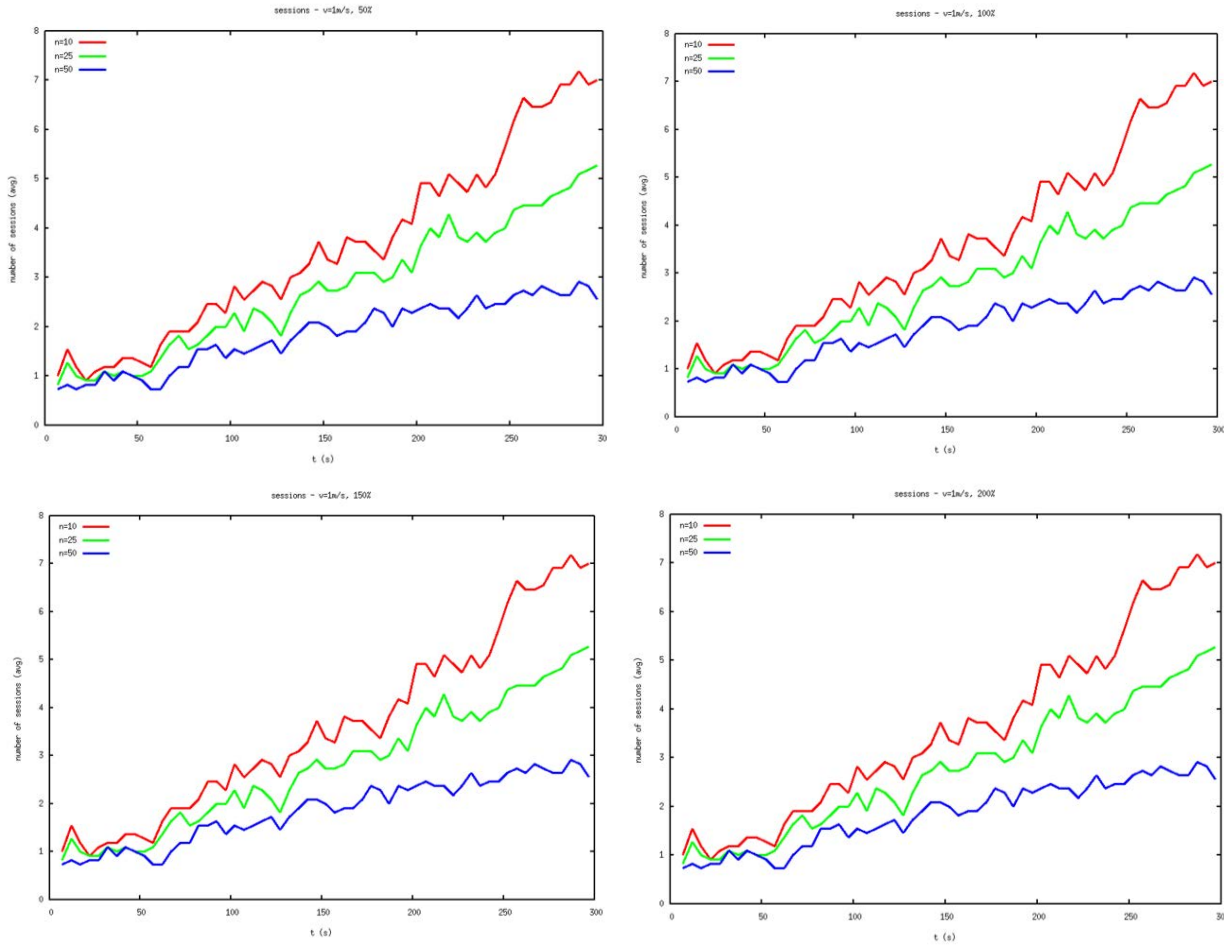


Figure 4-27 : Nombre moyen de sessions pour $n \in \{10, 25, 50\}$; V=1m/s et $\xi \in \{50\%, 100\%, 150\%, 200\%\}$

D'après les courbes obtenues, nous remarquons que le facteur (ξ) n'a pas d'effet sur l'évolution du nombre de sessions détectées. La pente des courbes est invariante quelle que soit la valeur du facteur (ξ). Alors que le nombre (n) influe sur le nombre de sessions. Plus le nombre (n) est grand, plus le nombre de sessions est petit.

4.8.2.2.3 Nombre de nouvelles sessions

Nous avons pu évaluer aussi le nombre moyen des sessions nouvellement créées en fonction du temps en variant le nombre (n) $(n \in \{10, 25, 50\})$ et le facteur ξ $(\xi \in \{50\%, 100\%, 150\%, 200\%\})$ pour deux vitesse V=1m/s (**Erreur ! Source du renvoi introuvable.**) et V=2m/s (Figure 4-29)

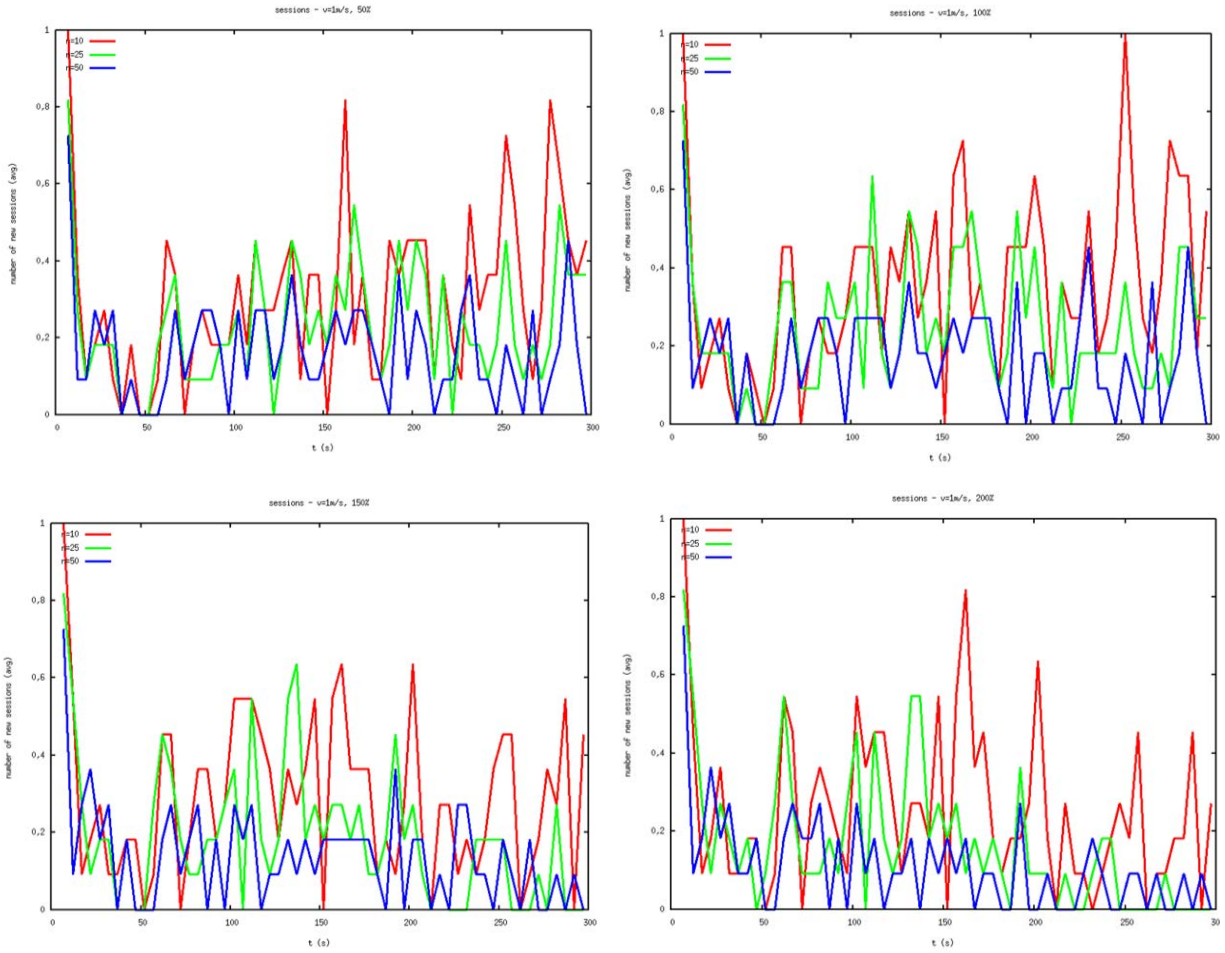


Figure 4-28 : Nombre moyen des nouvelles sessions pour $n \in \{10, 25, 50\}$; V= 1m/s et $\xi \in \{50\%, 100\%, 150\%, 200\%\}$

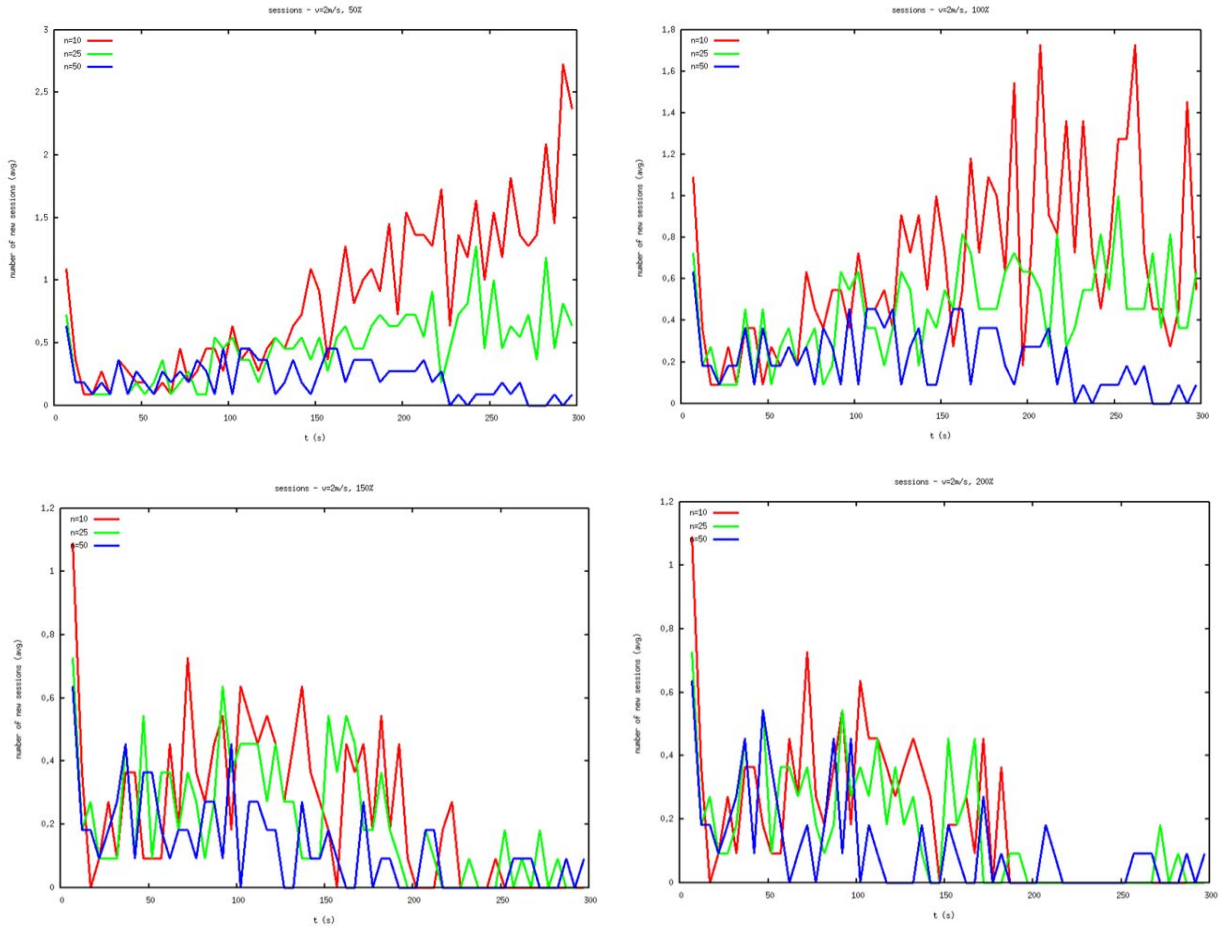


Figure 4-29 : Nombre moyen de nouvelles sessions pour $n \in [10, 25, 50]$; V= 2m/s et $\xi \in [50\%, 100\%, 150\%, 200\%]$

Les figures (Figure 4-27, Figure 4-28, Figure 4-29) montrent la variation du nombre de sessions nouvellement créées en fonction du facteur ξ. Les simulations ont été effectuées pour des valeurs du nombre (n) égales à 10, 25 et 50 nœuds. Et à une vitesse de déplacement des nœuds de l'ordre de 2m/s.

Nous constatons, clairement qu'à la vitesse V=2m/s, que :

- le nombre moyen des nouvelles sessions créées diminue dans le temps. Plus (n) est grand, plus le nombre des nouvelles sessions est petit. Ce résultat est logique du fait que les nappes tendent à devenir des gouttelettes composées d'un nombre (n) trop petit de nœuds pour être considérées comme des sessions.
- pour de grandes valeurs du facteur ξ (150% et 200%), nous remarquons que le nombre des nouvelles sessions diminue dans le temps même pour des valeurs de (n) trop grandes.
- pour de valeurs du facteur ξ égales à 50% et 100%, le nombre de nouvelles sessions créées augmente dans le temps pour des valeurs de (n) petite et diminue pour les grandes valeurs de

(n). Ce qui représente le résultat recherché c'est à dire faible perte de contexte. Ce résultat est confirmé sur les figures Figure 4-30, Figure 4-31, Figure 4-32.

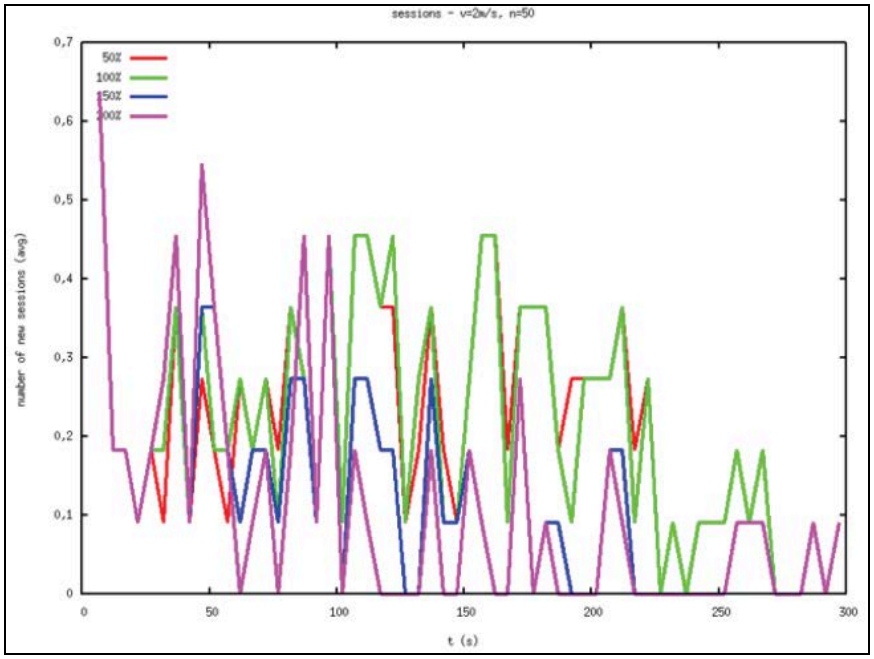


Figure 4-30 : Nombre moyen des nouvelles sessions pour $n = 50\ noeuds$; V= 2m/s et $\xi \in [50\%, 100\%, 150\%, 200\%]$

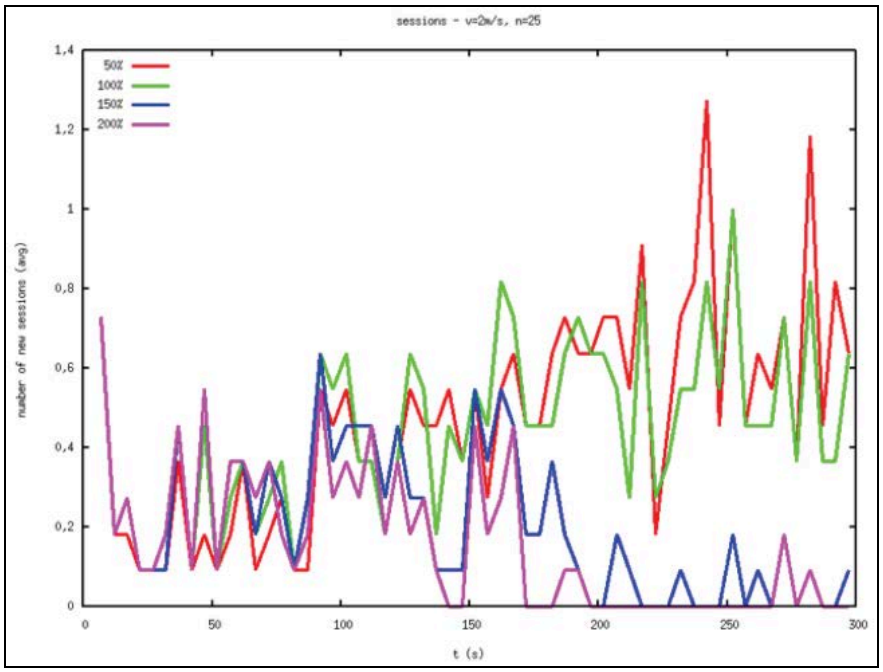


Figure 4-31 : Nombre des nouvelles sessions pour $n = 25\ noeuds$; V= 2m/s et $\xi \in [50\%, 100\%, 150\%, 200\%]$

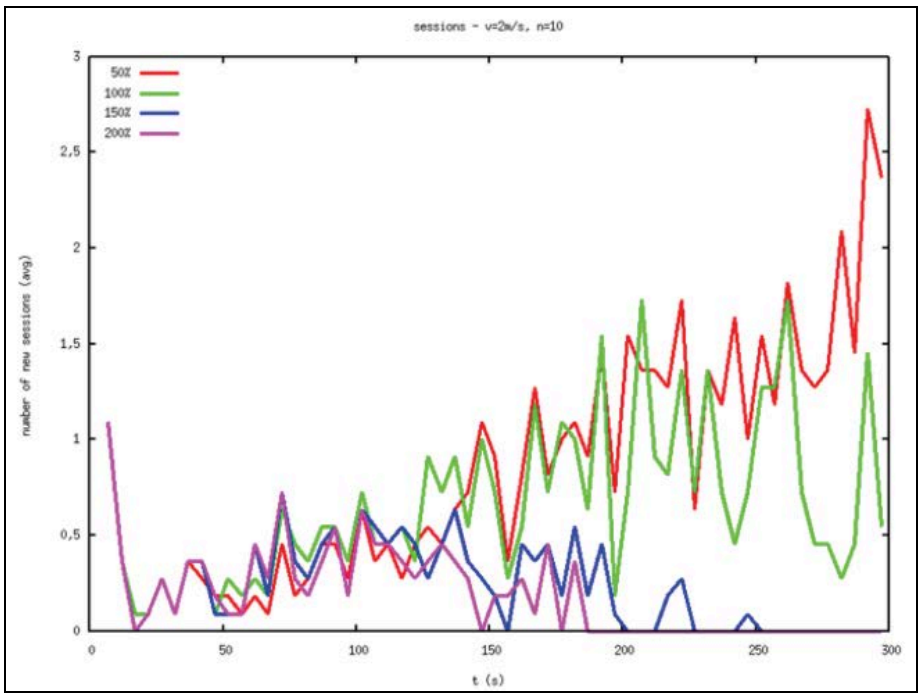


Figure 4-32 : Nombre moyen des nouvelles sessions pour $n = 10$ *noeuds* ; V= 2m/s et $\xi \in [50\%, 100\%, 150\%, 200\%]$

Ces figures montrent que quand le facteur ξ est petit, le nombre de nouvelles sessions créées à tendance à augmenter, ce que nous cherchons pourtant à éviter, puisqu'elles représentent une perte de contexte.

Résumé des observations :

Plus le nombre (n) **grand**, plus

- le nombre moyen de session est **petit**
- le nombre moyen des nœuds hors sessions est **grand**. Donc perte des capacités de mesure du réseau.
- le nombre de nouvelles sessions est **petit**. Donc moins de création de nouvelles sessions pour cause de moins d'héritage non enregistré et par conséquent moins de perte de contexte.

Puisque le facteur ξ pourrait influer sur le suivi de la dynamique du réseau, je pense qu'il sera plus intéressant de voir l'allure des courbes du nombre de relations en variant ce facteur.

Tableau résumant

	Nombre de session	**Nombre de nouvelles sessions**	**Nombre de nœuds hors session**
Le nombre (n)	plus le nombre (n) est grand, plus le nombre de nœuds hors sessions est grand.	Plus (n) est grand, plus le nombre des nouvelles sessions est petit	plus le nombre (n) est grand, plus le nombre de nœuds hors sessions est grand.
Le facteur ξ	Pas d'effet	Pour de valeurs du facteur ξ égales à 50% et 100%, le nombre de nouvelles sessions créées augmente dans le temps pour des valeurs de (n) petite	Pas d'effet

Conclusion :

Ces figures montrent que quand ξ est petit, le nombre de nouvelles sessions à tendance à augmenter, ce que nous cherchons pourtant à éviter, puisqu'elles représentent une perte de contexte.

Dans les précédents résultats, nous avions démontré que plus n est grand, plus le nombre de nœuds hors sessions augmente, ce que nous cherchons à éviter, puisque l'écart entre monde réel et virtuel augmente par non prise en compte d'une partie importante des capacités de mesure du réseau de capteurs.

Comme un groupe n'hérite que si sa taille satisfait $|g| \geq \xi.n$, il faut trouver un compromis entre le suivi statique du phénomène physique porté par n, et le suivi de la dynamique de ce même phénomène porté par ξ.

Pour gérer cette contradiction l'exploitant du système d'information devra non seulement alimenter le simulateur avec les paramètres correspondant à la situation précise à surveiller (étendue de la marée noire, conditions météorologiques, …), mais aussi raffiner ses exigences en termes de finesse d'observation (taille de plus petite nappe que l'on souhaite observer), pourcentage du nombre total de nœuds du réseau dont on veut prendre en compte les mesures, …, et enfin définir les périodes de visite, la tolérance à la quantité d'information non contextualisée, …

Tout ce paramétrage préalable permettra de trouver par simulation des valeurs optimales de n et ξ pour une catastrophe donnée.

4.9 Conclusion

Ce chapitre a fait l'objet de la présentation de notre troisième contribution dans cette thèse à savoir un protocole de session sous ses deux versions centralisée et décentralisée. Ainsi, la première partie de ce chapitre a été consacrée à la définition de la notion de session, des attributs de session et les indicateurs de sessions que nous avons exploités dans ce chapitre.

Dans la deuxième partie de ce chapitre, nous avons présenté la version centralisée de l'algorithme de session proposé. Cet algorithme a été placé au niveau du nœud photographe que nous avons développé. Ce nœud photographe nous a permis d'avoir des vues virtuelles (photos) à des instants donnés de l'évolution des nappes de pétrole. Et par conséquent, il nous a donné la possibilité de tester la détection des nouvelles sessions et celles héritées des sessions précédentes. Les outils de détection de fusion/scission des nappes de pétrole sur lesquels repose l'algorithme ont été présentés aussi, dans ce chapitre.

La version décentralisée de l'algorithme de session proposée a fait l'objet de la troisième partie. Cet algorithme a été implémenté au niveau du collecteur mobile. Et par conséquent, les messages d'échange entre le collecteur mobile et le chef de cluster visité ont été présentés. Différents diagrammes, tels les diagrammes d'états fini du collecteur mobile et du chef de cluster, le diagramme de séquence et celui de Use case ont été également présentés.

Une évaluation de notre protocole par une série de simulations sous l'environnement Opnet a clôturé cette troisième partie. Cette évaluation nous a permis d'énumérer les différents facteurs influant en montrant leurs effets sur le bon fonctionnement de notre protocole. Et ainsi, conclure qu'il existe un compromis entre les taux de seuil et la taille de session. Plus la taille de session est petite, plus il y a création de nouvelles sessions et par conséquent il y a perte du contexte. Ceci influe évidement sur la qualité du suivi de la dynamique de notre réseau et plus concrètement sur le suivi de l'évolution des nappes de pétrole en mer.

Toutes ces constatations doivent permettre de dimensionner le réseau à déployer en cas de catastrophe pour espérer une bonne surveillance de la marée noire.

Chapitre 5 Conclusion et perspectives

Grace aux grands pas effectués ces dernières années dans le développement technologique et plus particulièrement la microélectronique et les techniques de communication sans fil, des petits capteurs communicants en réseau et peu onéreux sont de plus en plus utilisés dans l'industrie et les applications de l'observation de l'environnement. Toutefois, l'utilisation des réseaux de capteurs sans fil dans de telles applications doit faire face à plusieurs limites imposées par les capteurs telles la capacité de traitement, la petite taille de mémoire et l'énergie. Ou les limites imposées par le réseau lui-même telles l'étroite bande passante, la dynamique du réseau due à la variation topologique du réseau et les protocoles de communications adéquats et adaptés à ce type de réseau.

Cette thèse a porté sur l'élaboration d'un protocole de communication de type session permettant de recréer un contexte de collecte des données dans un environnement dynamique présentant des ruptures de communications. L'objectif visé est d'essayer de maintenir une connectivité discrète, partielle ou complète sur ce type de réseau dynamique.

Synthèse

Dans la première partie de cette thèse et suite à un état de l'art sur les architectures de collecte de données utilisées dans des projets très connus mondialement exploitants les réseaux de capteurs sans fil, nous avons choisi pour notre système du suivi des nappes de pétrole en mer, une architecture de collecte de données à trois niveaux.

Le choix d'une telle architecture a été imposé par les spécificités du contexte applicatif choisi. Ce contexte est caractérisé par une mobilité individuelle des nœuds capteurs et une mobilité du groupe des nœuds capteurs formant les nappes de pétrole. Cette mobilité induit une variation continue de la topologie du réseau.

La solution de clustering adoptée au niveau 1 de cette architecture consiste à former des groupes de nœuds disjoints représentés par un chef. L'élection du chef de groupe est réalisée selon une métrique exploitant l'énergie, la densité et la mobilité des nœuds capteurs. L'originalité première de cette solution, est due au fait qu'on peut faire varier le facteur de pondération α selon le niveau de stabilité désiré. Ce dernier est défini en termes de nombre de clusters formés, et du temps qu'un nœud passe à l'état chef de cluster. En outre, la mobilité des nœuds capteurs permet d'éviter l'élection de chefs de clusters pour une longue durée afin de ne pas les pénaliser en consommation d'énergie. Enfin, plus le nombre de clusters formés est faible, plus le circuit de visite du collecteur mobile aux chefs de clusters est court et ainsi la qualité de la collecte est meilleure, mais au détriment de la finesse d'observation.

Aussi, le modèle du nœud capteur développé sous l'environnement de simulation OPNET, permet en plus de réaliser nos simulations, de concevoir le système de monitoring à déployer en termes de nombre de nœuds, de période de mesure, de dynamique du clustering et de poids de la métrique en prenant en compte la surface de zone à surveiller et les conditions climatiques.

La seconde partie de cette thèse a été consacrée au protocole de session. Guidés par l'objectif de maintenir une connectivité discrète partielle ou complète sur un réseau dynamique, nous avons dû faire face à plusieurs contraintes. La première contrainte était la découverte de la structure clustérisée du réseau durant la durée de la collecte des données captées. C'est à dire détecter la formation de nouveaux clusters, la fusion ou la scission d'anciens clusters.

Ainsi, un algorithme de session sous sa version centralisé a été placé au niveau d'un nœud que nous avons nommé « photographe ». Cet algorithme nous permet de donner des vues (photos) à des instants donnés des nappes de pétrole dispersées en mer. Et en même temps nous permet d'évaluer la validité des relations d'équivalence totale et partielle que nous avons définies.

Ensuite, nous avons détaillé le fonctionnement du protocole de session en implémentant l'algorithme sous sa version décentralisée au niveau du collecteur mobile. Cet algorithme gère la collecte des données selon deux régimes : un régime initial et un régime permanent. Les simulations montrent enfin comment en peut dimensionner le système d'observation à déployer en cas de catastrophe.

Perspectives

Les travaux que nous avons réalisés et les voies d'investigation explorées dans le cadre de cette thèse nous ouvrent de multiples perspectives de recherche. Nous organisons nos réflexions comme suit :

- Le déploiement de la solution proposée dans un environnement réel d'exploitation. Nous pensons que l'expérimentation permettrait de déceler et de mieux cerner les aléas des communications et du déploiement afin d'en tenir compte dans le modèle du nœud capteur proposé.

- L'utilisation de modèles de mobilité et de propagation plus réalistes. Nous pensons qu'il est plus intéressant de tester nos propositions en s'appuyant sur

    - des modèles de mobilité prenant en compte la mobilité du groupe (mobilité des nappes) et la mobilité individuelle (mobilité des nœuds).

    - des modèles de propagation tenant compte des interférences.

- La consommation énergétique de notre protocole de session que nous n'avons pas pu évaluer pour cause de temps de simulation trop grand et données de mesure à sauvegarder trop grande. Une machine parallèle puissante est donc nécessaire pour pousser plus loin les simulations.

- Le déploiement de plusieurs collecteurs mobiles dans le réseau. Nous pensons que la collaboration de plusieurs collecteurs mobiles dans le réseau pourrait apporter beaucoup d'avantages et améliorerait le temps de collecte des données. Notamment en adoptant une solution d'équilibrage de charge entre les différents collecteurs mobiles dans le processus de collecte.

Bibliographie

Abbasi, A.A. et Younis, M., 2007. A survey on clustering algorithms for wireless sensor networks. *Computer Communications*, 30, pp.2826-41.

Akkaya, K. et Younis, M., 2005. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3, pp.325-49.

Akyildiz, I.F., Su, W., Sankarasubramaniam, Y. & Cayirci, E., 2002. Wireless sensor networks: a survey. *Computer Networks*, 38, pp.393-422.

Albert, R. et Barabàsi, A.-L., 2002. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74, pp.47--97.

An, B. et Papavassiliou, S., 2001. A mobility-based clustering approach to support mobility management and multicast routing in mobile ad-hoc wireless networks. *Int. J. Netw. Manag.*, 11, pp.387-95. Available at: HYPERLINK "http://dx.doi.org/10.1002/nem.415" http://dx.doi.org/10.1002/nem.415 .

Arboleda, L.M.C. et Nasser, N., 2006. Comparison of Clustering Algorithms and Protocols for Wireless Sensor Networks. In *Electrical and Computer Engineering, 2006. CCECE '06. Canadian Conference on.*, 2006.

Atay, N. et Bayazit, O.B., 2009. Mobile Wireless Sensor Network Connectivity Repair with K-Redundancy. *Algorithmic Foundation of Robotics VIII*, 57, pp.35-49.

Baker, D.J., 1981. A distributed algorithm for organizing mobile radio telecommunication networks. In *International Conference on Distributed Computing Systems.*, 1981.

Baker, D.J., 1981. The architectural organization of a mobile radio network via a distributed algorithm. 29 , Issue: 11, pp.1694-701.

Bandyopadhyay, E.J.S., 2003. An energy efficient hierarchical clustering algorithm for wireless sensor networks. In *Proc. INFOCOM 2003. Twenty-Second Annual Joint Conf. of the IEEE Computer and Communications. IEEE Societies.*, 2003.

Barabási, A.-L. et Albert, R., 1999. Emergence of Scaling in Random Networks. *Science*, 286, pp.509--512.

Basagni, S., 1999. Distributed Clustering for Ad Hoc Networks. *Parallel Architectures, Algorithms, and Networks, International Symposium on*, 0, p.310.

Basagni, S. et al., 2008. Controlled sink mobility for prolonging wireless sensor networks lifetime. *Wirel. Netw.*, 14, pp.831-58. Available at: HYPERLINK "http://dx.doi.org/10.1007/s11276-007-0017-x" http://dx.doi.org/10.1007/s11276-007-0017-x .

Basagni, S. et Jonsson, E., 1999b. *Distributed and Mobility-Adaptive Clustering for Ad Hoc Networks*. Tech. rep.

Basu, P., Khan, N. & Little, T.D.C., 2001. A Mobility Based Metric for Clustering in Mobile Ad Hoc Networks. In *International Workshop on Wireless Networks and Mobile Computing (WNMC2001)*., 2001.

BenSaad, L. et Tourancheau, B., 2009. Multiple Mobile Sinks Positioning in Wireless Sensor Networks for Buildings. Washington, DC, USA, 2009. IEEE Computer Society.

Bouhafs, F., Merabti, M. & Mokhtar, H., 2006. A semantic clustering routing protocol for wireless sensor networks., 2006.

Broch, J. et al., 1998. A performance comparison of multi-hop wireless ad hoc network routing protocols. New York, NY, USA, 1998. ACM.

Bulusu, N., Estrin, D., Girod, L. & Heidemann, J., 2001. Scalable coordination for wireless sensor networks: self-configuring localization systems. In *in Proc. 6th International Symposium on Communication Theory and Applications (ISCTA '01), Ambleside, Lake District*., 2001.

Caillouet, C., P & Rivano, H., 2011. Framework for optimizing the capacity of wireless mesh networks. *Comput. Commun.*, 34(13), pp.1645-59.

Carter, 1981. Time delay estimation for passive sonar signal processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pp.463-70.

Chakrabarti, A., Sabharwal, A. & Aazhang, B., 2003. Using predictable observer mobility for power efficient design of sensor networks. In *Proceedings of the 2nd international conference on Information processing in sensor networks*. Berlin, Heidelberg, 2003. Springer-Verlag.

Chan, Y.T.a.H.K.C., 1994. A simple and efficient estimator for hyperbolic location. *IEEE Transactions on Signal Processing*, pp.1905-15.

Chatterjee, M., Das, S.K. & Turgut, D., 2001. WCA: A Weighted Clustering Algorithm for Mobile Ad hoc Networks. *Journal of Cluster Computing (Special Issue on Mobile Ad hoc Networks*, 5, pp.193-204.

Cheng, D.K., 1989. *Field and Wave Electromagnetics (2nd Edition)*. 2nd ed. Addison-Wesley.

Chen, W.p., Hou, J.C. & Sha, L., 2004. Dynamic clustering for acoustic target tracking in wireless sensor networks. *IEEE_J_MC*, 3, pp.258-71.

Ciciriello, P., Mottola, L. & Picco, G.P., 2007. Efficient routing from multiple sources to multiple sinks in wireless sensor networks. In *Proc. of the 4th European Conference on Wireless Sensor Networks (EWSN)*., 2007.

Clark, B.N., Colbourn, C.J. & Johnson, D.S., 1991. Unit disk graphs. *Discrete Math.*, 86(1-3), pp.165--177.

Correal, N.P. et al., 2005. Locating the Nodes -- Cooperative localization in wireless sensor networks. *IEEE Signal Processing Magazine*, 22(4), pp.54-69.

CRUISE, W.D.1., 2006. Report on WSN applications, their requirements, application-specific WSN issues and evaluation metrics. Available at: HYPERLINK "http://www.istcruise.eu" http://www.istcruise.eu .

Dai, F. et Wu, J., 2004. An Extended Localized Algorithm for Connected Dominating Set Formation in Ad Hoc Wireless Networks. *IEEE Trans. Parallel Distrib. Syst.*, 15(10), pp.908-20.

Dai, F. et Wu, J., 2006. On constructing k-connected k-dominating set in wireless ad hoc and sensor networks. *J. Parallel Distrib. Comput.*, 66(7), pp.947-58.

Doan, C.Q. et Wood, A., 2005. Wireless sensor networks for in-home healthcare: Potential and challenges., 2005.

Ekici, E., Gu, Y. & Bozdag, D., 2006. Mobility-based communication in wireless sensor networks. *Communications Magazine, IEEE*, 44(7), pp.56-62.

Ephremides, A., Wieselthier, J.E. & Baker, D.J., 1987. A design concept for reliable mobile radio networks with frequency hopping signaling. In *Proceedings of the IEEE.*, 1987. Journals \& Magazines.

Erard, P.-J. et Déguénon, P., 1996. *Simulation par événements discrets*. 1st ed. PPUR presses polytechniques.

Erdős, P. et Rényi, A., 1960. On the Evolution of Random Graphs. *publication of the mathematical institute of the hungarian academy of sciences* , pp.17--61.

e-SENSE, .D.2.1.e., 2006. : Scenarios and audio visual concepts. Available at: HYPERLINK "http://www.ist-esense.org" http://www.ist-esense.org .

Fall, K., 2003. A Delay-tolerant Network Architecture for Challenged Internets. *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp.27--34.

Ferreira, A. et Viennot, L., 2002. RR-4403 *A Note on Models, Algorithms, and Data Structures for Dynamic Communication Networks*. Rapport de recherche. INRIA.

Francesco, M.D., Das, S.K. & Anastasi, G., 2011. Data Collection in Wireless Sensor Networks with Mobile Elements: A Survey. *ACM Trans. Sen. Netw.*, 8(1), pp.7:1--7:31.

Gandham, S.R., Dawande, M., Prakash, R. & Venkatesan, S., 2003. Energy efficient schemes for wireless sensor networks with multiple mobile base stations., 2003.

Gao, S., Zhang, H. & Das, S.K., 2011. Efficient Data Collection in Wireless Sensor Networks with Path-Constrained Mobile Sinks. *IEEE Transactions on Mobile Computing*, 10, pp.592-608.

Gezici, S.a.T.Z.a.G.G.B.a.K.H.a.M.A.F.a.P.H.V.a.S.Z., 2005. Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks. *IEEE Signal Processing Magazine*, pp.70-84.

Guo, Y. et al., 2006. Animal Behaviour Understanding using Wireless Sensor Networks. In *1st IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp2006).*, 2006.

Gupta, G. et Younis, M., 2003. Load-balanced clustering of wireless sensor networks. In *Proc. IEEE Int. Conf. Communications ICC '03*., 2003.

Haenggi, M., 2004. Twelve reasons not to route over many short hops. In *Vehicular Technology Conference, 2004. VTC2004-Fall. 2004 IEEE 60th*., 2004.

Handcock, R.N. et al., 2009. Monitoring Animal Behaviour and Environmental Interactions Using Wireless Sensor Networks, GPS Collars and Satellite Remote Sensing. *Sensors*, 9(5), pp.3586-603.

Heinzelman, W.B. et al., 2002. An Application-Specific Protocol Architecture for Wireless Microsensor Networks. *IEEE Transactions on Wireless Communications*, 1, pp.660-70.

Huang, J.-H., Amjad, S. & Mishra, S., 2005. CenWits: a sensor-based loosely coupled search and rescue system using witnesses. New York, NY, USA, 2005. ACM.

Jea, D., Somasundara, A. & Srivastava, M., 2005. Multiple controlled mobile elements (data mules) for data collection in sensor networks., 2005.

Johnson, D.B. et Maltz, D.A., 1996. Dynamic source routing in ad hoc wireless networks., 1996. Kluwer Academic Publishers.

Juang, P. et al., 2002. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet. *SIGOPS Oper. Syst. Rev.*, 36(5), pp.96-107.

Jurčík, P. et Koubâa, A., 2007. *The IEEE 802.15.4 OPNET Simulation Model: Reference Guide v2.0*. Tech. rep. IPP-HURRAY.

Kansal, A. et al., 2004. Intelligent fluid infrastructure for embedded networks. In *Proc. ACM MobiSys'04*., 2004.

Keeney, et Raiffa, H., 1976. *Decisions with multiple objectives: Preferences and value tradeoffs*. J. Wiley, New York.

Klein, A. et Tran-Gia, P., 2007. Energy Consumption Framework for Wireless Sensor Networks. In *OPNETWORK 2007*. Washington D.C., #aug# 2007.

Kondepu, K., Restuccia, F., Anastasi, G. & Conti, M., 2012. A hybrid and flexible discovery algorithm for wireless sensor networks with mobile elements., 2012.

Kone, C.-T., 2011. *Conception de l'architecture d'un réseau de capteurs sans fil de grande dimension*. Thèse. Université de la Lorraine - Nancy- France.

Krishnakumar, A.S. et al., 2004. A System for LEASE: Location Estimation Assisted by Stationery Emitters for Indoor RF Wireless Networks. *IEEE INFOCOM*, pp.1001--1011.

Kumarawadu, P., Dechene, D.J., Luccini, M. & Sauer, A., 2008. Algorithms for Node Clustering in Wireless Sensor Networks: A Survey. In *Information and Automation for Sustainability, 2008. ICIAFS 2008. 4th International Conference on*., 2008.

Lehsaini, M., 2009. *Diffusion et couverture basѵ©es sur le clustering dans les rѵ©seaux de capteurs : application ѵ† la domotique*. Master's thesis.

Li, J. et al., 2009. Connectivity, Coverage and Placement in Wireless Sensor Networks. *Sensors*, 9(10), pp.7664--7693.

Liang, Y. et Li, H., 2007. An Energy-Efficient Clustering Algorithm for Wireless Sensor Network. In *Intelligent Information Hiding and Multimedia Signal Processing, 2007. IIHMSP 2007. Third International Conference on*., 2007.

Li, J. et Mohapatra, P., 2007. Analytical modeling and mitigation techniques for the energy hole problem in sensor networks. *Pervasive Mob. Comput.*, 3(3), pp.233-54.

Luo, H. et al., 2005. TTDD: Two-Tier Data Dissemination in Large-Scale Wireless Sensor Networks. *Wireless Networks*, 11, pp.161-75.

Lu, J.-L., Valois, F., Dohler, M. & Barthel, D., 2008. Quantifying Organization by Means of Entropy. *IEEE Communications Letters*, 12(3), pp.185-87.

Madan, R., Cui, S., Lall, S. & Goldsmith, A., 2006. Cross-Layer Design for Lifetime Maximization in Interference-Limited Wireless Sensor Networks. *Wireless Communications, IEEE Transactions on*, 5, pp.3142-52.

Mainwaring, A. et al., 2002. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. New York, NY, USA, 2002. ACM.

Malpani, N., Welch, J.L. & Vaidya, N., 2000. Leader election algorithms for mobile ad hoc networks. In *Proceedings of the 4th international workshop on Discrete algorithms and methods for mobile computing and communications*. New York, NY, USA, 2000. ACM.

Marin-Perianu, R.S., Hurink, J.L. & Hartel, P.H., 2008. A generalized clustering algorithm for dynamic wireless sensor networks. In *Parallel and Distributed Processing with Applications, 2008. ISPA '08. International Symposium on*. Enschede, September 2008. Centre for Telematics and Information Technology, University of Twente.

Marin-Perianu, R.S. et al., 2007. Tandem: A Context-Aware Method for Spontaneous Clustering of Dynamic Wireless Sensor Nodes. In *Proceedings of the First International Conference on Internet of Things (IOT2008), March 2008*. Zürich, Switzerland, September 2007. Verlag, Lecture Notes in Computer Science (4952). Springer.

Marin-Perianu, R.S., Scholten, J., Havinga, P.J.M. & Hartel, P.H., 2007. Cluster-based service discovery for heterogeneous wireless sensor networks. *International Journal of Parallel, Emergent and Distributed Systems*, 23(TR-CTIT-07-05), pp.325-46. Available at: HYPERLINK "http://doc.utwente.nl/66943/" http://doc.utwente.nl/66943/ .

Mitton, N., 2006. *Auto-organisation des réseaux de capteurs sans fil multi-sauts à grande echelle.* Thèse. Lyon, France: INSA de Lyon.

Mitton, N., Fleury, E., Lassous, I.G. & Tixeuil, S., 2009. Self-stabilization in self-organized multihop wireless networks. *Ad Hoc and Sensor Wireless Networks*, pp.909-15.

Mnif, K. et Kadoch, M., 2006. Construction and Maintenance of Backbone for Routing Protocols Enhancement in Mobile Ad Hoc Networks. In *Proc. 10th IEEE Singapore Int. Conf. Communication systems ICCS 2006*., 2006.

Monteiro, J., Goldman, A. & Ferreira, A., 2006. Performance Evaluation of Dynamic Networks using an Evolving Graph Combinatorial Model. In *Proceedings of WiMob*., 2006.

Naruephiphat, W. et Charnsripinyo, C., 2009. Clustering Techniques in Wireless Sensor Networks. pp.1273-78.

Ochiai, H.a.E.H., 2008. Mobility entropy and message routing in community-structured delay tolerant networks. In ACM, ed. *in Proceedings of the 4th Asian Conference on Internet Engineering, New York, NY, USA*., 2008.

Oyman, E.I. et Ersoy, C., 2004. Multiple sink network design problem in large scale wireless sensor networks. In *Proc. IEEE Int Communications Conf*., 2004.

Padhy, P. et al., 2005. Glacial Environment Monitoring using Sensor Networks. In *Real-World Wireless Sensor Networks*., 2005. Event Dates: June 20-21 2005.

Pentland, A.(.a.F.R.a.H.A., 2004. DakNet: Rethinking Connectivity in Developing Nations. *IEEE Computer Society*, 37, pp.78--83.

Shah, R.C., Roy, S., Jain, S. & Brunette, W., 2003. Data MULEs: modeling a three-tier architecture for sparse sensor networks. *Ad Hoc Networks*, 1(2-3), pp.215-33.

Smith, J.a.A.J., 1987. The spherical interpolation method of source localization. *IEEE Journal of Oceanic Engineering*, 12(1), pp.246--252.

Theoleyre, F. et Valois, F., 2008. A self-organization structure for hybrid networks. *Ad Hoc Netw.*, 6(3), pp.393-407.

Tran, D.-H., Yang, J. & Sattler, K.-U., 2011. Decentralized Change Detection in Wireless Sensor Network Using DFT-based Synopsis. Washington, DC, USA, 2011. IEEE Computer Society.

Vey, Q., Berthou, P. & Gayraud, T., 2013. *Improvement of PEAR signaling in energy efficient Delay Tolerant Wireless Sensor Network*. Rapport LAAS n\degre 13323 Rapport LAAS n\degre 13323.

Vivek, S.S., 2010. Clustering Algorithms for Heterogeneous Wireless Sensor Network: A Survey. *International journal of applied engineering research, DINDIGUL Volume 1, No 2*.

Wang, G., Cao, G., Berman, P. & Porta, T.F.L., 2007. Bidding Protocols for Deploying Mobile Sensors. *IEEE Transactions on Mobile Computing*, 6, pp.563-76.

Werner-Allen, G. et al., 2006. Deploying a wireless sensor network on an active volcano. *Internet Computing, IEEE*, 10(2), pp.18-25.

Wu, J. et Li, H., 1999. On calculating connected dominating set for efficient routing in ad hoc wireless networks. In *Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications*. New York, NY, USA, 1999. ACM.

Xing, G., Wang, T., Jia, W. & Li, M., 2008. Rendezvous design algorithms for wireless sensor networks with a mobile base station. In *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*. New York, NY, USA, 2008. ACM.

Yang, S., Wu, J. & Dai, F., 2007. Efficient Backbone Construction Methods in MANETs Using Directional Antennas. In *Proceedings of the 27th International Conference on Distributed Computing Systems*. Washington, DC, USA, 2007. IEEE Computer Society.

Yoon, J., Liu, M. & Noble, B., 2003. Random Waypoint Considered Harmful. *NFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, pp.1312-21.

Younis, O.e.F.S., 2004. Distributed clustering in ad-hoc sensor networks: a hybrid, energy-efficient approach. *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 1.

Yu, J.Y. et Chong, P.H.J., 2005. A survey of clustering schemes for mobile ad hoc networks. *Communications Surveys Tutorials, IEEE*, 5(1), pp.32-48.

Annexes

Autres fonctions

Fonction	Description
MAJ(LS))	Fonction de mise à jour de la liste des chefs de clusters (LS* ← LS)
MAJ(id(x,CH))	Fonction de mise à jour de l'ensemble des identifiants de sessions
Destruct_param {(Paramètres (id(x,CH*))}	Fonction de destruction de {(Paramètres (id(x,CH*))}
Find_old_CH()	Fonction recherche d'un ancien chef parmi les membres dans V*(CH)
Find_exist_CH()	Fonction recherche d'un chef existant da la liste des chefs de clusters déjà associés à un identifiant de session dans LS*
Long getCreationTime()(	Retourne le temps de la création de la session
Object getAttributes(integer Id)	Retourne l'identifiant de la session, *null* s'il n'existe pas
String getId()	Génère un identifiant de session long
getCreationTime()(	Retourne le temps de la requête précédente pour cette session
Void sup_session()	Supprime la session
Boolean isnew()(	Retourne *true* si la session vient d'être créée, sinon *false*
Void putValue(integer Id, Object Value)	Stocke l'objet *Value* dans la session
Void removeValue(integer Id)	Supprime l'élément *Id* de la session
Int setMaxInactiveInterval(int interval)	Définit l'intervalle de temps maximum entre deux requêtes avant que la session n'expire
Int getMaxInactiveInterval(int interval)	Retourne l'intervalle de temps maximum entre deux requêtes avant que la session n'expire

Tableau 5-1 : Autres fonctions utilisées

Autre stratégies possibles pour la trajectoire du collecteur mobile durant le régime initial.

Durant le régime initial, le collecteur peut suivre une trajectoire selon l'une des stratégies suivantes :

Stratégie 1 : Dans cette stratégie, le collecteur est placé d'une manière aléatoire à un point de la zone de déploiement. Pour balayer d'une manière optimale l'aire de la zone de déploiement, nous proposons d'utiliser la fonction spirale dont la distance entre tournures de la spirale prend comme valeur le rayon de la zone de couverture du nœud. A chaque non réception du message d'écho sur 2 ou 3 périodes normales, le centre de la spirale est déplacé d'un point vers un autre de telle manière à avoir une couverture totale de l'aire de la zone. La fonction qui prend comme variable le centre de la spirale est une fonction représentant la courbe serpent comme présentée sur les figures Figure 5-1 et Figure 5-2.

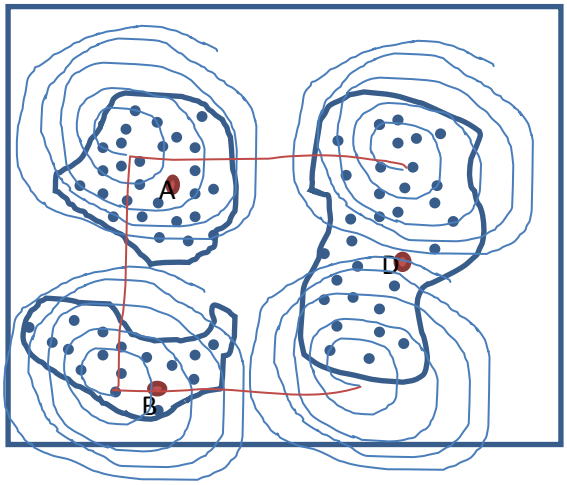


Figure 5