Titre: Title:	Réseaux ad hoc : système d'adressage et méthodes d'accessibilité aux données
Auteur: Author:	Thérence Houngbadji
Date:	2009
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Houngbadji, T. (2009). Réseaux ad hoc : système d'adressage et méthodes d'accessibilité aux données [Ph.D. thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/234/

UNIVERSITÉ DE MONTRÉAL

RÉSEAUX AD HOC: SYSTÈME D'ADRESSAGE ET MÉTHODES D'ACCESSIBILITÉ AUX DONNÉES

THÉRENCE HOUNGBADJI
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE INFORMATIQUE)
DÉCEMBRE 2009

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

RÉSEAUX AD HOC: SYSTÈME D'ADRESSAGE ET MÉTHODES D'ACCESSIBILITÉ AUX DONNÉES

est présentée par : HOUNGBADJI Thérence
en vue de l'obtention du diplôme de : Philosophiæ Doctor
a été dûment acceptée par le jury d'examen constitué de :

Mme. NICOLESCU Gabriela, Doct., présidente
M. PIERRE Samuel, Ph.D., membre et directeur de recherche
M. DAGENAIS Michel, Ph.D., membre
M. BENSLIMANE Abderrahim, Ph.D., membre

À ma famille…

REMERCIEMENTS

Ma reconnaissance va en premier à ma famille dont la confiance, l'appui moral et financier m'ont permis de m'épanouir.

Mes remerciements vont également à mon Directeur de recherche, M. Samuel Pierre pour son soutien indéfectible et ses encouragements tout au long de ce travail.

Enfin, je tiens à témoigner toute ma reconnaissance au Directeur de la Fondation de l'œuvre Saint-Justin en Suisse, M. Nicolas Scherrer, pour tout son appui dans l'entreprise de mes études ici au Canada.

RÉSUMÉ

Au cours de la dernière décennie, un nouveau type de réseaux sans fil a suscité un grand intérêt dans la communauté scientifique: ce sont les réseaux ad hoc. Ils existent sous la variante des réseaux mobiles ad hoc (MANET), et des réseaux de capteurs sans fil (RCSF). Les réseaux mobiles ad hoc sont constitués de nœuds mobiles qui communiquent les uns avec les autres sans l'aide d'une d'infrastructure centralisée. Les nœuds se déplacent librement et sont soumis à des déconnexions fréquentes en raison de l'instabilité des liens. Cela a pour conséquence de diminuer l'accessibilité aux données, et de modifier la façon dont les données sont partagées dans le réseau. Comparable aux réseaux MANET, un RCSF est composé d'un ensemble d'unités de traitements embarquées, appelées capteurs, communiquant via des liens sans fil et dont la fonction principale est la collecte de paramètres relatifs à l'environnement qui les entoure, telles que la température, la pression, ou la présence d'objets. Les RCSF diffèrent des MANET de par le déploiement à grande échelle des nœuds, et trouvent leur application dans diverses activités de la société, tels les processus industriels, les applications militaires de surveillance, l'observation et le suivi d'habitat, etc. Lorsqu'un grand nombre de capteurs sont déployés avec des dispositifs d'actionnement appelés acteurs, le RCSF devient un réseau de capteurs et d'acteurs sans fil (RCASF). Dans une telle situation, les capteurs collaborent pour la détection des phénomènes physiques et rapportent les données afférentes aux acteurs qui les traitent et initient les actions appropriées. De nombreux travaux dans les RCSF supposent l'existence d'adresses et d'infrastructures de routage pour valider leurs propositions. Cependant, l'allocation d'adresses et le routage des données liées aux événements détectés dans ces réseaux restent des défis entiers, en particulier à cause du nombre élevé de capteurs et des ressources limitées dont ils disposent.

Dans cette thèse, nous abordons le problème de l'accessibilité aux données dans les MANET, et les mécanismes d'adressage et de routage dans les RCSF de grande taille.

Ceci implique la mise en œuvre de techniques telles que la mise en cache et la réplication des données, pour prévenir la détérioration de l'accessibilité aux données. Le système d'adressage dans les RCSF inclut le déploiement d'un mécanisme d'allocation d'adresses, et d'une infrastructure de routage. En outre, avec l'avènement des capteurs multimédias, le trafic dans les RCSF devient hétérogène avec des données exigeant d'êtres rapportées dans des délais courts et bornés, tandis que d'autres nécessitent un certain niveau de fiabilité. À cette fin, nous abordons également le problème de fournir une qualité de service (QdS) dans l'infrastructure de routage pour les RCSF. Notre thèse se présente sous la forme de trois articles scientifiques, chacun traitant d'une problématique bien spécifique.

Dans notre premier article intitulé "Joint Data Caching and Replication Scheme in Ad Hoc Networks", et qui a été soumis à la revue *Ad Hoc Networks (Elsevier)*, nous étudions le problème du partage et de l'accessibilité aux données dans les MANET. Un serveur distant détient des données faisant l'objet de requêtes de nœuds mobiles. En utilisant conjointement la mise en cache et la réplication de données, nous avons proposé deux modèles de partage de données. Le premier est un modèle de mise en cache coopérative, homogène qui permet au nœud de collaborer avec ses voisins pour le partage des données, à l'aide d'une infrastructure virtuelle dynamique. Cette dernière a pour rôle de lier entre eux les espaces de stockage de nœuds du même voisinage. Le second modèle combine la mise en cache coopérative et la réplication partielle de données sur des nœuds clés. À cette fin, la réplication partielle des données est formulée comme un problème de programmation linéaire en nombres entiers qui vise à minimiser l'accès aux données et les coûts de réplication de celles-ci. En démontrant que le problème de réplication partielle des données est NP-difficile, nous avons introduit une heuristique efficace pour le résoudre. L'heuristique proposée sélectionne des nœuds clés et y effectuent la réplication partielle des données populaires dans leur cache, en explorant différentes stratégies de réplication. Les analyses de performance et les

résultats de simulation montrent l'efficacité des schémas proposés comparativement à certaines stratégies de mise en cache existantes.

Le deuxième article intitulé “SubCast: A distributed Addressing and Routing System for Large Scale Wireless Sensor and Actor Networks”, et publié dans la revue *Computer Networks (Elsevier)* propose un système distribué d’adressage et de routage basé sur un mécanisme de construction de grappes d’intérêts, et une technique de discrétisation basée sur la théorie fractale des systèmes de fonctions itérés. Afin de minimiser les coûts de livraison des données d’événements détectés, l'architecture proposée s'appuie d'abord sur un réseau surcouche d’acteurs, puis par la suite permet l’allocation d’adresses aux nœuds du réseau. L’information de localisation dans les adresses allouées permet d'établir des chemins de transit des données liées aux événements détectés. Les résultats de simulations confirment que le système proposé garantit de manière efficace l'allocation d’adresses uniques, et effectue un routage fiable des données tout en réduisant les coûts de communication, les délais de bout-en-bout, ainsi que les imperfections dues à l’imprécision de la localisation des nœuds.

Enfin, dans notre troisième article intitulé “ QoSNet: An Integrated QoS Network for Routing Protocols in Large Scale Wireless Sensor Networks”, soumis à la revue *Computer Communication (Elsevier),* nous avons proposé un protocole de routage multi-chemins offrant la QdS dans les RCSF de grande taille. Le routage est basé sur la séparation des nœuds en deux sous-réseaux. Le premier sous-réseau comprend des capteurs spécifiques qui sont occasionnellement impliqués dans la décision de routage tandis que les autres nœuds restants constituent le deuxième sous-réseau et prennent part à toutes les décisions de routage. Le routage avec QdS est formulé comme un problème d'optimisation qui vise à étendre la durée de vie du réseau, en tenant compte des requis de QdS. En appliquant la théorie de la percolation, nous avons conçu un algorithme de routage pour la résolution du problème en considérant alternativement le voisinage du nœud dans les deux sous-réseaux. Les résultats de simulation montrent l'efficacité de

cette approche en termes de délai de bout-en-bout, de fiabilité, et de durée de vie du réseau.

ABSTRACT

During the last decade, a new type of wireless networks has stirred up great interest within the scientific community: there are ad hoc networks. They exist as mobile ad hoc networks (MANET), and wireless sensor (WSN). The mobile ad hoc networks consist of mobile nodes that communicate with each other without using a centralized infrastructure. The nodes move freely and are subject to frequent disconnections due to links instability. This has the effect of reducing data accessibility, and change the way data are shared across the network. Similar MANET networks, a WSN consists of a set of embedded processing units called sensors that communicate with each other via wireless links. Their main function is the collection of parameters relating to the environment around them, such as temperature, pressure, motion, video, etc. WSNs differ from the MANETs due to the large scale deployment of nodes, and are expected to have many applications in various fields, such as industrial processes, military surveillance, observation and monitoring of habitat, etc. When a large number of sensors which are resource-impoverished nodes are deployed with powerful actuation devices, the WSN becomes a Wireless Sensor and Actor Network (WSAN). In such a situation, the collaborative operation of sensors enables the distributed sensing of a physical phenomenon, while actors collect and process sensor data to perform appropriate action. Numerous works in WSN assumes the existence of addresses and routing infrastructure to validate their proposals. However, assigning addresses and delivering detected events remains highly challenging, specifically due to the sheer number of nodes.

In this thesis, we address the problem of data accessibility in MANET, and that of addressing and routing in large scale WSN. This involves techniques such as data caching and replication to prevent the deterioration of data accessibility. The addressing system in WSN includes a distributed address allocation scheme and a routing infrastructure for both actors and sensors. Moreover, with the birth of the multimedia sensors, the traffic may be mixed with time sensitive packets and reliability-demanding

packets. For that purpose, we also address the problem of providing quality of service (QoS) in the routing infrastructure for WSN.

Our thesis runs through three scientific papers, each addressing a specific problem.

In our first paper entitled “Joint Data Caching and Replication Scheme in Ad hoc Networks” and submitted to the *Ad hoc Networks Journal (Elsevier)*, we investigate the problem of data accessibility in MANET where a server stores data items requested by mobile nodes. To this end, two caching models are proposed. The first one allows the mobile nodes to cooperatively share data items with neighbour nodes through a neighbourhood caching table. The role of this table is to link the caching space of the nodes in the same neighborhood. The second scheme combines the cooperative caching scheme and data replication on some key nodes. For that purpose, the partial data replication is formulated as an Integer Linear Programming model that aims to minimize data access and data replication costs. By showing that the partial data replication problem is NP-hard, we introduce an efficient heuristic to solve it. The proposed heuristic selects key nodes and partially replicates popular data items in their caching space by exploring various replication strategies. Performance analysis and simulation results show the effectiveness of the proposed schemes over some existing caching strategies.

The second paper entitled “SubCast: A distributed Addressing and Routing System for Large Scale Wireless Sensor and Actor Networks” published in the *Computer Networks (Elsevier)* proposes a distributed address assignment and routing scheme based on a Topic Clustering System and fractal theory Iterated Function Systems. In order to minimize data delivery costs, the proposed architecture first builds an actor overlay network and allocates addresses to network nodes. Location information in the allocated addresses allows establishing data delivery paths. Simulation results confirm that the proposed system efficiently guarantees the allocation

of unique addresses and performs efficient data delivery while reducing communication costs, delays as well as the impact of imprecise locations.

Finally, our third paper entitled “QoSNet: An Integrated QoS Network for Routing Protocols in Large Scale Wireless Sensor Networks” and submitted in *Computer Communications (Elsevier)* proposes a promising multipath QoS routing protocol based on a separation of the nodes in two sub networks. The first sub-network includes specific nodes that are involved occasionally into the routing decision while the remaining nodes in the second sub-network fully take part into it. The QoS routing is formulated as an optimization problem that aims to extend the network lifetime, under the QoS constraints. Using percolation theory we design a routing algorithm to solve the problem on the respective sub-networks. Simulation results show the efficiency of this novel approach in terms of average end-to-end delay, on-time packet delivery ratio, and network lifetime.

TABLES DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES SIGLES ET ABRÉVIATIONS

CPU	Central Processing Unit
GPS	Global Positioning System
IP	Internet Protocol
MANET	Mobile Ad hoc NETwork
QdS	Qualité de Service
RCSF	Réseau de Capteurs Sans Fil
RCASF	Réseau de Capteurs et d'Acteurs Sans Fil
TCP	Transmission Control Protocol
WiMax	Worldwide Interoperability for Microwave Access
WSN	Wireless Sensor Network
WSAN	Wireless Sensor and Actor Network
WMSN	Wireless Multimedia Sensor Network
P2P	Peer to Peer
TTL	Time to Live
QR	Quasi Replica
QoS	Quality of Service
IMANET	Internet Mobile Ad hoc NETwork
SAF	Static Access Frequency
DAFN	Dynamic Access Frequency and Neighborhood
DCG	Dynamic Connectivity Based Grouping
DHCP	Dynamic Host Configuration Protocol
STS	Storage Set
MMSPEED	Multi-Path and Multi-SPEED Routing Protocol
MCMP	Multi Constrained Multi Paths
GRAB	GRAdient based Broadcast
MU	Mobile Unit
NCT	Neighborhood Caching Table

AODV	Ad hoc On-Demand Distance Vector
DSR	Dynamic Source Routing
DSDV	Destination-Sequenced Distance-Vector Routing
CRR	Cache to Replication Ratio
LRU	Least Recently Used
WRR	Write to Read Ratio
PS	Publish Subscribe
AC	Address Controller
FCR	Fractal Cell Routing
APA	Access Point Actor
RSSI	Radio Signal Strength Indicator
SC	Simple Clustering
SIC	Simple Infrastructure less Clustering
GF	Geographical Forwarding
CT	Centre de Traitement

LISTE DES ANNEXES

CHAPITRE 1

INTRODUCTION

La tendance croissante vers les réseaux de communication sans fil affecte de façon dramatique les dispositifs électroniques ainsi que les habitudes dans toutes les sphères de la vie. Les réseaux sans fil conventionnels sont gérés par des stations de bases; les unités mobiles doivent interagir avec ces stations de bases selon le modèle classique client/serveur. Pour rendre les réseaux sans fil plus autonome, les efforts de recherche se focalisent sur les réseaux sans fil complètement plats. Ceux-ci, dénués de toute infrastructure centralisée permettent aux nœuds de communiquer entre eux malgré la couverture restreinte offerte par leur antenne. Dans un scénario de communication à saut multiples, un paquet généré par une source est relayé par plusieurs nœuds intermédiaires jusqu'à destination. Ces réseaux sont dits ad hoc. Un réseau mobile ad hoc, appelé généralement MANET (Mobile Ad hoc NETwork), consiste en une population relativement dense, d'unités mobiles qui se déplacent dans un territoire quelconque et dont le seul moyen de communication est l'utilisation des interfaces sans fil, sans l'aide d'une infrastructure d'administration centralisée. La topologie du réseau peut changer à tout moment et est imprévisible avec la déconnexion fréquente des unités mobiles. Le caractère aléatoire des MANET pose de sérieux défis lorsque les unités mobiles doivent coopérer pour partager ou accéder à des données et services.

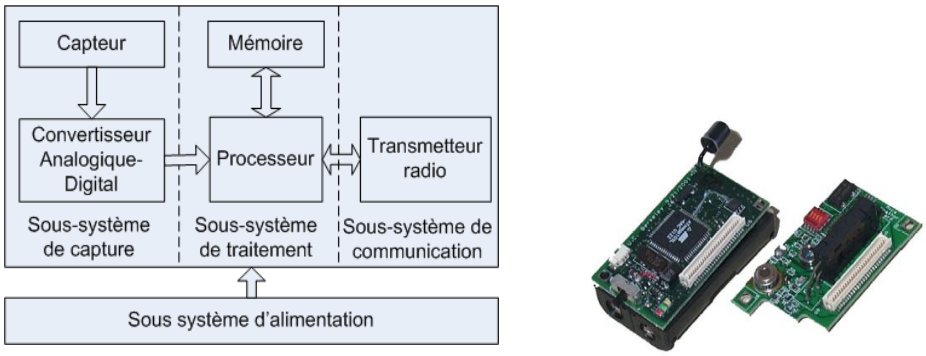
Les défis liés aux réseaux ad hoc sont encore plus importants lorsqu'il s'agit des RCSF à cause du facteur d'échelle, c'est-à-dire que le nombre de nœuds déployés dans les RCSF est beaucoup plus important comparativement aux MANET. En effet, les récentes avancées dans les technologies de miniaturisation et des systèmes embarqués ont permis le développement à faible coût de minuscules systèmes micro-électromécaniques (MEMS) appelés capteurs, capables de détecter, mesurer et

rapporter des données physiques liées à leur environnement. Ces capteurs caractérisés par de faibles ressources (énergie, capacité de calcul, mémoire, etc.) sont munis d'une interface sans fil qui leur permet de collaborer dans l'objectif de réaliser les trois fonctions principales: (a) la capture de données reliées à leur environnement physique (température, pression, vibration, lumière, mouvement, etc.); (b) le traitement des données collectées; (c) la transmission de ces données à un centre de traitement(CT). Une variante de ces capteurs, plus puissante et plus couteuse disposent de fonctionnalités additionnelles telle que la capacité à se mouvoir pour réaliser des tâches : ce sont des acteurs. La cohabitation entre capteurs et acteurs donne lieux à plusieurs autres variantes de RCSF. Les réseaux de capteurs et d'acteurs sans fil (RCASF) sont composés de capteurs qui recueillent des informations du monde physique et les rapportent aux acteurs qui prennent des décisions et effectuent les actions pertinentes. De même, avec l'avènement de capteurs d'imagerie, CMOS caméras, micros, capables de générer des flux multimédias, la dénomination de réseaux de capteurs multimédias sans fil (RCMSF) est née. Tous ces types de RCSF sont destinés à une multitude d'applications dans les domaines militaire, industriel et domestique, et suscitent les intérêts aussi bien de la communauté scientifique que des milieux industriels. Pourtant le déploiement en grand nombre des capteurs pose plusieurs défis liés aux mécanismes d'adressage et aux techniques de routage des données d'événements détectés. Dans ce chapitre, nous commencerons par donner une brève description des MANET et des RCSF, leurs applications et leurs principaux défis. Ensuite, nous présenterons quelques éléments de problématiques liées principalement au mécanisme de partage de données dans les MANET et au système d'adressage et de routage dans les RCSF. Puis, nous détaillerons nos objectifs de recherche. Par la suite, nous exposerons nos principales contributions à la recherche et enfin, nous présenterons le plan de cette thèse.

1.1 Définitions et concepts de base

1.1.1 Anatomie des capteurs

Les RCSF sont composés de dispositifs électroniques de petite taille et de faible coût, intégrant un ensemble de fonctionnalités telles que : une unité de calcul, une mémoire, des sondes et une interface sans fil. Cet ensemble appelé capteur interagit avec l'environnement, de concert avec ses pairs pour détecter et contrôler les phénomènes physiques tels que la température, la pression, le bruit, la machinerie, etc. Un capteur est un système embarqué (de l'électronique et un Built-in OS), tel qu'illustré à la Figure 1.1(a). Il est composé essentiellement de quatre blocs fonctionnels [1] : l'unité centrale composée d'un microprocesseur et d'un microcontrôleur; le bloc d'émission/réception doté d'une interface radio opérant sur une courte distance; le sous-système de capteurs qui relie le nœud au monde physique est constitué d'un ensemble de capteurs; le sous-système d'alimentation en énergie intègre une batterie ainsi qu'un convertisseur de tension continue.

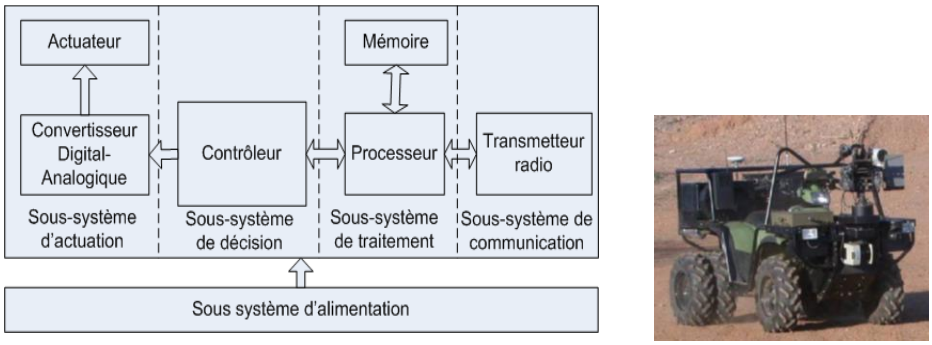


(a) Anatomie d'un capteur (b) Capteurs

Figure 1.1 Vue d'ensemble d'un capteur

L'unité centrale est le centre d'intelligence du capteur. Elle est responsable du contrôle du capteur, de l'exécution des protocoles de communication et des algorithmes de traitement des signaux sur les données captées. Elle supporte plusieurs modes de fonctionnement souvent guidés par le souci de l'économie d'énergie : le

mode actif, le mode libre et le mode de veille. Le sous-système radio assure la communication avec le monde extérieur. Plusieurs facteurs affectent sa consommation en énergie : la technique de modulation utilisée, le débit d'interface, la puissance d'émission, et le cycle d'activité. Il dispose de quatre modes de fonctionnement : émission, réception, libre, veille. Le sous-système de capteurs est responsable de l'observation et de la conversion des phénomènes physiques en signaux électriques; les capteurs peuvent être analogiques ou numériques dépendamment du type de signal qu'ils génèrent sous l'effet du phénomène contrôlé. Ils peuvent mesurer la température, la luminosité, le son, le champ magnétique et capter aussi des images. À son niveau, il existe plusieurs sources de consommation d'énergie : l'échantillonnage du signal reçu, la conversion du phénomène physique en signaux électriques, la mise en forme du signal (filtrage), et la conversion analogique à numérique. L'alimentation est responsable de la fourniture d'énergie aux autres entités, et détermine par ce fait même la durée de vie du capteur. Elle intègre un convertisseur de tension continue responsable de la fourniture en tension constante des autres entités du capteur. La Figure 1.1(b) montre une vue réelle d'un capteur..



(a) Anatomie d'un acteur (b) Acteur

Figure 1.2 Vue d'ensemble d'un acteur

Outre les composants décrits, lorsque le capteur est en plus doté d'une unité de décision qui génère des commandes ou est équipé d'actionneurs, on parle alors d'*acteur*. Ces commandes sont ensuite converties en signaux analogiques par le

convertisseur numérique-analogique (DAC) et sont transformées en actions via l'unité d'actionnement comme indiqué dans la Figure 1.2(a). Un système intégré capteur/acteur est capable à la fois de détection et de mise en œuvre d'actions appropriées, tel l'exemple illustré à la Figure 1.2(b) représentant un robot [2]

1.1.2 Architecture des réseaux mobiles ad hoc

La Figure 1.3 illustre l'architecture d'un MANET formée d'unités mobiles (Laptop, Palm, PocketPC, etc.) autonomes connectées les unes aux autres à l'aide d'une interface sans fil et sans l'aide d'une infrastructure d'administration centralisée. Les unités mobiles sont libres de leur mouvement souvent aléatoires, et s'auto-organisent de façon arbitraire. De ce fait, la topologie du réseau peut changer rapidement et de façon imprévisible. De tels réseaux peuvent opérer avec ou sans connexion au réseau Internet. Les nœuds constituent des routeurs les uns pour les autres, lors de l'acheminement des paquets d'une source à une destination.

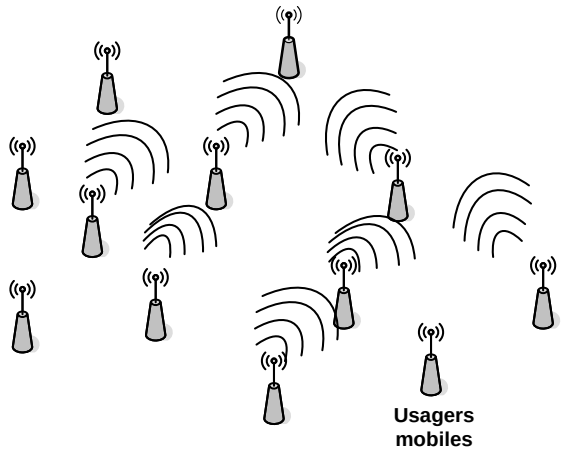


Figure 1.3 Architecture d'un MANET

Chaque nœud est capable de communiquer avec les voisins se trouvant à sa portée radio. Les MANET héritent des caractéristiques générales des réseaux sans fil, mais disposent eux aussi de caractéristiques intrinsèques. Les nœuds disposent d'une portée de transmission limitée par la puissance de transmission, l'interférence ainsi que la topologie du terrain sur lequel ils sont déployés. La mobilité est une

caractéristique majeure des MANET. Le libre mouvement des nœuds engendre une connectivité volatile des liens de communication dans le temps, si bien que la transmission de paquets dépend d'opportunité de connexions. Cette mobilité des nœuds doublée de leur autonomie induit une topologie dynamique du réseau non pas seulement parce que source et destination sont mobiles, mais aussi parce que les nœuds intermédiaires dans une route sont transitoires et sont dotés des mêmes caractéristiques de mobilité. Dans ce contexte, les mécanismes permettant le stockage et le partage des données doivent aussi s'adapter à cette topologie dynamique. Un système de partage de données nécessite la mise en place de mécanismes tels que : la *localisation de données*, la *mise en cache*, la *réplication*, ainsi que la *consistance* des données.

La localisation de données consiste en la détermination du nœud qui dans le réseau détient l'information désirée, puis à utiliser le protocole de routage sous-jacent pour y accéder. Les techniques de mise en cache de données et de réplication de données sont souvent confondues, même si elles présentent certaines différences. En effet, avec la mise en cache, chaque fois qu'un utilisateur requiert une donnée pour la première fois, une copie est servie en guise de réponse à partir du serveur de données. Avant de mettre la donnée à la disposition de l'application requérante, le nœud stocke la donnée localement dans une mémoire cache. Lorsqu'elle est requise à nouveau, la donnée peut être simplement récupérée depuis la mémoire cache locale. La mise en cache des données fréquemment accédées au point d'utilisation peut économiser la bande passante de communication et raccourcir le temps de réponse puisque la communication avec le serveur distant n'est plus nécessaire. La réplication de données elle, consiste à l'entreposage de copie de données sur des nœuds stratégiques, souvent des serveurs, afin de permettre une disponibilité de l'information même en cas de partitionnement du réseau. Le réplica qui constitue la copie de la donnée est en tout moment indépendant du serveur sur lequel il réside, et de ce fait dénommé *réplica flottant*. En particulier le fournisseur ou le serveur de

données place de manière proactive les copies des données sur différents serveurs dans le réseau, prévoyant que suffisamment d'usagers feront usage de ces copies [3]. Un modèle générique de réplication de données fonctionne comme suit: (a) les données à répliquer sont définies, (b) les statistiques sont collectées; (c) le placement de réplicas est décidé en fonction de critères et de contraintes; (d) une méthode de résolution des requêtes des usagers permet d'acheminer ces dernières vers les points de réplication qui peuvent les satisfaire [4]. La cohérence des données elle, consiste à invalider ou non une donnée détenue par un nœud avant son usage. Quand une donnée est mise à jour sur le serveur de données, il est nécessaire de s'assurer que les copies mises en cache par les nœuds soient valides avant qu'elles ne soient de nouveau utilisées.

1.1.3 Architectures des réseaux de capteurs sans fil

La Figure 1.4 illustre la variante des réseaux ad hoc que sont les RCSF. Un RCSF est composé de capteurs généralement déployés en grand nombre et munis de transmetteurs sans fil. Chaque capteur est susceptible de collecter des données (traitées ou pas) de l'environnement et de les acheminer vers un centre de traitement par l'intermédiaire d'une passerelle ("sink" ou "gateway") et ceci en un ou plusieurs sauts. La passerelle peut être fixe ou mobile et transmet ensuite ces données à un gestionnaire de tâches au centre de traitement (CT) via Internet ou par satellite. Le CT pourrait être un simple ordinateur central dont le rôle est l'analyse des données rapportées, et la prise de décisions. Le CT effectue certaines opérations sur les données, comme le filtrage et l'agrégation pour en extraire l'information utile. Les capteurs sont déployés en général par centaines ou milliers, avec une densité de nœuds généralement de l'ordre de 20 nœuds/m^3 [5]. La transmission de l'information entre capteurs peut s'effectuer à l'aide d'un support radio, infrarouge, ou optique.

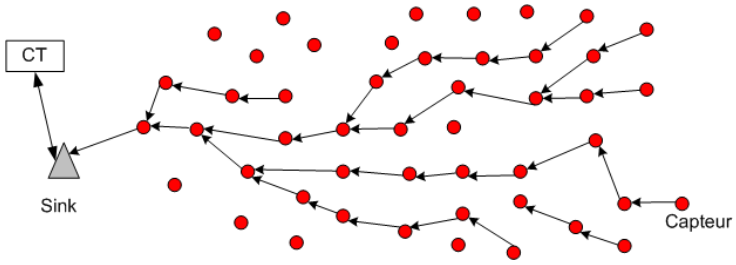


Figure 1.4 Architecture d'un RCSF

Parfois, les capteurs peuvent cohabiter avec des dispositifs plus riches en ressources de toute sorte (puissance de calcul, longue portée de transmission, batterie durable), appelés *acteurs* [2], ce qui donne lieu à l'architecture de la Figure 1.5 illustrant les réseaux de capteurs et d'acteurs sans fil (RCASF).

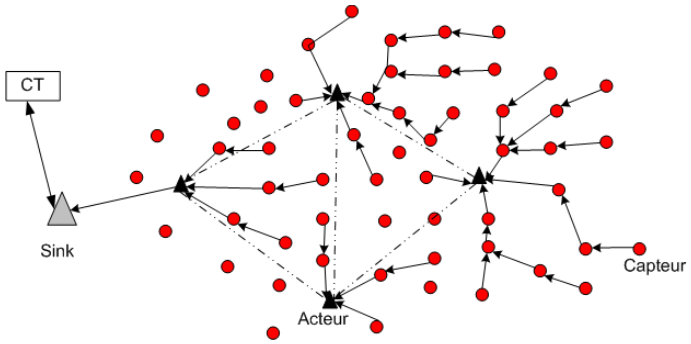
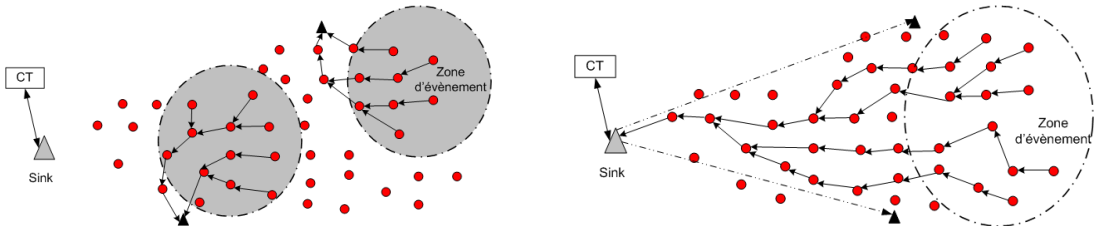


Figure 1.5 Architecture générique d'un RCASF

Des coopérations s'établissent entre capteurs et acteurs mais aussi entre acteurs pour faciliter la gestion d'événements à l'aide d'architectures illustrées à la Figure 1.6. Dans l'architecture automatique (Figure 1.6(a)), les capteurs rapportent les événements détectés aux acteurs qui traitent les données rapportées et initient les actions appropriées. Quant à la Figure 1.6(b), elle illustre l'architecture semi-automatique dans laquelle les capteurs rapportent les événements détectés au Sink qui initient les commandes appropriées aux acteurs.



(a) Architecture automatique (c) Architecture semi-automatique

Figure 1.6 Les différents types d'architecture des RCASF

Généralement, dans les RCSF, la nature des données collectées et acheminées par les capteurs est beaucoup plus importante que l'identité des capteurs: on dit que les RCSF sont *data-centric* [6]. Les données sont requises soit par le Sink, soit par un acteur dans le réseau sur la base d'attributs: on parle alors *d'adressage par attributs.* Ce type d'adressage permet de trouver des routes de plusieurs sources vers une seule destination. Pour cela, les données des événements détectés sont décrites par une liste de paires attribut-valeur [7].

1.1.4 Domaines d'application

Les MANET préliminairement pressentis pour les communications militaires connaissent maintenant un plus grand intérêt dans les domaines commercial et de l'éducation. Dans le domaine militaire, il est possible de doter les soldats de dispositifs sans fil afin qu'ils puissent communiquer les uns avec les autres sans l'aide d'une infrastructure centrale vulnérable. Ils peuvent faire l'objet d'installations domestiques comprenant des Laptop, imprimantes, téléphones mobiles, etc. Les MANET peuvent aussi êtres déployés dans les applications civiles pour permettre la communication entre taxis et polices [8], et aussi dans les opérations de sauvetage d'urgence durant les catastrophes naturelles. Durant des événements spéciaux tels que les jeux Olympiques, la Coupe du Monde de Football, ou dans les aéroports, les MANET peuvent êtres déployés pour l'extension de la couverture radio a plusieurs usagers [9]. De façon générale, les MANET peuvent êtres déployés partout où le recours à une infrastructure centralisée pour la communication est difficile.

Quant aux RCSF, leur usage permet la gestion et l'analyse de phénomènes complexes sur une large région et sur une longue période de temps. Des données telles que la température, l'humidité, la pression, le bruit ainsi que toute sorte d'informations liées à la machinerie peuvent être obtenues et traitées moyennant le déploiement de capteurs sans fil. Les RCSF sont généralement déployés en grand nombre dans une zone géographique où ils vont capter, mesurer et rapporter certains phénomènes physiques. Dans le domaine militaire, ils peuvent être utilisés pour surveiller le déplacement de troupes ennemies dans un espace donné, ou récolter des données dans une zone dangereuse où l'envoi d'êtres humains serait à risque. À titre d'exemple, les RCSF ont été utilisés, sous une forme assez primitive, lors de la guerre froide par les Américains qui ont placé dans l'océan un système appelé SOSUS [10] composé de capteurs acoustiques pour surveiller les sous-marins silencieux russes. Dans le domaine civil, les RCSF peuvent être utilisés dans la surveillance des infrastructures, des installations et des zones critiques. Ils peuvent également être utilisés pour surveiller des habitations et contribuer au confort domestique, en transformant les logements en environnements intelligents dont les paramètres (température, pression, humidité, luminosité, etc.) s'adaptent automatiquement au comportement des individus [11]. Dans le domaine de l'environnement, ils peuvent localiser et suivre le mouvement de certains animaux, détecter des feux de forêt ou encore surveiller l'évolution de la densité moyenne de CO2 dans l'air [1]. CORIE [12] est un exemple réel d'application où des capteurs récoltent des informations sur l'eau de la rivière Columbia (température, salinité, flux et niveau de l'eau, etc.), lesquelles sont utilisées pour gérer les opérations de secours et les interventions sur l'écosystème. Dans le domaine industriel, les capteurs peuvent être placés sur des boites de marchandise dans une usine ou un entrepôt pour suivre en temps réel le mouvement des marchandises, ou encore détecter le dysfonctionnement d'une machine en analysant ses émissions acoustiques et vibratoires. Actuellement, il existe quelques applications commerciales assez primitives des RCSF, par exemple pour la lecture distante des

appareils commerciaux de mesure de la consommation énergétique domestique (les compteurs d'électricité, de gaz, etc.).

1.2 Eléments de la problématique

Même si les MANETs et les RCSF présentent des caractéristiques similaires, le facteur d'échelle rend les défis encore plus considérables dans les RCSF. Dans cette section, nous exposons les défis liés à l'accessibilité aux données dans les MANET, puis ceux relatifs aux mécanismes d'adressage et de routage avec QdS dans les RCSF.

1.2.1 Cas des MANET

Les MANET fournissent une solution attrayante pour la mise en réseau dans les situations où l'infrastructure de réseau n'est pas disponible. Ils peuvent communiquer avec des réseaux extérieurs tels que l'Internet à travers une passerelle [13]. La Figure 1.7 illustre l'architecture d'un MANET composé de nœuds mobiles et d'un serveur distant hébergeant des données ou servant de passerelle vers Internet.

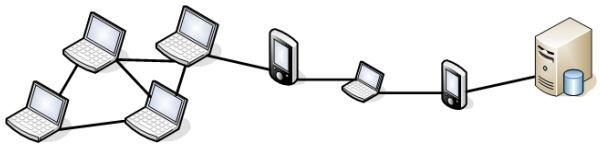


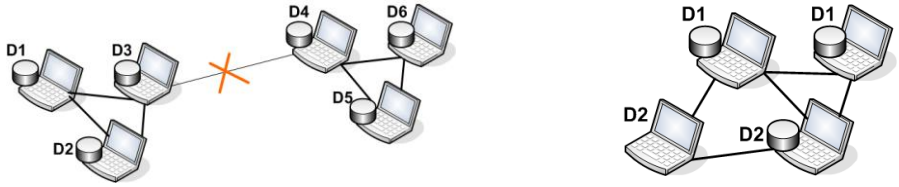
Figure 1.7 Architecture d'un RCSF

Toutefois, les demandes d'accès aux données devront faire face aux problèmes engendrés par les caractéristiques spéciales des MANET, notamment :

Le médium sans fil: les nœuds mobiles communiquent entre eux par l'intermédiaire du médium sans fil ; une seule diffusion peut couvrir tous les voisins a portée radio. Cette nature de la diffusion est un avantage pour le trafic multicast [14], mais pour le trafic unicast, les nœuds entrent en compétition pour avoir accès au médium. En particulier, pour les applications client-serveur, plusieurs sessions simultanées basées sur une communication unicast entre les nœuds et le serveur de données peuvent créer

de la congestion, des pertes de paquets ainsi que des interférences lors de la transmission. Cela a aussi pour conséquence d'augmenter la latence d'accès à l'information.

La topologie dynamique: La topologie du réseau connait des changements fréquents dus à la mobilité des nœuds, la rupture en énergie, et les interférences. Le changement de topologie du réseau provoque aussi des ruptures de routes et induit une surcharge supplémentaire pour l'établissement de nouvelles liaisons. Pour les demandes d'accès aux données, si la route vers le serveur est rompue, la communication client-serveur sera retardée voire supprimée si une nouvelle route ne peut s'établir à temps. Ce retard ou cette déconnexion est plus susceptible de se produire lorsque plusieurs nœuds sont impliqués dans une interaction client-serveur. Comme illustrée à la Figure 1.8(a), la division produite par la rupture du lien empêche les nœuds de part et d'autre d'accéder aux données pourtant diverses qu'ils détiennent.



(a) Problème de rupture de liens lors de l'acces aux données

(b) Problème de redondance du stockage de données

Figure 1.8 Problèmes d'accessibilité aux données dans les MANET

La redondance des données : Si toutes les unités mobiles doivent disposer d'une cache et stocker les mêmes données dans cette dernière, cela relèverait d'un gaspillage d'espace mémoire. Par exemple, on peut imaginer qu'un groupe de travail traitant du même sujet accède pratiquement tout le temps aux mêmes informations et constitue ainsi un scénario typique de gaspillage d'espace, puisque chaque membre du groupe stocke les mêmes informations que ses voisins. La Figure 1.8(b) illustre cette situation dans laquelle des nœuds voisins détiennent les mêmes données D1 et D2, alors qu'il aurait pu y avoir un mécanisme pour ne stocker que des données tout à fait différente dans le même voisinage. Cette situation arrive lorsque chaque nœud

effectue individuellement une politique de remplacement de sa cache qui ne tienne pas compte des données stockées par ses voisins. Il y a donc une nécessité de mettre en place un mécanisme coopératif de remplacement de cache qui tienne compte du voisinage d'un nœud. Les travaux de [15-17] se sont penchés sur les deux précédents problèmes en utilisant la technique de réplication, sans toutefois définir une stratégie de réplication des données telles que spécifiée par les travaux conventionnels dans [18].

La communication multi-saut : Pour les demandes d'accès aux données, lorsque le nombre de sauts dans une route entre un serveur et un client augmente, cela diminue la fiabilité de la route et, en conséquence augmente le temps de réponse. Ainsi la connexion, si elle est établie entre unités mobiles n'est pas toujours fiable, offrant du coup une qualité de service imprévisible.

Des ressources limitées : Un nœud a généralement des ressources limitées en termes d'énergie, de capacité de calcul, et d'espace de stockage. De ce fait, les nœuds subissent des déconnexions fréquentes, c'est-à-dire que les terminaux dont ils disposent ne restent pas indéfiniment connectés, ceci dans le but d'économiser de l'énergie. En conséquence, la résolution des requêtes d'accès aux données doit être effectuée avec le moins de ressources possibles.

La cohérence des données: La faible bande passante ainsi que le motif de déconnexion des unités mobiles rendent difficiles les mises à jour ainsi que la maintenance de la cohérence des données. La mise en cache par les unités mobiles de données résidant sur un serveur distant apparaît comme une technique efficace pour améliorer les performances de l'accessibilité aux données dans les réseaux filaires. Cela réduit considérablement la latence d'accès aux données et diminue le risque d'usage abusif de la bande passante. Cependant, le schéma classique utilisé dans les réseaux filaires où les usagers sont toujours connectés, ne pourrait pas convenir à un environnement aussi dynamique que celui des MANET, à cause de leur déconnexion

fréquente et de leur mobilité incontrôlable. Dans un environnement réel, les données sont généralement mises à jour. Les nœuds du réseau peuvent accéder à des données invalides alors que les originales de celles-ci ont subit une mise à jour. De tels accès aux données invalides consomment inutilement les ressources des nœuds. Pour résoudre ce problème, des mécanismes d'invalidation et de rafraîchissement de données sont nécessaires, ce qui représente tout de même des opérations coûteuses en bande passante. Plusieurs travaux se sont penchés sur le problème [19-21], mais laissent toujours en suspend la question récurrente de la surcharge de signalisation induite par la mise à jour des données.

Toutes ces caractéristiques intrinsèques aux MANET rendent l'accessibilité aux données, complexe. Pour prévenir la détérioration de cette accessibilité aux données ou services, les mécanismes de conservation et de partage de données utilisent en général la réplication ou la mise en cache de données. Même si ces deux techniques sont attrayantes pour le partage et le stockage des données dans les MANET, aucune investigation relative à une stratégie hybride des deux techniques n'a été menée pour approcher le problème de l'accessibilité des unités mobiles aux données.

1.2.2 Cas des RCSF

Le nombre croissant de propositions qui essayent de traiter des problèmes liés aux RCSF reflète l'intérêt de la communauté scientifique pour cette technologie. La résistance au facteur d'échelle, l'adaptabilité, les contraintes d'énergie et la topologie dynamique sont les facteurs de conception qui doivent être considérés lors du déploiement des RCSF [1].

Les RCSF diffèrent justement des MANET par ce facteur d'échelle caractérisé par le déploiement d'une multitude de capteurs souvent dans le but de garantir une redondance dans la détection des événements. Dans ce contexte, l'adressage IP ou *node-centric* qui a été défini pour identifier et localiser un nœud dans les réseaux traditionnels doit être repensé. L'adressage par attribut ou *data-centric* proposé

comme alternative dans les RCSF exploite massivement l'inondation du réseau pour acheminer requêtes et données [5, 7, 22]. L'inondation est en générale très gourmande en énergie, car elle affecte plus de nœuds dans le réseau qu'il n'en faut. Ce problème posé, le défi à relever revient à cibler lors d'une requête d'intérêt, les capteurs susceptibles de garantir des réponses précises dans des délais courts et bornés, sans avoir recours à une inondation massive du réseau. Un mécanisme d'adressage similaire à celui d'IP nécessite la maintenance de tables de routage dans chaque nœud, la connaissance partielle ou totale de la topologie du réseau alors que les capteurs sont déployés par centaines. En outre, la capacité mémoire des capteurs ne peut leur permettre de gérer des mécanismes de fragmentation ou de réassemblage de paquets tels que dans IP.

Avec un réseau aussi dense que celui des capteurs, l'allocation d'adresses IP est complexe parce qu'un mécanisme d'allocation centralisé tel que celui du protocole DHCP deviendrait rapidement une source de congestion et de consommation d'énergie, à cause de la signalisation. La récupération d'adresses constitue un problème à gérer, en considérant que la topologie du réseau est dynamique (essentiellement à cause de la défaillance de nœuds). La mise à jour de tout ou d'une partie du réseau est une opération qui nécessite l'identification précise des nœuds destinés à être affectés. Ce problème non encore exploré dans les RCSF revêt une importance toute particulière car la reconfiguration de nœuds dans les réseaux traditionnels permet de prolonger l'usage du matériel en réduisant les coûts d'investissement dans l'achat de nouveaux matériels. L'usage de la localisation géographique des capteurs comme adresses apparaît comme une solution facile pour répondre au problème d'adressage dans ces réseaux, et par la même occasion permettre un routage géographique des données d'événements détectés. Cependant l'adoption d'un tel mécanisme d'adressage pose les problèmes de coût de consommation énergétique, et d'efficacité de la localisation dans des environnements fermés.

La mise en œuvre d'un schéma d'adressage devrait aider au routage et à la collecte de données dans les RCSF. C'est ce que s'efforce de faire l'adressage par attribut sans pour autant résorber le problème de l'inondation massive du réseau. Dépendamment de la catégorie de routage, le nombre de nœuds impliqués dans l'acheminement des paquets peut être élevé. Dans ce contexte, les ressources énergétiques des nœuds sont fortement sollicitées, diminuant progressivement la durée de vie du réseau. C'est pourquoi tout protocole de routage doit aussi minimiser les dépenses énergétiques des capteurs. Étant prévu que les RCSF devront aussi intégrer des capteurs multimédias (vidéo audio, etc.)[23], les protocoles de routage doivent garantir aussi une certaine QdS pour les applications multimédias et temps réel. Celles-ci produisent généralement un volume considérable de données qui en plus requiert des taux élevés de transmission, et des traitements intermédiaires (compression, décompression, etc.) qui consomment beaucoup d'énergie. En particulier, la transmission d'un flux vidéo en utilisant fréquemment le plus court chemin va absorber l'énergie du nœud sur ce chemin, et ainsi raccourcir la durée de vie du réseau. La plupart des propositions pour le routage dans les RCSF sont basées sur un réseau plat homogène où chaque capteur est identique en termes de capacités physiques. Cependant, les topologies plates ne sont pas toujours les plus adaptées pour traiter la quantité de trafic généré par des applications multimédias [23]. Par conséquent, il est impératif de développer de nouvelles stratégies de routage qui tiennent compte de cette particularité, tout en maximisant la durée de vie du réseau. Outre ces exigences, ces stratégies doivent satisfaire les requis de QdS, ce qui représente un problème ouvert à résoudre dans les RCSF.

1.3 Objectifs de recherche

Cette thèse traite principalement les problématiques liées à l'accessibilité aux données dans les MANET, et celles liées aux systèmes d'adressage et de routage dans les RCSF. L'objectif principal de cette thèse est de concevoir une stratégie de partage et de stockage de données dans les MANET, et une nouvelle génération de schémas, d'algorithmes, de protocoles et de mécanismes pour la conception et l'analyse des

réseaux de capteurs de grande taille, sujets à des contraintes de consommation d'énergie et de qualité de service, en tenant compte de la mobilité éventuelle des nœuds. Plus spécifiquement, dans cette thèse nous visons à:

- Analyser les mécanismes existants d'accessibilité aux données dans les MANET, les systèmes d'adressage existants et les protocoles de routage basés sur la QdS dans les RCSF, afin de déceler certains problèmes encore ouverts, ou dont les solutions proposées sont encore à améliorer;
- Concevoir une stratégie permettant une accessibilité accrue aux données dans les MANET ;
- Concevoir des schémas d'adressage global pour les RCSF de grande taille, dans un souci de réduire la surcharge utilisée pour la maintenance des identificateurs;
- Concevoir des algorithmes et protocoles de routage efficaces en terme de consommation d'énergie, pour supporter la QdS dans les RCSF, destinés aux applications multimédias et temps réel, en tenant compte du caractère dynamique de la topologie de tels réseaux;
- Évaluer la performance des schémas proposés par des méthodes analytiques et par simulation.

1.4 Principales contributions

Les principales contributions de cette thèse touchent une grande partie des défis posés par les réseaux ad hoc. Dans un premier temps nous avons abordé le problème de l'accessibilité aux données dans les MANET par la mise au point d'une stratégie homogène de partage des données fondée sur une mise en cache coopérative de données. Puis, au dessus de celle-ci, nous avons intégré un mécanisme de réplication partielle de données, faisant ainsi de l'ensemble une stratégie hybride de mise en cache et de réplication de données. Ensuite nous avons abordé le problème d'adressage dans les RCSF en incluant des acteurs, et en discrétisant l'aire de déploiement de ce type de réseau par la théorie fractale des systèmes de fonctions

itérés. Outre cette discrétisation, un réseau logique virtuel utilisant le paradigme de dissémination de données Publish/Subscribe(PS) permet l'acheminement des événements aux acteurs adéquats pour la prise de décision. Enfin, pour résoudre le problème du routage garantissant une certaine QdS, nous avons proposé un protocole de routage multi-chemin basé sur la commutation par percolation entre deux sous-réseaux. Ces différentes contributions peuvent être détaillées de la façon suivante :

- **Solution au problème d'accessibilité aux données dans les MANET:** dans les réseaux mobiles ad hoc, l'accessibilité aux données ou service est en général difficile à cause de la mobilité et de l'instabilité des liens entre unités mobiles. Pour prévenir la détérioration de cette accessibilité aux données, les mécanismes de conservation et de partage de données utilisent en général deux techniques exclusives : la réplication de données ou la mise en cache de données. Aucune autre proposition mixant les deux stratégies n'a été explorée pour la résolution du problème de l'accessibilité des unités mobiles aux données. Dans ce contexte, nous avons proposé deux stratégies de partage et de stockage des données dans les MANET. Pour mettre clairement en évidence l'effet d'une stratégie mixte, une mise en cache basée sur la coopération des unités mobiles dans le même voisinage est conçue au préalable. Puis, la stratégie mixte combine la précédente avec un mécanisme de réplication partielle de données sur des nœuds clés. L'évaluation de performance de ces stratégies comparées à d'autres existantes démontre leur efficacité tout en montrant l'avantage de la coopération entre les nœuds pour le partage des données.
- **Solution au problème d'adressage et de routage dans les réseaux de capteurs et d'acteurs de grande taille:** la plupart des travaux dans les RCSF suppose l'existence d'adresses ou d'une infrastructure de routage. Pourtant l'allocation d'adresses ainsi que l'acheminement des évènements détectés posent tout un défi étant donné le nombre important de capteurs dans ce type

de réseau. Dans ce contexte, nous avons proposé SubCast, un nouveau système d'adressage et de routage basé sur la construction de grappes contextuelles et les systèmes de fonctions itérés de la théorie fractale. Dans le but de minimiser le coût de l'acheminement des événements détectés aux différents acteurs, un réseau logique intégrant les acteurs est construit conjointement au mécanisme d'allocation d'adresses. L'information sur la localisation intégrée dans l'adresse allouée au nœud permet l'établissement de chemins de transit des données associées aux événements détectés. Les résultats de simulation confirment l'efficacité du système d'adressage et celle du protocole de routage en termes de réduction des coûts de communication et de réduction des effets dû aux erreurs de localisation.

- **Solution au routage basé sur la qualité de service dans les RCSF de grande taille :** nous avons proposé un protocole de routage multi-chemin en formulant dans un premier temps le routage avec QdS sous la forme d'un problème d'optimisation globale sous contraintes. En argumentant le fait qu'une résolution centralisée du problème dans un réseau aussi grand est couteux en termes de collecte de paramètres et de temps calcul, nous l'avons reformulé sous la forme d'un problème d'optimisation sous contraintes locale au nœud, lui permettant ainsi une résolution exacte basée exclusivement sur un ensemble restreint à son voisinage de un saut. La résolution du problème sous contraintes locales pouvant conduire à l'absence de solution ou la sélection de la même configuration, nous avons opté pour une séparation virtuelle du réseau en deux sous-réseaux, l'un incluant des contrôleurs de cellules et l'autre constitué de simples nœuds. Une technique générique fondée sur l'application de la théorie de la percolation permet de résoudre le problème posé en considérant l'un ou l'autre des sous-réseaux. La configuration obtenue représente un sous-ensemble de nœuds voisins auxquels est retransmis le paquet reçu par le nœud. Les résultats obtenus par simulation montrent que notre approche génèrent de meilleures performances, en termes

de taux d'acheminement des paquets tout en respectant les requis de délai et de fiabilité imposés par la source de l'événement détecté. En outre, elle prolonge la durée de vie du réseau à cause de l'effet de la percolation.

1.5 Plan de la thèse

Le reste de cette thèse est organisé comme suit. Ayant opté pour une thèse par articles, certains chapitres sont donc la transcription d'articles publiés dans, ou soumis à des revues scientifiques. Suite au présent chapitre d'introduction, le Chapitre 2 présente une revue critique et sélective de la littérature sur des problèmes clés des MANET et des RCSF à savoir, l'accessibilité et le partage de données dans les MANETs, ainsi que les mécanismes d'adressage et de routage dans les RCSF. Des différents algorithmes et mécanismes rencontrés dans la littérature, nous avons identifié des problèmes et défis qui ont servi comme base de recherche pour cette thèse. Le Chapitre 3 présente le premier article intitulé « Joint Data Caching and Replication in Mobile Ad Hoc Networks» soumis à la revue *Ad Hoc Networks (Elsevier)*. Dans cet article nous proposons en premier lieu une stratégie coopérative de mise de données en cache, basée sur une infrastructure virtuelle dynamique de liaison entre les espaces de stockage de données de nœuds voisins. Au dessus de cette stratégie est conçu un mécanisme hybride de réplication et de mise en cache de données. Le Chapitre 4 présente notre deuxième article intitulé «SubCast : A Distributed Addressing and Routing System in Large Scale Wireless Sensor and Actor Networks », qui a été publié dans la revue *Computer Networks (Elsevier)*. Dans cet article nous abordons le problème d'adressage dans les RCSF, en appliquant la théorie fractale des systèmes de fonction itérés, ainsi que la construction d'un réseau virtuel d'acteurs logiquement connectés selon leurs intérêts aux événements détectés par les capteurs. Le Chapitre 5 présente notre troisième article intitulé « QoSNET : An integrated QoS routing protocol for Large Scale Wireless Sensor Networks » soumis à la revue *Computer Communication (Elsevier)*. Nous y proposons un protocole de routage multi-chemin permettant de garantir la QdS par application de la

théorie de la percolation. Au Chapitre 6, nous faisons une discussion générale des différents résultats obtenus ainsi qu'une synthèse de nos contributions scientifiques. Le Chapitre 7 conclut la présente thèse en mettant l'accent sur les principales contributions apportées, et en dégageant les principales limitations de nos travaux. Des recommandations pour des travaux futurs y sont également apportées.

CHAPITRE 2

REVUE DE LITTÉRATURE

Plusieurs travaux ont proposé ou traité de modèles de mise en cache ou de réplication de données comme moyen de rendre l'information plus accessible pour les unités mobiles dans les MANET. Les mécanismes proposés dans ces travaux utilisent soit la mise en cache, soit la réplication de données, mais jamais une approche hybride combinant ces deux techniques. Un problème de conception majeur dans la plupart des réseaux est celle de l'adressage des nœuds c'est-à-dire l'attribution d'une identification aux nœuds pour désigner les nœuds ou leur emplacement dans le réseau. Les numéros de téléphone, adresses IP, adresses postales, etc, tous représentent des mécanismes qui aident à identifier, localiser et établir une communication avec des entités d'un réseau. Les caractéristiques uniques des RCSF exigent toutefois de repenser le rôle traditionnel de l'adressage dans ces réseaux. Par exemple le déploiement à grande échelle des RCSF fait que les nœuds se prêtent moins à des adresses soigneusement configurées (comme dans les réseaux IP ou les réseaux de téléphonie). La nature des applications centrées sur les données dans les RCSF conduit à des mécanismes d'adressage qui ont principalement pour rôle d'aider à la collecte et au traitement de données plutôt que la communication pair à pair des réseaux traditionnels. Dans ce chapitre, nous allons passer en revue les différents mécanismes d'accessibilité aux données dans les MANET, puis nous présenterons les systèmes d'adressage et les protocoles de routage basés sur la QdS dans les RCSF.

2.1 La mise en cache coopérative dans les MANET

Le partage et la coordination des données mises en cache permettent d'améliorer les performances de l'accessibilité aux données dans les réseaux en général. Dans [24], les auteurs proposent trois techniques de mise en cache : *CachePath*, *CacheData*, *HybridCache.* Ces techniques utilisent le protocole de routage sous-jacent en mettant en cache les données localement, ou en suivant le chemin d'accès à celles-ci. Dans ces trois approches, un nœud ayant besoin d'une donnée s'adresse au nœud possédant la copie originale de celle-ci. La cohérence des caches est maintenue grâce à un TTL (Time-To-Live). Dans CacheData, les nœuds intermédiaires relayant un message, mémorisent la donnée si elle a été fréquemment demandée auparavant ou s'ils ont assez d'espace mémoire libre, pour pouvoir répondre aux futures requêtes. Dans CachePath, les nœuds intermédiaires mémorisent le chemin de la donnée pour rediriger les requêtes futures vers le nœud le plus proche possédant la donnée. Dans HybridCache, des seuils sont instaurés. Ces seuils concernent certains critères comme la taille de la donnée, la proximité entre le nœud intermédiaire et le nœud possédant une copie et le TTL associé à la donnée. Si la donnée est de petite taille, CacheData peut être adopté car seulement une petite partie du cache va être utilisée pour stocker la donnée. Si le TTL de la donnée est faible, il n'est pas judicieux d'utiliser CachePath puisqu'elle risque d'être invalidée rapidement. Si le nœud intermédiaire et le nœud possédant une copie de la donnée sont très proches, CachePath peut faire économiser un grand nombre de sauts pour accéder à la donnée. Ces approches sont efficaces en termes de temps de requête et de complexité de messages. Leur inconvénient est que si des nœuds proches du nœud demandeur ne se trouvent pas entre celui-ci et le nœud possédant la copie originale de la donnée, ils ne seront pas détectés.

Une technique de mise en cache proactive est proposée dans [9]. Avec cette technique, le nœud requérant diffuse la requête à ses voisins. Si un nœud qui a reçu la requête détient la donnée dans sa cache, il envoie un accusé de réception au nœud demandeur, autrement, il rediffuse la requête à ses voisins. De cette façon, la requête

inonde les nœuds jusqu'à éventuellement atteindre la source de la donnée, ou un nœud ayant une copie de celle-ci en cache. Cette technique est limitée par le coût relativement important de l'inondation, en particulier lorsque le réseau est de grande taille. Les travaux dans [25] exploitent la mise en grappe des nœuds pour la mise en cache de données dans les réseaux ad hoc. Dans chaque grappe est sélectionnée une tête de grappe pour maintenir l'information sur l'état de la cache des membres de la grappe. La cohérence des caches est maintenue grâce à un TTL. Lorsqu'une requête reçue échoue à trouver la donnée en cache, le nœud vérifie l'existence de cette dernière auprès de sa tête de grappe, avant de le rediriger vers le serveur.

Un schéma de mise en cache en agrégat est proposé dans [26] pour améliorer la performance de la communication d'un IMANET, un réseau ad hoc connecté à Internet a travers une passerelle. La technique proposée associe la cache locale de chaque nœud pour former une cache unifiée afin d'améliorer l'accessibilité aux données. Un cocktail de mécanismes de mise en cache appelé COOP est proposé dans [27]. Cette stratégie de mise en cache coopérative utilise une inondation de voisinage à voisinage pour résoudre une requête. Des règles sont utilisées pour minimiser les duplications de la mise en cache entre des nœuds voisins. Cette stratégie présente l'inconvénient d'une surcharge du réseau par inondation, et ne propose aucun mécanisme de remplacement de la cache. Les auteurs dans [28] considèrent la mise en cache comme un problème d'optimisation. Les données sont placées de façon stratégique sur les nœuds, en tenant compte de leur capacité mémoire sous contrainte, et tout ceci dans le but de minimiser le coût d'accès à ces données. Cependant l'algorithme distribué qu'ils ont proposé ne permet pas aux nœuds d'éviter la mise en cache redondante des données.

2.2 La réplication de données dans les MANET

Plusieurs stratégies de réplication de données ont été proposées pour les MANET. Ces schémas supposent des environnements où les nœuds mobiles accèdent à des données entreposées sur des serveurs localisés sur des sites fixes. Trois schémas de

réplication dans [15] se basent sur le fait que le stockage des données dans le même voisinage doit éviter la redondance pour améliorer l'accessibilité aux données. Dans la méthode gloutonne SAF (Static Access Frequency), chaque nœud stocke les données qui l'intéressent sans tenir compte de son voisinage et tout en incluant dans sa stratégie de mise en cache des paramètres tels que la taille et la fréquence de la donnée. La méthode DAFN (Dynamic Access Frequency and Neighborhood) quant à elle prend en compte, en plus de la fréquence d'accès des données, le voisinage, en évitant que la même donnée ne soit mise en cache par des voisins directs. Enfin, la méthode DCG (Dynamic Connectivity Based Grouping) étend l'évitement de redondance aux groupes de nœuds et non plus au simple voisinage. Elle se base sur l'usage de groupes de nœuds mobiles disposant de liens relativement stables. En terme de variété des données stockées, l'algorithme SAF est moins efficace que celui DAFN qui lui-même l'est moins que la méthode DCG. Cependant la surcharge de signalisation est plus importante avec DCG. Une extension de ces méthodes intégrant la mise à jour des données est proposée dans [16, 17]. Comme dans [15], ces techniques adressent les problèmes liés à la mobilité des nœuds puisque la réplication des données est effectuée périodiquement en prenant en compte les changements topologiques ainsi que la fréquence d'accès des nœuds aux données. Bien qu'attrayant, l'usage de ces méthodes consomme une quantité importante de la bande passante à l'échange de messages. Par ailleurs, aucune stratégie élaborée de réplication n'est définie (nombre de copies par données).

Les schémas gloutons dans [29] allouent les données les plus fréquemment accédées jusqu'à ce que la mémoire soit remplie. Ces schémas privilégient la mise en cache des données de petite taille et de ce fait ne garantissent pas une faible latence d'accès. Les travaux dans [30, 31] ont proposé REDMAN, une solution « middleware » décentralisée qui alloue, récupère, et distribue les réplicas des données et des composants de service. Avant la distribution des réplicas, certains nœuds sont sélectionnés comme gestionnaires de réplicas afin de maintenir une table des

ressources partagées. Les réplicas sont distribués dans le réseau par ces nœuds gérants qui se chargent de l'allocation à d'autres nœuds délégués. Chaque délégué sélectionne un voisin au hasard et lui transmet une copie de la donnée. Le mécanisme s'effectue de voisinage à voisinage jusqu'à ce que le degré de réplication spécifié soit atteint. Cette approche ne fournit par contre aucune stratégie de réplication ni de garantie sur la cohérence des données. Une technique nommée Storage Set (STS) est proposée dans [32] pour le partage des données. Cette technique maximise la probabilité d'accéder à la version la plus récente de la donnée tout en minimisant la charge moyenne du réseau par unité de temps. Un client qui a besoin des services d'un serveur envoie sa requête à un serveur sélectionné au hasard dans le STS. Cette approche ne tient pas compte du partitionnement du réseau et propose seulement une stratégie de mise à jour de réplicas.

La technique de réplication prédictive de données décrite dans [33] se base sur la formation de grappes. Un nœud partage avec les membres de sa grappe les données mises en cache dans le but de réduire la redondance des données dans la grappe. La mobilité des nœuds et les déconnexions fréquentes qui causent le partitionnement du réseau sont prédites à l'aide de la position actuelle des nœuds ainsi que leur trajectoire. La sélection d'un nœud candidat pour la réplication est basée sur les ressources du nœud (énergie, capacité mémoire, degré de connexion, etc.). Plusieurs autres techniques de réplication des données dans les MANET sont largement couvertes par la revue faite dans [34].

2.3 Les mécanismes d'adressage dans les RCSF

Bien qu'il y ait quelques travaux sur les mécanismes d'adressage dans les RCSF, la littérature afférente est extrêmement limitée malgré le fait qu'ils aient été décrits dans [2] comme un problème ouvert de la couche réseau des capteurs. L'une des meilleures approches d'allocation d'adresses connues est celle du protocole DHCP (Dynamic Host Configuration Protocol) [35] utilisé dans les réseaux traditionnels. Ce

protocole client-serveur outre d'autres fonctionnalités, délivre des paramètres spécifiques aux clients ainsi qu'une adresse IP pour l'identification globale d'un hôte dans le réseau.

Bien que le caractère dynamique de ce protocole soit adéquat pour les RCSF, il présente le défaut d'échanger beaucoup de messages de signalisation pour allouer les adresses. L'échange multiple de messages de signalisation constitue une source de gaspillage d'énergie. En outre la gestion centralisée du mécanisme d'allocation d'adresse de ce protocole présenterait très vite des limites d'évolutivité dans un RCSF densément peuplés. Les variantes de protocole DHCP proposées dans [36-39] ne sont ni évolutives ni efficaces quant à la consommation d'énergie. Le processus d'allocation nécessite l'échange de messages multiples. En particulier les auteurs dans [39] proposent un processus d'allocation d'adresses en plusieurs étapes. Les étapes consistent à la détection de doublons d'adresses allouées, causant l'échange de message multiples. Il apparait qu'un protocole de type DHCP n'est attrayant que pour les réseaux de capteurs de petite taille et nécessite encore une adaptation pour des réseaux plus larges pour générer moins de signalisation. Les auteurs dans [38] ont proposé un mécanisme d'allocation d'adresses permettant d'allouer temporairement une adresse unique aux nœuds qui ont une information à rapporter et qui en font la demande explicite au Sink. Une technique de partitionnement géographique de l'aire de déploiement du réseau est effectuée de façon à ce que chaque nœud se retrouve dans une grille logique à deux dimensions. Le nœud qui se retrouve dans la grille en prend l'identité qui est réutilisable suivant une distance de réutilisabilité prédéfinie. Cette méthode d'allocation échoue a réduire la charge de trafic d'allocation d'adresse, puisqu'elle est centralisée et présente très rapidement un goulot d'étranglement autour du Sink.

D'autres systèmes d'adressage [40-43] divisent le réseau en couches hiérarchiques et chaque nœud ajoute le suffixe de son prédécesseur dans la hiérarchie pour former sa

propre adresse. En particulier la méthode TreeCast dans [40], adopte cette technique d'hiérarchisation, en offrant un adressage global des nœuds ainsi qu'une infrastructure de routage. Les nœuds sont organisés en une structure physique arborescente, et les adresses sont assignées dépendamment de la position des nœuds dans l'arborescence. Un nœud fils génère son adresse de façon aléatoire suivi d'une approbation du nœud parent. Dans la même lignée, le standard ZigBee propose pour la couche réseau des RCSF [43] une technique d'assignation distribuée ainsi que des infrastructures de routage arborescent comme dans [40], et réactif comme le protocole AODV. Un coordinateur ZigBee détermine le nombre maximum de fils par parent ainsi que la profondeur de l'arbre. L'espace d'adressage est divisé en blocs logiques alloués de façon descendante depuis la racine de l'arbre. Comme précédemment, lorsque la taille du réseau grandit, ces mécanismes d'allocation d'adresses arborescents sont limités puisque l'adresse s'allonge avec la profondeur de l'arbre et la taille du réseau. En particulier dans le cas de ZigBee, l'espace d'adressage de 16 bits est sous-utilisé pour des réseaux de petite taille, et ne supporte pas la mobilité éventuelle des nœuds. D'autres travaux [44-46] offrent des méthodes d'allocation utilisant la formation de grappes. Des têtes de grappe gèrent l'allocation d'adresse, et de ce fait rendent cette technique décentralisée et plus évolutive. L'efficacité de ces techniques ne dépend plus que de leur conception ainsi que du nombre de messages de signalisation nécessaires pour obtenir une adresse.

2.4 Le routage basé sur la QdS dans les RCSF

La disponibilité de matériels peu coûteux comme les mini-cameras ainsi que des microphones a favorisé l'intégration de capteurs multimédia dans les RCSF. La présence de tels capteurs dans les RCSF impose de nouveaux protocoles de routage prenant en compte une architecture densément peuplée de capteurs hétérogènes, plus ou moins performants, et éventuellement mobiles. Les applications multimédias exigent un certain niveau de QdS pour délivrer le contenu multimédia [23]. La littérature relative aux protocoles de routage avec QdS reste encore embryonnaire.

Le protocole SAR [47] est l'un des premiers introduisant la notion de QdS dans les décisions de routage. Le routage dépend de trois facteurs : les ressources énergétiques, les requis de QdS sur chaque chemin ainsi que le niveau de priorité du paquet à acheminer. Pour éviter l'échec du routage, une approche multi chemins est utilisée de même qu'un mécanisme de restauration de routes. Ce protocole présente l'inconvénient d'utiliser beaucoup de messages de signalisation pour maintenir les tables de routage ainsi que les états des routes au niveau de chaque capteur, lorsque la taille du réseau grandit. Le protocole SPEED [48] tient compte de la congestion du réseau pour maintenir une vitesse d'acheminement des paquets en déroutant le trafic vers des zones moins congestionnées. L'estimation du délai détermine l'occurrence de congestion. MMSPEED [49], une extension de SPEED permet quant à elle de supporter aussi la différenciation de service ainsi qu'une garantie probabiliste de la QdS. Pour respecter les requis de délais d'acheminement des paquets, plusieurs vitesses d'acheminement des paquets sont utilisées dépendamment du type de trafic. Comme dans SPEED, la décision de routage est prise localement au niveau du nœud sans l'aide d'aucune information globale du réseau, rendant par le fait même ces techniques évolutives et adaptées à la dynamique du réseau. Cependant, ces techniques présentent l'inconvénient majeur de ne pas tenir compte de la métrique de consommation énergétique, pourtant vitale dans les réseaux de capteurs.

Plusieurs autres stratégies de routage ayant pour objectif le respect du requis de délai de bout en bout ont été conçues [50-53]. Dans [54], les auteurs ont proposé une stratégie qui améliore les délais dans un réseau congestionné, en utilisant une technique de relocalisation du Sink. Ce dernier utilise une connaissance globale de la taille des queues au niveau de chaque nœud pour calculer le délai de bout-en-bout et générer les routes. Cette stratégie présente l'inconvénient d'être centralisée et nécessite la connaissance de l'état des nœuds. Les travaux dans [55] utilisent une heuristique pour déterminer le chemin optimal en terme de consommation d'énergie, et respectant les contraintes de délai. Leur technique utilise un mécanisme de contrôle

topologique ainsi qu'un modèle de contention du délai induit par la couche MAC. Cette solution présente l'inconvénient d'assumer que les nœuds sont équipés de deux interfaces radio : une de faible portée de transmission, et l'autre disposant d'une portée de transmission susceptible d'atteindre le Sink en un saut. Des protocoles de routage multi-chemin pour le maintien de la QdS ont été proposés dans [56, 57]. En particulier, le protocole MCMP [56] utilise comme paramètres de QdS les délais des liens ainsi que leur fiabilité dans la décision de routage prise localement par le nœud. Les paquets sont dupliqués à chaque nœud sélectionné, en résolvant un problème d'optimisation. Cette stratégie testée uniquement pour un RCSF de petite taille s'efforce de minimiser la consommation énergétique en sélectionnant un nombre restreint de chemins pour l'acheminement de paquets. Dans [58], le protocole ReInForM délivre les paquets avec la fiabilité requise en envoyant plusieurs copies de chaque paquet sur plusieurs chemins disjoints en direction du collecteur. Le nombre de chemins est déterminé de façon dynamique, et dépend de la probabilité d'erreur du canal de transmission. Au lieu d'utiliser des chemins disjoints, GRAB [59] emprunte des chemins entrelacés pour atteindre la fiabilité requise. La fiabilité dans les RCSF est d'une importance capitale à cause de la fréquence de défaillance des capteurs. L'usage du routage multi-chemin apparaît comme une avenue de solutions attractives, car l'agrégation de plusieurs chemins de transit de l'information augmente la fiabilité.

2.5 Conclusion

De nombreux algorithmes, mécanismes et protocoles ont été proposés dans la littérature scientifique pour traiter les problématiques d'accessibilité aux données dans les MANET en utilisant soit la mise en cache, soit la réplication de données. Il apparait qu'aucun des travaux présentés n'ait adopté une approche hybride des deux techniques. Cela ouvre la voie à une investigation d'un mécanisme combinant la mise en cache et la réplication de données dans les MANET. La littérature sur les systèmes d'adressage dans les RCSF est encore embryonnaire. De même les protocoles de routage basé sur la qualité de service se focalisent la plupart seulement sur la garantie du délai de bout-en-bout. En outre, ceux qui se sont penchés sur la QdS en utilisant un

routage multi chemin garantissant aussi la fiabilité l'ont fait pour des RCSF de petite taille. Dans un tel contexte la conception d'un système d'adressage ainsi que des protocoles de routage basés sur la QdS reste un problème ouvert.

CHAPITRE 3

JOINT DATA CACHING AND REPLICATION SCHEME IN ADHOC NETWORKS

Therence Houngbadji*, Samuel Pierre, Alejandro Quintero
Department of Computer Engineering, École Polytechnique de Montréal
E-mail: { Therence.Houngbadji, Samuel.Pierre, Alejandro.Quintero}@polymtl.ca
P.O. Box 6079, Centre-Ville Station, Montreal, Quebec, H3C 3A7, Canada
Phone: 514 340-3240 ext. 4685; Fax: 514 340-3240

Abstract

In ad hoc networks data accessibility is lower than that in traditional networks thus, caching strategies remain an open issue. The nodes in MANET move freely and are subject to frequent disconnections due to instability of the links. To prevent the deterioration of data accessibility, data caching and data replication are promising solutions. However, most existing studies on data accessibility in ad hoc networks use only data caching or data replication, but not both. This paper investigates the problem of data accessibility in mobile ad hoc networks where a server stores data items requested by mobile nodes by using jointly caching and replication. To this end, two caching models are proposed. The first one allows the mobile nodes to cooperatively share data items with neighbour nodes through a neighborhood caching table. The second scheme combines the cooperative caching scheme and data replication on some key nodes. For that purpose, the partial data replication is formulated as an Integer Linear Programming model that aims to minimize data access and data replication costs. By showing that the partial data replication problem is NP-hard, we introduce an efficient heuristic to solve it. The proposed heuristic selects the key nodes and partially replicates popular data items in their caching space by exploring various replication strategies. Performance analysis and simulation

*Corresponding author. Phone:+1 514 340 4711 x 2127
E-mails: therence.houngbadji@polymtl.ca (T. Houngbadji), Samuel.Pierre@polymtl.ca (S. Pierre).

results show the effectiveness of the proposed schemes over some existing caching strategies.

Index terms: ad hoc networks, cooperative caching, partial data replication, quasi-replica, simulations.

3.1 Introduction

Recent advances in wireless communication technologies have increased interest in ad hoc networks, in which every host is mobile and acts as an information router for its neighbors. Since a special infrastructure is not required, many applications are expected to be developed for use in ad hoc networks. Most previous research on ad hoc networks focused on developing dynamic routing protocols that can efficiently find routes between two communicating nodes. Although routing is an important objective, another interesting goal consists of providing mobile nodes with reliable access to information. Caching frequently accessed data in MANETs can reduce the delay to access the required information and also improve data accessibility.

There are several ways in which copies of data can be distributed. Usually, two techniques are used to distribute copy of data in a network, namely: *caching* and *replication*. These techniques are often mistaken for one another even if they present some differences. Indeed, with caching, whenever a user requests a data for the first time, the client process will fetch a copy from the data server. Before passing it to the user, the data is stored locally in a cache. When that item is requested again, it can simply be fetched from the cache. In principle, there is no need to contact the server again since the request can be entirely handled locally. Distributed caching of frequently retrieved data at the point of usage can save communication bandwidth and shortens response time. Consequently, the local copy is returned to the requesting user so that communication with remote server is no longer necessary. In order to ensure valid data access, the cache consistency among the data owned by the data source node and the cache copies held by the caching nodes, must also be maintained properly. Depending on the user application, different cache consistency strategies

can be adopted with a replacement policy leading to more or less optimal cache exploitation [60]. Thus, distributed caching can be seen as working from the user's point of view.

In the case of data replication, a data server or provider proactively places copies of data at various servers in the network, anticipating that enough clients will make use of those copies warranting their replication [3]. This kind of caching from the data provider's point of view is known as replication. A generic replication scheme works as follows: (a) the data to be replicated are defined; (b) statistics are collected; (c) replica placement is decided based on some optimization criteria and constraints, and (d) a resolution method is provided to send client requests to the best replicator that can satisfy them[4]. While caching data is useful for the device itself, such mechanism must allow its peer to cooperatively share the cached data, even if peer nodes are located multi-hops away. Existing studies on data accessibility issues in ad hoc networks use only data caching or data replication, but not both. This raises the opportunity to investigate a new data sharing model that combines data caching and replication to improve the performances of data accessibility.

The contributions of this paper consist of the following two building blocks. First, we design a cooperative neighborhood caching strategy named CoonCache. To this end, a cache resolution mechanism coupled with a replacement policy allows the link between the caching spaces of nodes in the same vicinity. In particular, for the cache replacement policy, we develop an analytical model of a utility function that takes into account parameters such as the number of hops to the data source, the data size and its access frequency. Second, we combine the designed caching model, CoonCache with partial replication strategies to form the qCache data sharing system. Indeed, to conceive qCache, key nodes referred to as *quasi-replica* (QR) are selected by the server to host the most popular data items while the remaining *mobile units* (MUs) use the CoonCache scheme. The partial data replication on QRs is formulates as an Integer Linear Programming problem and a proof of its NP-hardness is provided. A dynamic heuristic is outlined to solve the problem. Mathematical

analysis and simulation results prove the effectiveness of qCache over Cooncache and existing caching strategies.

The remainder of the paper is organized as follows: Section 3.2 presents an overview of the related works; Section 3.3 describes the proposed schemes and outlines the design methodologies; Section 3.4 elaborates on the performance evaluation and the main results obtained through extensive simulations; Section 3.5 concludes the paper.

3.2 Related Work

Many papers have proposed or dealt with models that involve caching or replication as a way to make information more accessible to mobile devices. These models belong to either one of two categories: cooperative caching and replication.

3.2.1 Cooperative Caching

Cooperative caching nodes sharing and coordinating cached data make it possible to improve Web performance in wired networks. In [24], the authors propose three caching techniques: *CachePath*, *CacheData*, *HybridCache*. These techniques use the routing protocol by caching data locally or caching the data path in order to save space. In *CachePath*, a node need not record the path information of all passing data; instead, it only records the data path when it is closer to the caching node than the data source. In *CacheData*, the router nodes cache data rather than the path, in cases where data is frequently accessed. To avoid wasting caching space, a node does not cache data if all data requests originate from the same node. In *HybridCache*, a node caches data or its path based on certain parameters that include data size and data time-to-live (TTL).

Another caching approach, called *proactive cooperative caching*, is proposed in [9]. With this approach, the requesting node actively searches data on other nodes. The requesting node broadcasts a request to its neighboring nodes. If the node that receives the request holds the data in its local cache, it sends an acknowledgement (ACK) to the requesting node; otherwise, it forwards the request to its neighbors. This

way, a request is flooded to other nodes and eventually acknowledged by the data source or a node with the cached copy. This approach is limited by its high flooding cost in a densely populated network. The work in [25] exploits clustering for data caching in ad hoc networks. In each clustering area, a super node is selected to maintain the cluster cache state information for cluster members. Each item in the list of cached items has a specific TTL to indicate data freshness. For a local cache miss, each client checks the data with its cluster head before forwarding the requests to the server. An aggregate caching scheme is proposed in [26] to improve the communication performance of an IMANET, a ubiquitous communication infrastructure consisting of both the wired Internet and a wireless MANET. The proposed approach combines the local cache of each user, to form a unified cache that can alleviate the limited data accessibility and longer access latency problems. A cocktail approach named COOP is proposed in [27]. This cooperative caching strategy uses a flooding-based data discovery for the cache resolution process. Inter-category and intra-category rules are used to minimize caching duplications between neighboring nodes. This scheme presents the disadvantage of the high flooding overhead and does not consider parameters such as data size and consistency for the cache replacement. Tang et al. [28] consider the cache placement problem of minimizing total data access cost in ad hoc networks with multiple data items and nodes with limited memory capacity. However the distributed algorithm they have proposed does not allow nodes to avoid caching redundant data.

3.2.2 Replication

Several data replication schemes have been proposed for wireless networks. These schemes assume environments where mobile nodes access databases located at fixed network sites, and create data replicas on the mobile nodes. Three data replication schemes have been proposed in [15]. These schemes are based on the premise that replicating the same data near neighboring nodes should be avoided in order to improve data accessibility. They are the following: Static Access Frequency (SAF), Dynamic Access Frequency and Neighborhood (DAFN) and Dynamic Connectivity

Based Grouping (DCG). In the SAF method, only the access frequency of each data item is taken into account while in the DAFN method, the access frequency of all data and the neighborhood among mobile hosts are taken into account. In the DCG method, stable groups of mobile hosts are created and replicas are shared within each group. Extension of these methods with data update has been provided in [16, 17]. As in [15], these techniques address the issues caused by host mobility, as they make the decision to replicate data items every time during the relocation period based on the current topology of the network and the access frequencies of the mobile hosts. Using these previous methods will consume a considerable amount of bandwidth due to the huge number of exchanged messages. The greedy schemes in [29] allocate the most frequently accessed data items until the memory is filled up. These schemes make the assumption that smaller data require less memory size and their replication can save memory space for other data. However, it does not guarantee lower access delays since all data items are treated equally whatever is their size.

The works in [30, 31] have proposed REDMAN, a decentralized middleware solution that manages, retrieves and disseminates replicas of data and service components. Prior to replica distribution, some replicas managers are elected to maintain a shared resource table. The replicas are distributed in the network by managers which appoint the owner of each data item as a delegate of the respective data items. Each delegate selects the neighbor randomly and forwards the data item until the degree of replication is reached. This approach does not provide a mechanism to determine the number of replicas required for each data item nor data consistency guarantee. A storage set (STS) is proposed by Luo et al. [32]to share data in replicated environment. Subset of nodes is predetermined based on the power level of each server. This technique maximizes the probability of fetching the most recent value of the data item while minimizing the average network load per unit time. A client that needs services from a server sends its request to a random server in the STS. This approach is not aware of network partitioning and proposes only a replica update strategy. The predictive data replication technique described in [33] uses the

group-based data accessibility scheme where a set of mobile nodes forms a separate group. Nodes within a group collectively host a set of data items that are available for the group members. This reduces data redundancy within that group. The mobility of the nodes and frequent disconnections that cause network partitioning are predicted based on the current position of nodes and their patterns of movement. The selection of a candidate node for replication is based on the nodes' capabilities. Numerous other techniques that address data replication in MANET are largely covered in the survey made in [34].

3.3 Proposed Systems

In this section, we provide a step by step realization of the components that made the CoonCache and the qCache data sharing system in MANETs. We first present the system model, followed by the design overview of the CoonCache scheme. Over the CoonCache data caching system, we build qCache, a combination of the caching system and partial data replication. Our goal is not to propose a new replication scheme but rather to provide evidence that such scheme when coupled with caching performs better. Therefore, this work is complimentary with the above described efforts.

3.3.1 System Model

The considered MANET architecture consists of MUs that have access to a data server. The server is connected to other networks such as Internet. MUs are connected with each other and form a self-configuring network with arbitrary topology. The server contains a set of data items and can answer all the requests initiated by the mobile nodes. MUs can move freely and generate queries at anytime and from anywhere. Some of them can communicate directly with the server while others located far enough, interact with the server through multi-hop communication. The server always keeps up to date the data requested by mobile nodes. Before using the services provided by the server, MUs authenticate themselves to the server and, over time, they may be asked for re-authentication. The Time-To-Live data consistency

scheme is adopted. Data consistency ensures that clients only access valid states of the data. There has been an extensive amount of literature work on cache consistency mechanisms in MANETs [3, 20, 21, 60-62]. Two widely used cache consistency models are the weak consistency and the strong consistency. In the weak consistency model, a stale data might be returned to the client while in the strong consistency, after an update completes, no stale copy of the modified data will be returned to the client [20]. To maintain strong cache consistency, the broadcast based schemes were used in [21, 61]. However in these works, the server is able to join mobile units in one broadcast shot or a network-wide flooding is used in invalidation report. In MANET, the server may not be able to join nodes using one hop broadcast, as the nodes move freely, possibly out of the server transmission range. Broadcasting an invalidation report will use network-wide flooding each time the server has to report changes in its database. The flooding cost can become tremendous when the update rate increases at the server side. Due to bandwidth and power constraints in ad hoc networks, it is too expensive to maintain strong consistency. The weak consistency model based on Time-To-Live (TTL) is more appealing [24], [28]. The basic principle of TTL is that each data item is associated with a TTL value and the data is considered valid by the client as long as the TTL does not expire. This strategy is attractive in MANETs because the MUs are not relying on the servers' help to validate their cached items. Unlike the invalidation strategy where the clients have to be connected and tuned-in to receive the invalidation reports from the server periodically or asynchronously, the TTL method allows clients to be free to move and disconnected.

3.3.2 Applications

The data replication anycast service should be useful for several applications. An application scenario is that of MUs organized as an ad hoc network which accesses a hot spot like a data server. This kind of scenario occurs frequently during special events such as Olympic Games, World Cup Soccer or in airports when users access information through their laptops or handheld devices. Since the demand from users to access the Internet via a fixed infrastructure is very high, the use of replication

beside data caching can help to reduce the server load and the delay to access the requested item. As pointed out by [9], a visitor in a downtown, museum or shopping mall may need to access various type of information (e.g., exhibition info, tour info including maps, restaurants of choice, hotels, theaters and so on). A local service provider usually provides an electronic guide such as an info-station that contains the relevant information. Sometimes, the visitor may lose connection to the info-station because of mobility; by using caching and replication the visitor can still access or share the information.

3.3.3 Cooperative Neighborhood Caching

This section presents a cooperative neighborhood caching strategy *CoonCache*, based on the system model illustrated in Figure 3.1. When dealing with data caching, three steps are required: Cache resolution, Cache replacement policy and Cache consistency. As previously mentioned, the Time-To-Live data consistency model is considered. We then focus on the cache resolution and the cache replacement policy.

3.3.3.1 Cache Resolution

The objective of the cache resolution is to resolve the MU's request with minimum delay and resources utilization. The caching strategy proposed in this section exploits the hop-by-hop cache resolution and the node neighborhood to deliver answers to the MU's queries. The following operations are executed when a MU requests a data item: (1) it first looks up the data in local cache. In the case of a hit, the data is served to the requesting application and there is no extra communication cost; (2) when the request fails to find the data in local cache, the node looks in its Neighborhood Caching Table (NCT) to determine if a one-hop neighbor has the requested data. If it is the case, the request is forwarded to this neighbor. Note that the NCT stores the meta data composed of neighbor node addresses, the indexes of the items they cached, their associate TTL and timestamps; (3) if the request fails to get an answer from the NCT, it is forwarded hop-by-hop to the server following the path provided by the underlying routing protocol. The intermediate nodes in the path toward the

server execute the above described operation as well. If one of the intermediate nodes in the path succeeds to get an answer, it stops forwarding and returns the data to the requester. Figure 1 illustrates the forwarding chain followed by the cache resolution process when the originating node N1 requests a data item. The four data retrieval scenarios are:

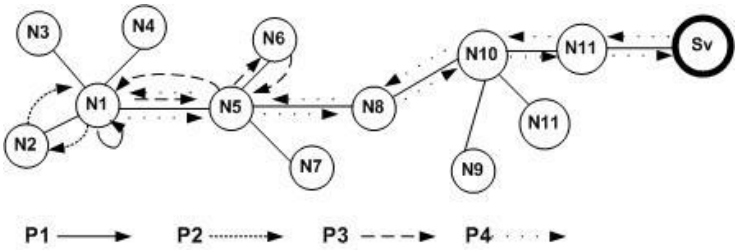


Figure 3.1 CoonCache resolution process.

P1: The node N1 finds the data item valid in local cache and serves it to the application.

P2: The node N1 requests a data item but does not find it in local cache and realizes that the item is valid in its neighborhood when looked up in the NCT. It then forwards the request to the caching neighbor node N2 which answers by sending the requested item.

P3: The request from node N1 fails to get an answer from either the local cache or from the NCT. In such case, the request is forward to the next hop node N5 in the path towards the server. The latter executes P1 and P2. When the N5 looks up in the NCT, it finds a valid copy of the requested data available at N6 and then forwards the request to this node. N6 sends the data back to N5 which forwards it to N1.

P4: The requested data is not found anywhere around node N1 or in the intermediate nodes in the path N1-N5-N8-N10-N11-Sv towards the server. In this case, the request is forwarded to the server which sends the data back**.** Along the reverse path, intermediate nodes refresh the data if it exists in local cache.

3.3.3.2 The Neighborhood Caching Table

As previously described, each node in the network maintains a neighborhood caching table (NCT) that records the meta-data of the items stored by the nodes in its vicinity. Indeed, each time a node caches a new data item, it broadcasts the meta-data of the items stored in cache that are still valid. The meta-data of an item is a description of its properties described by the tuple $< d_k, TTL_k, TS_k >$ that represents respectively the item index, its life time and the time it was cached. Each entry in the NCT is of the form $< N_i, < d_k, TTL_k, TS_k > \cdots >$, where N_i represents the node's address and the following tuples, the meta data of the valid items stored by N_i. The rationale of using the NCT is motivated by the following reasons:

a. Suppose a node N_i requests an item d_k by flooding its neighborhood. Such a blind request may result in multiple answers and raises the communication cost especially for large data items. Instead, a proactive approach that consists of advertising neighbors about newly cached items, allows unicast request to the appropriate caching neighbor node, with low cost communication.
b. Using the NCT would incite neighbors to cache different items and allow a link between one-hop neighbor caches. Indeed, prior to caching the incoming item, the node looks up in the NCT to determine if the incoming item is already stored by a neighbor. If the item is not yet cached around, it is stored in the cache according to the replacement policy. The above process allows data redundancy avoidance and favors the storage of a great variety of items in the same neighborhood.

An issue that affects the performance of a caching scheme is the cache replacement policy. To cache incoming item when the cache space is full, a node must estimate the benefit of caching it versus evicting some others from a full caching space. Dealing with such issue is the objective of the next section.

3.3.3.3 Cache Replacement Policy

A Utility function for cache replacement policy is developed in this section, based on parameters that are easier to get. Recall that cache replacement policy is needed,

when the MU wants to cache a data item but the cache space is full. In such a situation, suitable subset of items must be victimized and evicted from the cache. For that purpose, the following analysis provides an insight of how the utility function is built. The notations used are:

- $\lambda_k = 1/TTL_k$: the update rate of a data item d_k; λ_k^a the access request arrival rate of data item d_k; λ_a the application request arrival rate;
- s_k is the size of data item d_k, and s_{req} that of a request;
- $H(u, v)$ the number of hops from the requester u to the node v which hold d_k ; B_w is the wireless bandwidth;
- $p_k = \lambda_k^a/\lambda_a$: the average access probability to d_k; β_k the probability that when the MU queries the data item d_k, it is still valid (i.e., the data item has not been modified since the last queries); $c(d_k)$ the communication cost to retrieve d_k; C the capacity of the MU cache space.

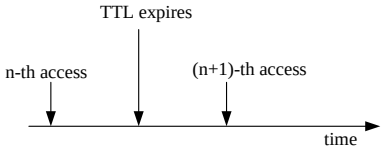


Figure 3.2 The timing diagram

Figure 3.2 depicts the kind of events that can occur during the application request. Indeed, between two requests, the TTL of the data item d_k can expire so that the arriving request must be forwarded to another host that holding a valid copy of d_k. The probability β_k that the data item d_k is still valid when the MU requests it is given as follows:

$$\beta_k = \int_0^\infty e^{-\lambda_k t}\, \lambda_k^a e^{\lambda_k^a t} dt = \frac{\lambda_k^a}{\lambda_k^a + \lambda_k} \tag{3.1}$$

The communication cost $c(d_k)$ of the MU requesting a data item d_k is given by:

$$c(d_k) = p_k \beta_k H(u, v) \frac{s_k + s_{req}}{B_w} \tag{3.2}$$

The greater the number of hops, the higher the communication cost of the requested data item. Thus, caching data items which are further away saves bandwidth and reduces latency for subsequent requests. Ideally, the request size should remain constant as well as the bandwidth. Thus, the utility function of caching a valid data item d_k can be approximated as $B_w c(d_k)$ with a closed form written as follows:

$$U(d_k) = p_k \frac{\lambda_k^a}{\lambda_k^a + \lambda_k} s_k H(u, v) \quad (3.3)$$

In a practical point of view, the number of hops $H(u, v)$ can be provided by the underlying routing protocol. It is easy to get the size of the data item in the cache as well as the data access rate λ_k^a which can be estimated by employing the sliding window method of last access times [63]. The average update rate of the data item λ_k can be estimated by the server using the same approach. If the server maintains this average, it can be sent to the MU during its query to the server. Based on the above analysis, the idea is to maximize the total utility value for the data items kept in the cache. Let z_k be a 0-1 variable that indicates whether data item d_k in the cache should be victimized or not. The replacement policy, given a cache of size *C* so that $s_k \ll C$, should always retain a set of data items that maximize the utility function as follows:

$$max \sum_k U(d_k) z_k \qquad s.t. \quad \sum_k s_k \leq C \quad (3.4)$$

This optimization problem can be mapped to a 0-1 knapsack problem which is known to be NP-hard. For this kind of problem, a simple greedy heuristic ***Greedy-U*** can be used as follows:

Greedy-U:

i. Evict the cached data item d_k with the minimum utility function value until the free cache space is sufficient to store the incoming data;
ii. Broadcast the new cached item's meta-data $< d_k, TTL_k, TS_k >$.

The cooperative neighborhood caching mechanism CoonCache described in this section provides a novel approach to share data in MANET. However, when some data items are more frequently accessed than others, and the cache replacement policy treats them as it does for the less popular items, this can affect the latency to retrieve

the requested item. This situation raises the opportunity to investigate in the next section a novel approach named qCache that combines the CoonCache scheme with partial replication. The goal is to discriminate lower accessed data items from the popular ones and to allow some MUs not to use a replacement policy, but instead, a replication strategy triggered by the server.

3.3.4 Partial Data Replication in MANET

In the previous sections, we have designed CoonCache, a cooperative neighborhood caching scheme that aims to improve data accessibility. However, with this caching strategy, popular data items may not be always available either in local cache or in the node neighborhood. Indeed, intermittent node disconnection that occurs frequently in MANET causes a node to find itself sometimes with only few neighbors and then reducing its accessibility to popular items. To deal with this issue, we explore in this section the achievable performance gain when deploying cooperative caching in conjunction with replication. The basic idea of using data caching in conjunction with data replication in MANETs consists of indicating key MUs to replicate partitions of the server data items. These key MUs are called *Quasi-Replica* (QR). Then, some MUs access data stored on the QRs, in their vicinity. A specific aspect of this idea is that the data items hosted by the server are subdivided into two sets: (a) a *hot set* composed of the most popular items replicated by the server on the QRs, near MUs; (b) a *cold set* composed of the remaining items. The cache replacement policy is used by the caching nodes conversely to the QRs nodes which do not apply any replacement policy. Instead, they only store and maintain consistently the popular items allotted by the server. The QRs are selected by the server on the basis of the registered nodes. Indeed, prior to requesting any item from the server, a joining MU registered itself to the server for security or billing purposes for example. A full data replication on every node in the network will be costly in terms of maintaining data consistency. Having multiple copies of items in the network introduces a consistency problem since replicated nodes have to update periodically and constantly the replicas they hold. This can become costly in terms of signaling overheads if multiple nodes

are selected as QR. Consequently, a partial replication that consists of replicating popular items on a few hosts appears to be an appealing solution. However, data replication requires the knowledge of the number of nodes to be replicated and a replication strategy that specifies the number of copies of each item to be replicated.

3.3.4.1 Optimal Number of QRs

Fewer numbers of QRs may create hot spots if popular items are only accessed on these QRs. Furthermore, a given QR can be overloaded compared to others, when many MUs in its vicinity requested the associated popular items. Such a situation may drain the power of that QR rapidly, raising the load distribution and power consumption issues. To deal with these issues, we determine in this section the optimal number of QRs, given the population of MUs. Let n_{MU} denote the number of MUs that carry out their queries and n_{QR} the number of QRs. Let C defines the data sharing space capacity of nodes. With a perfect load distribution on QRs, on average n_{MU}/n_{QR} MUs can access an item on a given QR including itself. Then the total storage capacity C_{TOT} shared by nodes in the network can be expressed as follows:

$$C_{TOT} = n_{QR}C + \frac{n_{MU}}{n_{QR}}C \tag{3.5}$$

By setting the derivative of this function equal to zero, that is $\frac{\partial C_{TOT}}{\partial n_{QR}} = 0$, the optimal number of QRs is obtained as follows:

$$n_{QR}^{opt} = \sqrt{n_{MU}} \tag{3.6}$$

With the optimal number of QRs determined, it is now possible to address the replication strategy. Given a query frequency distribution, the replication strategy must specify for each item, the number of copies that have to be stored on the QRs.

3.3.4.2 Replication Strategy

This section addresses two issues: (a) the way copies of items are allotted to the QRs; (b) what items in the server database, will be replicated, and how the replication is

done between the server and the the QRs. To improve system performance, it is useful for any mobile node to minimize the number of hosts that have to be probed before a query is resolved. Replicating data on several hosts is a way to solve such issue. In this perspective, the conventional work in [18, 64] has designed three allocation strategies for unstructured peer-to-peer networks, namely: uniform, proportional, and square-root. These strategies give an idea of the number of copies that have to be replicated for a given data item. For the sake of clarity, let us define what an allocation strategy is.

Definition 1: An allocation strategy is a mapping of an item to the number of copies of that item. Let $m = |D|$ denote the number of distinct data items of the set D owned by the server, and $\boldsymbol{\lambda} = [\lambda_1^a, \lambda_2^a, \cdots, \lambda_m^a]$ the normalized vector of query rates such that $\lambda_1^a \geq \lambda_2^a \geq \cdots \geq \lambda_m^a$ with $\sum_k \lambda_k^a = 1$. Let γ_k denote the percentage of the total system storage allotted to item d_k. The allocation is represented by the vector $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \cdots, \gamma_m]$. The allocation strategy is a mapping from query rates distribution $\boldsymbol{\lambda}$ to the allocation vector $\boldsymbol{\gamma}$ **.**

Knowing $\boldsymbol{\gamma}$ and the overall system storage space**,** gives the number of copies to be allotted to each item provided by the vector $\mathbf{n} = [n_1, n_2, \cdots, n_m]$ where n_k represents the number of copies of the item d_k. Our objective is to use the strength of both data caching and replication to improve data accessibility. When caching data, a replacement policy is needed. However, when data is replicated, the copies remain static on the replicated nodes without any replacement policy contrary to data caching. Then, careful selection of data that will remain static after replication is needed. Indeed, if a group of users frequently read a data item say d_k, then it is beneficial to allocate a copy of d_k near the users. This way, the reads access the copy locally and reduce the communication overhead. In contrast, if users read d_k infrequently, the access should be on-demand. There are thus two sets of data items: the *hot data set* D_h composed of popular data frequently accessed, and the *cold data set* D_c which consist of lower accessed data such that $D = D_c \cup D_h$. Let us now describe how the server determines the two data sets. For this purpose, we assume

that queries considered as read operations on data item d_k, are issued from MU according to the Poisson distribution with parameter λ_k^a. Updates considered as writes on a data item d_k occur at the server, following an exponential distribution with rate λ_k. The probability that the next access to d_k on the server is a read, assuming that the last access was a read is equal to β_k. Items on the server are then sorted in descending order of the value of their β_k. Hot data items are located at the top of the list. The hot set is then composed of the k-th first items of the server in terms of their β_k. The hot set is then defined as follows:

$$D_h = \left\{ k \in D, \sum_{k=1}^{m} n_k s_k \leq n_{QR} C \right\} \quad \text{and } D_c = D - D_h \tag{3.7}$$

Note that to compute β_k, the server also uses the sliding window method of last access times [63]. With the knowledge of the number of QRs, and the hot set D_h of items to be replicated, the next step is to determine how QRs are selected among MUs, and where to place these hot items. To solve these issues, we formulate the replica allocation as an Integer Linear Programming problem named Partial Replica Placement Problem (PRPP) and provide a heuristic to solve it.

3.3.4.3 Partial Replica Placement Problem

PRPP consists of assigning MUs to QRs and performs the optimal replica placement on these QRs so that the access cost to hot data and their allocation cost to QRs is minimized. Let Q denote the QR set. To model this problem, we first define the following variables:

- $x_{ij} = \begin{cases} 1, & \text{if the MU i accesses data on the QR installed on MU j} \\ 0, & \text{else} \end{cases}$
- $y_{jk} = \begin{cases} 1, & \text{if the data item } d_k \text{ is allocated to the QR installed on MU j} \\ 0, & \text{else} \end{cases}$
- M the set of MUs; c_{ij} the cost of MU i accessing data on the QR j (the distance between the MU and the QR); h_{jk} the allocation cost of item d_k to QR j (the distance between the server and the QR); r_{jk} the demand of item d_k from QR j;

- $C_Q = C \cdot n_{QR}^{opt}$ the total storage space of the selected QRs.

Formally, PRPP is a combination of two sub-problems defined as follows:

PRPP 1: Given the MU set M, find the QR set $Q \subset M$ with $n_{QR}^{opt} = |Q|$ that minimizes the data access cost.

PRPP 2: Given the hot set D_h and the QR set Q, find n_k QRs from Q to put copy of item d_k, so that the items allocation cost is minimized.

Combining PRPP1 and PRPP2 leads to:

PRPP:

$$min \sum_{i \in M} \sum_{j \in Q} c_{ij} x_{ij} + \sum_{k \in D_h} \sum_{j \in Q} r_{jk} h_{jk} y_{jk} \qquad (3.8)$$

Subject to:

$$\sum_{j \in Q} x_{ij} = 1, \ \forall i \in M \qquad (3.8a)$$

$$x_{ij} \leq y_{jk}, \quad \forall i \in M, \ \forall j \in Q \qquad (3.8b)$$

$$\sum_{j \in Q} y_{jk} = n_k, \forall k \in D_h \qquad (3.8c)$$

$$\sum_{k \in D_h} n_k s_k \leq C_Q \qquad (3.8d)$$

$$\sum_{k \in D_h} y_{jk} s_k \leq C \qquad (3.8e)$$

$$x_{ij} \in \{0,1\} \quad \forall i \in M, \ \forall j \in Q \qquad (3.8f)$$

$$y_{jk} \in \{0,1\} \quad \forall j \in Q, \forall k \in D_h \qquad (3.8g)$$

Expression (3.8) represents the objective function of the model. The first term accounts for the access cost of MU to QRs, the second term represents the allocation cost of hot data items to QRs. Expressions (3.8a)-(3.8g), translate the constraints: (3.8a) ensures that each MU accesses one QR at a time; (3.8b) ensures that, whenever a MU is assigned to a QR j, then a QR must have been opened at MU j; (3.8c) ensures that n_k copies of item d_k are replicated among QRs; (3.8d) guarantees that the storage of item copies must not exceed the overall storage capacity of QRs; (3.8e) ensures

that the storage space of each QR will not exceed its capacity; (3.8f) and (3.8g) are general integrity constraints.

*Proposition 1: **PRPP** is NP-Complete.*

Proof:

To prove the NP-Completeness of PRPP, we derive a polynomial reduction to the p-median problem which is known to be NP-Complete [65]. We then define an instance I1 of the p-median problem, build an instance I2 of PRPP and then show that any algorithm that resolves I1 is able to resolve I2.

Any instance I1 of the p-median problem is defined by: (a) a set of clients; (b) a set of *p* centers; (c) a cost of the link of each client *i* to each center *j*; (d) the cost of assigning a client *i* to a center *j* denotes the distance between *i* and *j*. The instance I1 being defined, let us consider the instance I2 of PRPP with the following inputs: (a) number of centers *p* equal the number of QRs; (b) there are two client sets: the MU set *M* and the hot data set D_h; (c) the cost function of MUs to be assigned to QRs is $\sum_{i\in M}\sum_{j\in Q} c_{ij}x_{ij}$; (d) the cost function of replicas to be assigned to QRs is $\sum_{k\in D_h}\sum_{j\in Q} r_{jk}h_{jk}y_{jk}$;

Any admissible solution will have all MUs and all hot data items assigned to QRs. It then appears that any algorithm that is able to resolve any instance I1 of the p-median problem is able to resolve the instance I2 of PRPP defined. Such a reduction proves that PRPP is NP-Complete ■

3.3.5 Distributed Heuristic for Joint Data Caching and Replication

Solving a large instance of PRPP can be time consuming. Given that the network topology is highly dynamic and the variables used to model PRPP are a function of time, it is not easy to collect data needed to solve it without generating huge message overhead. In particular, the access cost c_{ij} which is the distance from the MU i to the QR j is not a constant value due to node mobility. Moreover, the demand r_{jk} of the data item d_k on the QR j can change over the time so that the server may need to reallocate new items by removing some existing ones which are no longer popular.

Replica reallocation is then important when dealing with data replication and this section addresses this issue. Reallocation periods also contribute to improve the suitability of nodes acting as replica managers because, during long time intervals, a QR could lose the properties that motivated its selection. In this perspective, PRPP is solved periodically by the server. For this reason, the variables used through the above formulations are temporal due to the changing topology of MANETs. However, for the sake of clarity, the formulation is made for one replication period. As PRPP is NP-Complete and the distance between nodes in network cannot be known without a localization algorithm, we propose the heuristic **DqCa-Server** be run periodically by the server. The configuration obtained when solving PRPP is transposed on nodes to form the new network configuration. The network will operate with this configuration for a predefined period T during which data transaction takes place between nodes in the network. The periodic execution of **DqCa-Server** allows a perfect adaptation of the heuristic to the changing network topology. The following steps are executed by the server during the reallocation period.

3.3.5.1 Selection of QRs

Since the QRs have the extra responsibility to answer requests on behalf of the server, they are prone to battery drainage. Therefore, a node with good residual battery power is a good candidate for being a QR. Although, the remaining battery power is easy to measure, the rate at which it will deplete is still uncertain. The server then selects the n_{QR}^{opt} most powerful MUs, in terms of energy, to become QRs.

3.3.5.2 Replica allocation and Update

Since the demand $r_{jk}$of the replicas from the QRs is not known a priori, we consider the simple allocation models defined in [18] for unstructured peer-to-peer networks that use uniform, proportional and square root allocation strategy. These allocation strategies were originally designed for wired peer-to-peer network without bandwidth constraints. However, we adapt them to our scheme, such that replication strategy

consists to allocate n_k copies of the most popular item to the n_{QR}^{opt} QRs until the sharing space is filled. The number n_k of copies of each item on the selected QRs is defined as follows:

$$n_k = \begin{cases} n_{QR}^{opt}, & \text{for uniform allocation} \\ \frac{n_{QR}^{opt}}{s_k} \cdot C \cdot \lambda_k^a & \text{for proportional allocation} \\ \frac{n_{QR}^{opt}}{s_k} \cdot C \cdot \sqrt{\lambda_k^a}, & \text{for square root allocation} \end{cases} \quad (3.9)$$

where $1 \leq n_k \leq n_{QR}^{opt}$. To allocate items to the sharing space offered by the QRs, the server executes the following steps:

i. Sort items in descending order of their β_k;
ii. Select a replication strategy as provided by (3.9);
iii. Repeat until the sharing space is filled

    Randomly pick n_k QRs from the QR set, and assign one copy of the associate item d_k to each of them;

    Add the new item to the QR item list.
iv. Broadcast the QR list with the associated items:

$$\langle < QR_1, \{d_1, d_2, \cdots\} >, < QR_2, \{d_1, d_2, \cdots\} >, \cdots < QR_{opt}, \{d_1, d_2, \cdots\} \rangle$$

where QR_i represents the address of the QR and d_k the associated items.

Replicas updates are made on on-demand basis. Indeed, the QR downloads a fresh copy of an expired replica only when this latter is subjected to a query from MUs. In such a situation, the QR after refreshing the requested data answers the MU query. By doing so, the qCache scheme is exempted of the extra communication overhead generated by the time-to-refresh mechanism and the invalidation reports [21, 66].

3.3.5.3 Reallocation Period

Some QRs can disappear over time due to reasons such as disconnection, departure, or battery failure. Furthermore, numerous novel MUs can join the network area to request the server services, after registration and authentication. MUs' demand for

data items can also vary over time and may increase the load on the server or a particular QR. Such situations require QRs expansion to accommodate users' requests and then motivate the need to reevaluate periodically the replica placement. To trigger the reevaluation, the server advertises by requesting each node for re-authentication. This provides the server a global view of the network topology (active QRs, number of active MUs). When a node authenticates, it sends the remaining energy information as well, and the server maintains a table that updates node profile. A node's profile includes its address and remaining energy. The QRs also reports the access frequency of replicas stored, since the last replication period. These statistics allows the server to efficiently compute the hot set. With this collected information, the server is able to trigger an expansion of the number of QRs, if the MU population has increased so that the number of QRs differs from the one obtained previously. Let $n_{QR}(T^-)$ and $n_{MU}(T^-)$ denote respectively the number of QRs and the MU population computed at the previous reallocation period T^-. An expansion of QRs occurs when the number of available QRs $n_{QR}(t)$ and the MUs population $n_{MU}(t)$ including QRs, at the beginning time t of the reevaluation period, fills the condition:

$$n_{QR}(t) < n_{QR}(T^-) \text{ and } n_{MU}(T^-) = n_{MU}(t) \quad \text{or} \quad n_{MU}(t) > n_{MU}(T^-) \qquad (3.10)$$

At the end of the reevaluation process, the server broadcasts the list of the selected QRs as well as the indexes of the replicas allotted, if changes (in hot set or, in QR set) occur compared to the previous evaluation. When receive such list, the QR knows the data it has to download from the server and maintain statically. The download is done progressively when MUs request items from the QR. The MU which also receives this list can ask for items from the QR in its vicinity.

3.3.5.4 Cache Resolution

Instead of evaluating the cost of accessing the objects stored on the QR, the server publishes the list of QRs. Each time a MU needs to send a query, it checks the local cache and the NCT. If the request fails to find an answer from them, the MU selects the QR in its vicinity from the QR list that holds the requested item. When the look

up fails to get an answer from the QR list, the request is sent hop after hop to the server. Note that the cache resolution is the same as the one described previously for CoonCache except that the requester and the nodes along the path to the server have to look up the QRs list as well. Many routing protocols in ad hoc networks (e.g. AODV, DSR, DSDV, etc.) provide the hop count information between a source and a destination. For a query involving a hot data $d_k \in D_h$ hosted by a QR, the MU selects the QR with the least hop count. In the case of a tie, one QR is picked randomly. This allows the distribution of load over the QR nodes. Figure 3.3 illustrates the cache resolution process through the following data retrieval scenarios:

Q1: The node N10 find the data item, valid in its local cache and serves it to the requesting application.

Q2: The node N10 requests a data item but does not find it in local cache, and realizes that the item is valid in its neighborhood, when looked up in the NCT. It then forwards the request to the caching neighbor node N1 which answers by sending the data back.

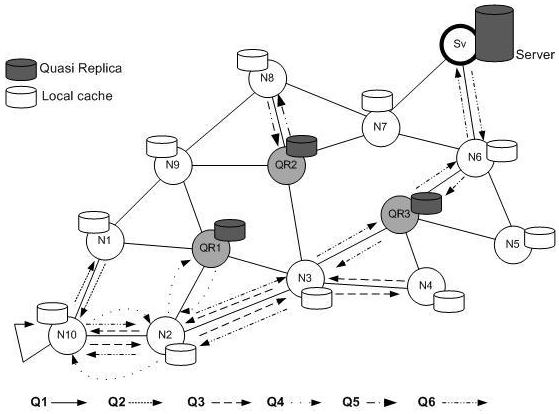


Figure 3.3 Request resolution process in qCache scheme

Q3: The request from node N10 fails to get an answer either from the local cache or from the NCT. In such case, the request is forward to the next hop node N2 in the path toward the server. This latter executes Q1 and Q2. Fortunately N3 which is in the path and realizes that in the NCT, the node N4 has the requested data valid. It forwards the request to N4 which sends the data back along the path N4-N3-N2-N10.

Q4: The requested data is not found anywhere around node N1 which forwards the request to N2. This latter found that the requested data is a popular one and is located in the QR1. N2 then forwards the request to QR1 which sends the data back along the path QR1-N2-N10.

Q5: The data requested by N8 is not found in local cache, or in the NCT. N8 found that it is a popular item and sent the request to QR2 which owned the data. QR2 answers by sending the data back.

Q6: The data requested by N10 is not found anywhere around or in the intermediate nodes in the path N10-N2-N3-QR3-N6-Sv toward the server. In this case, the request is forwarded to the server which sends the data back. Along the reverse path, intermediate nodes refresh the data if it exists in local cache.

3.3.5.5 Topology Changes

To ensure that the promoted QRs are active in the network, the server maintains a table that updates their entries every time a QR request is received. A refreshing "HELLO" message is also sent by the QRs to the server to allow the latter to ensure their availability. The underlying routing protocol messages can embody such refreshing message to avoid an extra communication overhead. If a QR becomes unavailable, the server destroys the corresponding profile. Periodic broadcast of the active QR list is initiated by the server to allow joining MUs to handle their requests accordingly. Receiving this list allows MUs to maintain their QRs' table up to date. When a QR temporarily disconnects from the network the replicas may remain valid as long as their TTL values are not expired. After a reconnection a QR must validate those items that are expired by requesting them from the server with one or several download requests. Downloads are made when the QR receives a request from MU for expired items. A joining nodes which miss the QR list due to temporary disconnection, must run the CoonCache caching strategies, until it is noticed of the QR list.

3.3.5.6 Algorithms

The pseudo-code executed by the server is given through the DqCa-Server algorithm and the one executed by the nodes in the network is illustrated by the DqCa-MU algorithm given in Appendix A.

3.4 Performance Evaluation

In this section, the performance of the proposed models is assessed. A mathematical analysis of CoonCache and qCache is first provided followed by the simulation experiments. The goal is to show theoretically and experimentally the effectiveness of these models by comparison with other MANET caching strategies. In particular, the effectiveness of the CoonCache caching strategy is first highlighted. Then, we show that the addition of partial data replication to CoonCache which defines the qCache scheme outperforms all of the studied caching strategies. Baseline comparisons are with respect to:

- The *SimpleCache* scheme [24] in which each node has a local cache and user's requests are forwarded to the server if local cache misses.
- The *HybridCache* scheme [24] in which the requested data item is checked at every forwarding node. If the forwarding node has the requested data in its local cache, it stops forwarding and sends back the data. Pass by data items in this scheme are updated under optimization conditions.

3.4.1 Performance Analysis

In this section, we develop an analytical model for the proposed caching schemes on the cost associated with communications among the caching nodes, the quasi replica and the data server. The goal is to determine the average path length experienced by the queries to reach the node which owned the requested data. Average path length functions are also derived for SimpleCache and HybridCache. The notations used for this analytical study are as follows:

- L_{Simp}, L_{Hyb}, L_{qCac}, L_{Coon}, denote respectively the average path length follows by a request to reach the node which has the data in SimpleCache, HybridCache, qCache and CoonCache;
- H: denotes the average number of hops between a mobile node and the data server; q_h the probability that the data item is in QR and p_h the probability that the data item is in MU local cache; ρ and R denote respectively the node density and transmission range.

Let us now compute the average path length. In SimpleCache scheme, after a local miss with probability $1 - p_h$, the request is forwarded to the server. The expected number of hops that a query takes from a given node to the node which has the data is then:

$$L_{Simp} = H(1 - p_h) \tag{3.11}$$

In HybridCache scheme, since the request is resolved hop-by-hop, the expected number of hops that a query takes from the requesting node to the node which has the data is:

$$L_{Hyb} = \sum_{i=0}^{H} i p_h (1 - p_h)^i \tag{3.12}$$

The closed form of the formula expressed by (3.12) can be obtained by rewriting L_{Hyb}:

$$L_{Hyb} = p_h(1 - p_h) \sum_{i=0}^{H-1} \frac{\partial}{\partial p_h}(1 - p_h)^i + H(1 - p_h)^H \tag{3.12a}$$

By reducing (12a), we finally get the closed form of the expected number of hops:

$$L_{Hyb} = \frac{1}{p_h}[(1 - p_h)(1 - (1 - p_h)^H)] \tag{3.12b}$$

In the proposed caching scheme CoonCache, the probability of getting the requested data from the neighborhood is given by taking into account: (a) the probability of local cache hit p_h; (b) the number of entry in the NCT which is equal to the number of one-hop neighbors $\rho\pi R^2 - 1$;(c) the probability of cache hit in the NCT which is the probability that there is a hit in the requesting node neighborhood, $1 - (1 - p_h)^{\rho\pi R^2 - 1}$. The average path length can then be written as:

$$L_{Coon} = 1 - (1 - p_h)^{\rho\pi R^2 - 1} + \sum_{i=1}^{H-1} i\left(1 - (1 - p_h)^{\rho\pi R^2}\right)(1 - p_h)^{i\rho\pi R^2} + H(1 - p_h)^{\rho\pi R^2 H} \quad (3.13)$$

This expression can be developed as follows:

$$L_{Coon} = 1 - (1 - p_h)^{\rho\pi R^2 - 1} + \left(1 - (1 - p_h)^{\rho\pi R^2}\right) \sum_{i=1}^{H-1} \frac{-(1 - p_h)}{\rho\pi R^2} \cdot \frac{\partial}{\partial p_h}(1 - p_h)^{i\rho\pi R^2} + H(1 - p_h)^{\rho\pi r^2 H} \quad (3.13a)$$

After reduction, the closed form of the average path length in CoonCache is:

$$L_{Coon} = 1 - (1 - p_h)^{\rho\pi R^2 - 1} + (1 - p_h)^{\rho\pi R^2} \cdot \frac{1 - (1 - p_h)^{\rho\pi R^2 H}}{1 - (1 - p_h)^{\rho\pi R^2}} \quad (3.13b)$$

By using the same logic as for CoonCache, the average path length L_{qCac} in the qCache scheme can be expressed as shown below considering that in our architecture, QR node is expected to be located at one hop from the requester and that the number of entry in the NCT is $\rho\pi R^2 - 2$:

$$L_{qCac} = 1 - (1 - p_h - q_h)^{\rho\pi R^2 - 2} + \sum_{i=1}^{H-1} i\left(1 - (1 - p_h - q_h)^{\rho\pi R^2}\right)(1 - p_h - q_h)^{i\rho\pi R^2} + H(1 - p_h - q_h)^{\rho\pi R^2 H} \quad (3.14)$$

The closed form of this expression is then:

$$L_{qCac} = 1 - (1 - p_h - q_h)^{\rho\pi R^2 - 2} + (1 - p_h - q_h)^{\rho\pi R^2} \cdot \frac{1 - (1 - p_h - q)^{\rho\pi R^2 H}}{1 - (1 - p_h - q)^{\rho\pi R^2}} \quad (3.14a)$$

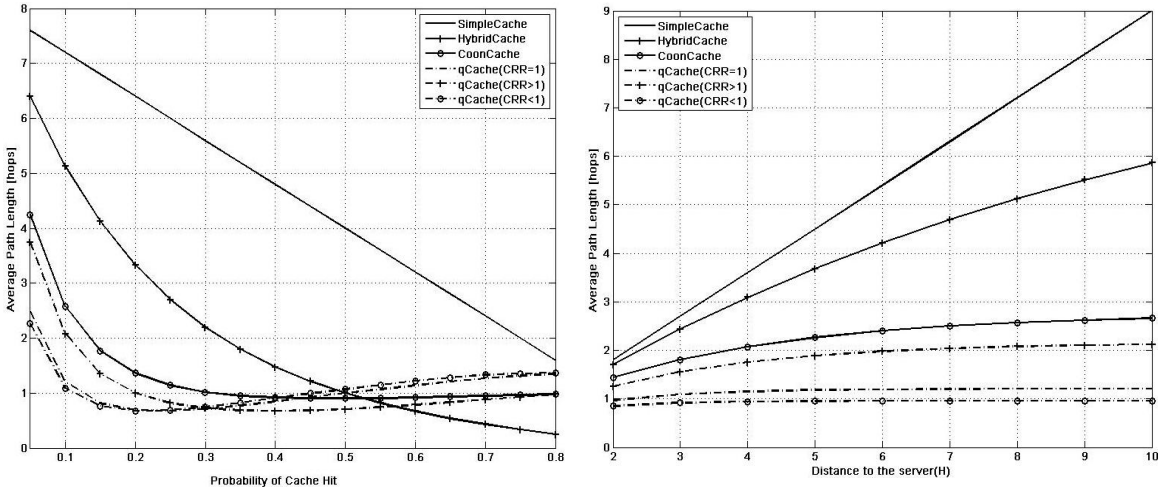
Figure 3.4 (a) and 3.4 (b) show the plot of the average path length when varying the value of p_h, and that of the average number of hops to the server (H) which reflects the network scale. It appears that as p_h increases, the average path length decreases in all schemes. The cooperative caching strategy CoonCache shows a huge gain in terms of number of hops, over the local caching strategies SimpleCache and Hybridcache. Furthermore, the use of partial data replication in qCache enhances the decrease of the path length to get the requested data. In particular the effect of caching to replication ratio (CRR) defined as $CRR = {p_h}/{q_h}$ is highlighted. The CRR shows the marked effects of partial data replication. Depending on the data popularity, the

probability q_h of finding the requested data on a QR compared to the probability p_h of finding the data in local cache suggested the following observations:

- When $CRR \leq 1$, the requested data is expected to be retrieved with a few number of hops from a QR with the qCache scheme as shown in Figure 3.4(a). This should occur when the hot items cannot be found in local cache so that the request has to be forwarded to the QR to be resolved.
- When $CRR > 1$, even if the benefit of using qCache tends to the one achieved with Cooncache scheme, qCache once more performs better compared to the plain caching strategies.

Finally, in all cases, qCache scheme outperforms the plain caching strategies and shows theoretically that using replication in conjunction with caching improves dramatically data accessibility, in terms of response time, for values of p_h and q_h that fill the conditions:

$$|1 - p_h - q| < 1 \text{ and } |1 - p_h| < 1. \quad (3.15)$$



(a) Effects of the cache hit probability on the average path length

(b) Effects of the network scale on the average path length

Figure 3.4 Theoretical performance evaluation of the caching strategies.

Figure 3.4(b) shows the plot of the average path length in the studied schemes, when varying the average number of hops to the server (H) for $\rho = 1$ and $R = 1$. As

illustrated, the average path length increases in all schemes with the difference that qCache scheme is less sensitive to this increase compared to the plain caching strategies. It is easy to see under condition (3.15), that:

$$\lim_{H\to\infty} L_{Simp} = \infty$$

$$\lim_{H\to\infty} L_{Hyb} = 1 - \frac{1}{p_h}$$

$$\lim_{H\to\infty} L_{Coon} = 1 - (1-p_h)^{\rho\pi R^2 - 1} + \frac{(1-p_h)^{\rho\pi R^2}}{1-(1-p_h)^{\rho\pi R^2}}$$

$$\lim_{H\to\infty} L_{qCac} = 1 - (1-p_h-q_h)^{\rho\pi R^2 - 2} + \frac{(1-p_h-q_h)^{\rho\pi R^2}}{1-(1-p_h-q_h)^{\rho\pi R^2}}$$

These evidences justify why the qCache strategy will always offer shorter average path lengths under conditions (15) compared to the plain caching strategies CoonCache, HybridCache and SimpleCache because $L_{qCac} \leq L_{Coon} < L_{Hyb} < L_{Simp}$ when $H \to \infty$.

3.4.2 Simulation Experiments

In this section, extensive simulations have been conducted regarding performance evaluation of the proposed scheme. In this perspective, we have implemented CoonCache and qCache as well as SimpleCache and HybridCache caching models in Qualnet simulator [67]. The goal is to compare these schemes with respect to various metrics such as the *Average Delay* and the *Hit ratio* under different situations: cache size, number of nodes, various Zipf request patterns and the *write-to-read ratio* (WRR) of the requested items. In particular, the Average Delay is defined as the time elapsed between a submitted query and the response; the Hit Ratio refers to the measurement of three sub-metrics: the *Local Hit Ratio* defined as the ratio between the number of responses obtained from the cache memory to the number of total responses received by the node; the *Remote Hit Ratio* defined as the ratio between the number of responses received by a node from other MUs (except QRs) nodes to the number of overall responses; the *QR Hit Ratio*, defined as the number of responses received from the QR nodes to the number of overall responses. The WRR is the ratio

between the mean TTL of the data items T_{Update} and the mean query time T_{Read}. For each experiment, 20 simulations were conducted and the mean value of the measured data is used to show the results. All schemes use the *Least Recently Used* (LRU) as cache replacement policy. Later in the section 4.2, the effects of the proposed cache replacement policy Greedy-U are shown on CoonCache and qCache schemes by comparing with the LRU caching policy.

3.4.2.1 Simulation Setups

In this section we describe the network setup, the client query model and the server model, and summarize all the simulation parameters in Table 3.1.

Network setup: The studied caching schemes are implemented on ad hoc nodes distributed randomly over a (1000 m × 1000 m) area. The mobile units use an 802.11b wireless interface in ad hoc mode with a bandwidth of 2 Mbps. For each MU, the transmission range was set to 250 m. Free space signal propagation model is adopted for the radio transmission. The underlying routing protocol used is AODV. The node density is changed by setting the number of nodes between 50 and 100.

Server model: The server is represented by a node whose bandwidth is set to 2 Mbps. The server is placed in the top-left corner of the area, i.e., at location (950 m, 950 m) and stores 2000 data items. Item size is uniformly distributed between s_{min} and s_{max}. The data are originally updated on the server which served MUs' requests on the First-Come, First-Served basis. When the server sends a data item to a MU, it sends the TTL of this data as well. The TTL value is set exponentially with a mean value T_{Update}.

MU model: MUs move according to the random waypoint model in which each client selects a random destination and moves towards it with a speed selected randomly in the interval [0, 5 m/s] with a pause time equal to 200 seconds. Each MU generates a single stream of read only queries and each new query is separated from the next one by an exponentially distributed think time T_{Read}. The request pattern follows Zipf-Like distribution which has been used to model various behaviors such as the web page

request pattern [68, 69]. The normalized expression of this distribution is $\lambda_k^a = 1/\left(k^\theta \sum_{i=1}^{m} \frac{1}{k^\theta}\right)$, is used to compute the access probability to the *k*-th data item. The parameter θ represents the Zipf factor. If $\theta = 0$, the MUs uniformly access the data items. As θ increases, the access to the data item becomes more skewed, i.e., more concentration is placed on popular data items. The value of θ varies for different applications. In our simulation, its default value is set to $\theta = 0.8$, as per the studies provided in [68]. All simulation parameters are listed in Table 3.1.

Table 3. 1 Simulation Parameters

Parameter	Default value	Range
Number of items	2000	
Number of nodes	80	50 to 100
$s_{min}(KB)$	0.5	
$s_{max}(KB)$	5	
Transmission range (m)	250	
Zipf Factor (θ)	0.8	0.2 to 1
Replica/Cache size (KB)	60	40 to 160
Bandwidth (Mbps)	2	
Simulation area (m x m)	1000 x 1000	
Speed (m/secs)		0 to 5
Pause time (secs)	200	
Mean query generate time T_{Read}(secs)	10	
Write Read Ratio(WRR)	30	10 to 35
Mean Time-To-Live T_{Update} (secs)	300	100 to 350
Simulation time *T* (secs)	7200	

3.4.3 Simulation Results

We present in this section the simulation results comparing the plain caching strategies Simplecache, HybridCache and CoonCache with the partial replication strategy qCache. In particular with qCache, we study the effect of using different replication strategies on the QR. Uniform, proportional and root-square replication strategies are examined through qCache-uni, qCache-prop and qCache-root. In these respective variants of qCache, the data items stored by the server are replicated on the selected QRs according to the formula (3.9).

3.4.3.1 Effects of the Cache Size

Figure 3.5 and 3.6 show the average delay and the hit ratio when varying the cache size. The average delay is closely linked to the Hit ratio. Indeed, if the hit ratio is high, better is the average delay since the MU can process most queries locally, or from MUs in neighborhood, or from a QR, without requiring the server's help. As shown in Figure 3.5, when the cache size grows, the average delay drops exponentially for all schemes. The trends obtained for the average delay are very similar to the one provided by the mathematical analysis in Figure 3.4(a). qCache scheme performs better and its performance over the other schemes is due to the fact that popular data are served from QRs which are close to the requesting node. In particular, the qCache-uni scheme performs less compared to other variants of qCache since the QRs store the same number of popular items. The qCache-prop scheme outperforms the other schemes, because QRs are populated with a broad range of popular items compared to the qCache-root.

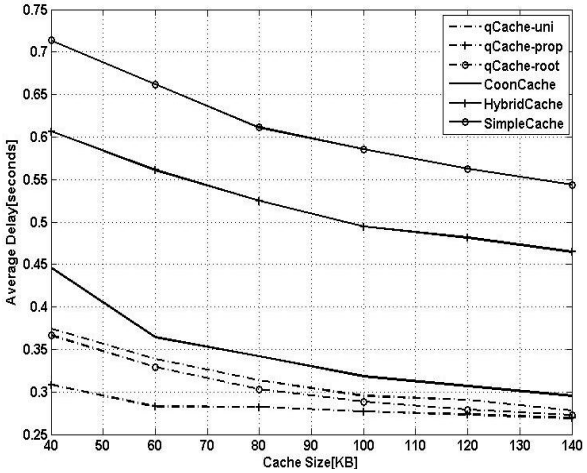
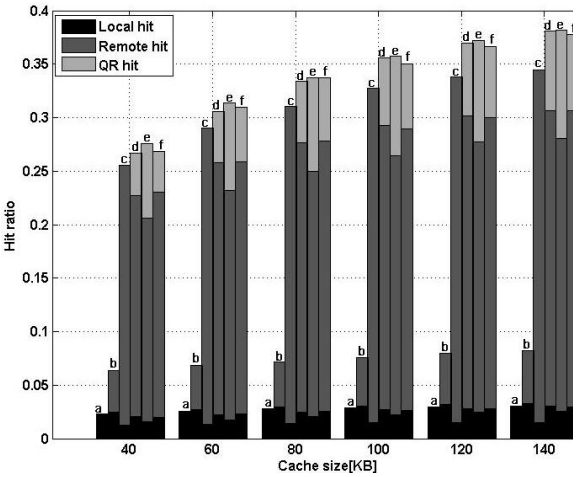


Figure 3.5 Effects of the cache size on the average delay



(a) SimpleCache, (b) Hybridcache, (c) CoonCache (d) qCache-uni (e) qCache-prop;(f) qCache-root

Figure 3.6 Effects of the cache size on the Hit ratio

The slim performance of qCache-root over qCache-uni is due to the fact that QRs store a few more items. The performance of the average delay reflects the gaps shown on Figure 3.6 between the Hit ratios experienced by the respective schemes. At least a

20% increase of the hit ratio is achieved when the CoonCache scheme is used compared to the HybridCache one because in addition of the local hit, the requesting node is able to retrieve data from its neighborhood by using the NCT, or from neighborhood nodes in path to the server. Moreover, the fact that the nodes in vicinity of each other store different data items contributes in large part of the increase of the Hit ratio. This lead is also enhanced by the popular data served from QRs and centers around 25% with the addition of the QR hit ratio in qCache schemes. It is worth noticing that Hybridcache performs better than SimpleCache because data are not only served from local cache but also from other nodes in paths to the server, while CoonCache outperforms HybridCache because data are served from local cache and remote cache.

3.4.3.2 Effects of the Node Density

Figure 3.7 shows the average delay as a function of the number of nodes in the network. As the number of nodes increases, the delay of all schemes grows exponentially. This trend is similar to the one obtained theoretically in Figure 3.4(b). The trend can be explained by the fact that more requests are initiated by additional nodes so that the load on the server is increased. More nodes compete for the limited bandwidth and consequently this increases the delay to get the requested data. However, the CoonCache and the qCache schemes perform better compared to other SimpleCache and HybridCache due to the accesses to the data in a short number of hops. Furthermore, it is worth noticing that the benefit with the qCache schemes is only significant in a densely populated MANET, as indicated by the higher values of node density. Figure 3.8 shows the associated hit ratio, which reflects the trends of the average delay. The QR hit ratio decreases as the number of nodes increases, providing evidence of load distribution among the QRs.

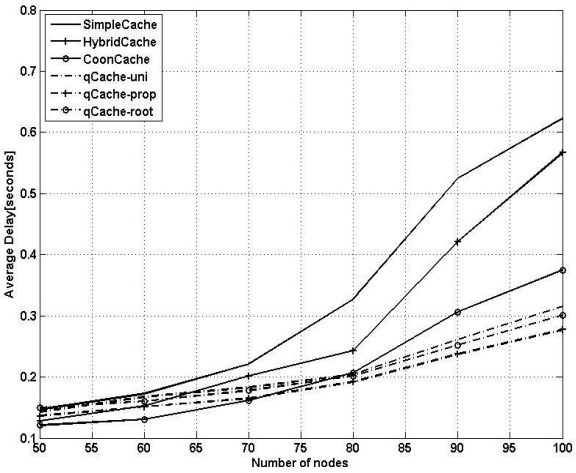
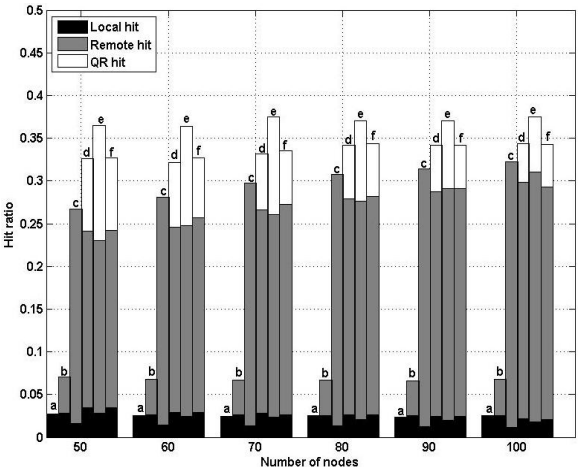


Figure 3.7 Effects of the density on the average delay



(a) SimpleCache, (b) Hybridcache, (c) CoonCache (d) qCache-uni (e) qCache-prop;(f) qCache-root

Figure 3.8 Effects of the density on the Hit ratio

3.4.3.3 Effects of Skewness

Figure 3.9 and 3.10 show the effects of the Zipf factor on the system's performance. As this coefficient grows, the user's access is focused on lower ranking items, the popular ones. In spite of the Zipf factor variation, the qCache schemes always guarantee shorter access delays than the other schemes. As the Zipf factor grows, it improves the average delay for all schemes. At the same time, the hit ratio improves as this factor grows. Indeed, the increase of this factor concentrates the request on a smaller percentage of the data items, thus improving the hit ratio and the average delay. Unlike the other schemes, with the qCache schemes, the fact that popular items are replicated on QRs explains the performance of the combined partial data replication and caching over the plain caching strategies.

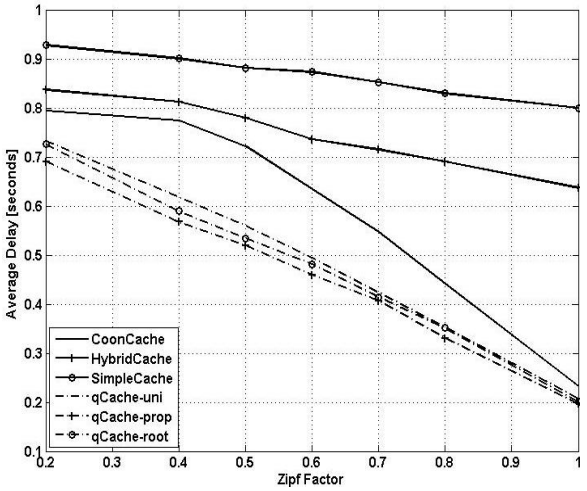
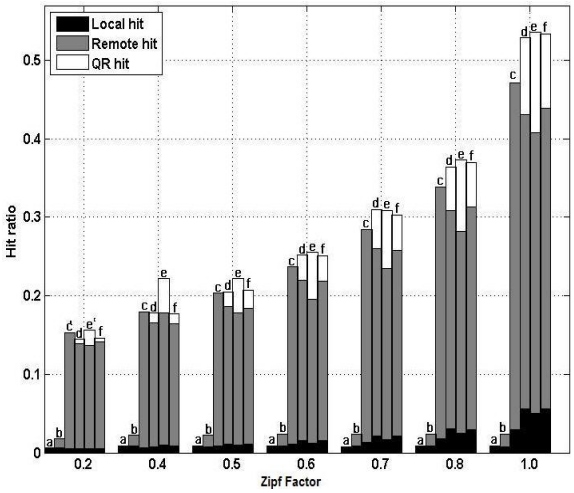


Figure 3.9 Effects of the zipf factor on the average delay



(a) SimpleCache, (b) Hybridcache, (c) CoonCache (d) qCache-uni (e) qCache-prop;(f) qCache-root

Figure 3.10 Effects of the zipf factor on the Hit ratio

3.4.3.4 Effects of the Write to Read Ratio

Figure 3.11 and 3.12 show the average delay and the related hit ratio when varying the WRR. As can be seen, when the value of the WRR grows, data update rate decreases and makes data items more available to be accessed in cache. Consequently, the average delay decays for the respective schemes. As SimpleCache does not allow nodes to cooperatively share their cached items, its average delay is higher than those experienced with other schemes. CoonCache performs less than the qCache schemes because the items stored on the QR nodes are more often consistent; hot data stored by QR nodes has the largest Time-To-Live among the server items; this guarantees that the qCache schemes will always perform better than the others. Lower values of the WRR drop the hit ratio because most data in the cache are invalid so that the cache miss is higher.

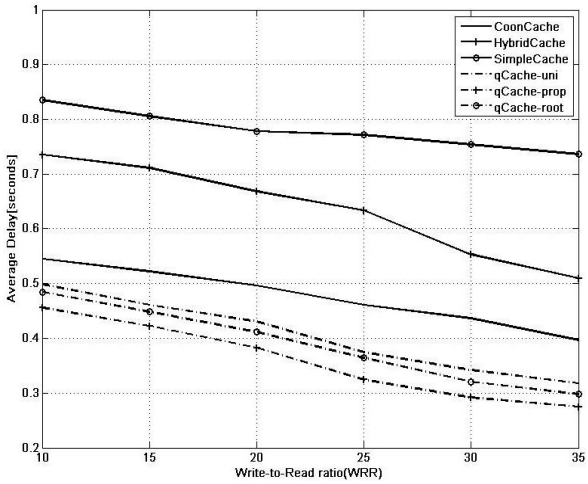
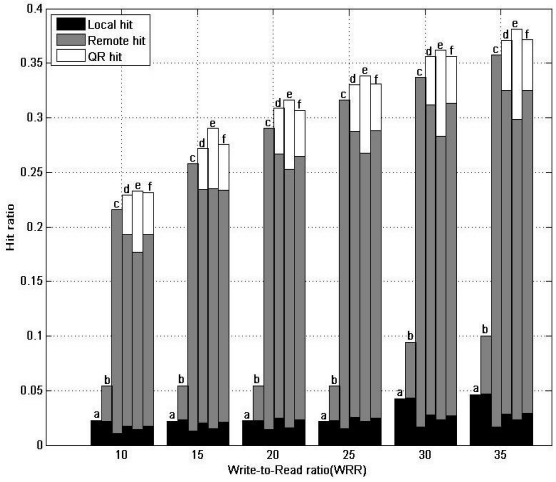


Figure 3.11 Effects of the WRR on the Average Delay



(a) SimpleCache, (b) Hybridcache, (c) CoonCache (d) qCache-uni (e) qCache-prop;(f) qCache-root

Figure 3.12 Effects of the WRR on the Hit ratio

3.4.3.5 Effects of the Replacement Policy

To evaluate the impacts of the proposed cache replacement policy Greedy-U (GU), we compare it to the least recently used policy (LRU) sets for the previous simulation. For that purpose, both CoonCache and q-Cache are evaluated under this new replacement policy. In particular, we select qCache-prop to make this comparison. Figure 3.13 and 3.14 plot the average delay and the hit ratio achieved with LRU and GU when the cache size is varied. The hit ratio for LRU policy is always lower whatever the caching scheme is, when compared to the GU policy. The GU policy performs better as it considers a broad range of parameters such as the data size, data TTL and data access frequency through the data utility function. The gain achieved with the hit ratio when using the GU policy is almost 10% higher compared to the LRU policy. This contributes to reduce the average delay. As the cache size increases, most items can be found in local and remote cache so that the query latency decreases.

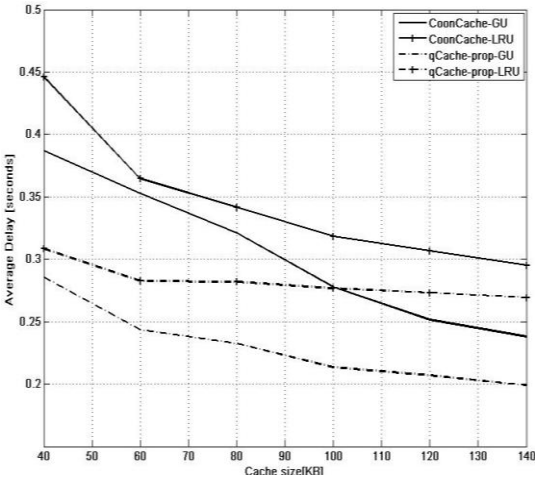
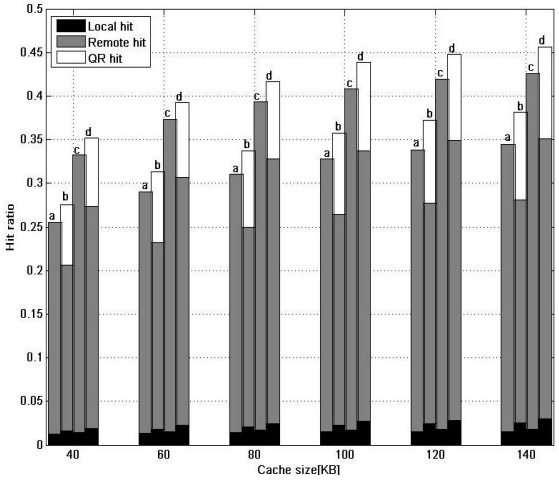


Figure 3.13 Effects of the replacement policies on the Average delay



(a) CoonCache-LRU, (b) qCache-prop-LRU (c) CoonCache-GU (d) qCache-prop-GU.
Figure 3.14 Effects of the cache replacement policies on the Hit ratio

3.5 Conclusion

In this paper we have proposed two data sharing systems that aim to improve data accessibility in mobile ad hoc networks. The CoonCache caching system allows mobile nodes to access a broad range of data items in the network by linking the cache space of nodes in vicinity of each other. The qCache scheme built over CoonCache proves that jointly using cached data and replication mechanisms allows data accessibility to perform very well. The proposed architecture consists of nodes that cache data and others that replicate the most popular data accessed by users. This kind of architecture is well suited to access Internet information and MANET services. More specifically, we formulate the issue of the partial data replication as an optimization problem and outline a heuristic to solve the related challenges. Furthermore, we have designed a cache replacement policy that brings together parameters such as the data size, the data access frequency and the data Time-To-Live, to deliver an efficient cache admission mechanism.

A simulation-based performance study was conducted to examine the efficiency of the proposed schemes with variations ranging from the effects of node density to those of replacement policy. The proposed schemes induced drastically low

costs for data accessibility. Moreover, the replica allocated to the selected QR by using the MU population helped reduce the server load while improving the overall hit ratio. The results also demonstrate an improved mean access delay compared to certain caching schemes.

CHAPITRE 4

SUBCAST: A DISTRIBUTED ADDRESSING AND ROUTING SYSTEM FOR LARGE SCALE WIRELESS SENSOR AND ACTOR NETWORKS

Therence Houngbadji*, Samuel Pierre
Department of Computer Engineering, École Polytechnique de Montréal
E-mail: { Therence.Houngbadji, Samuel.Pierre }@polymtl.ca
P.O. Box 6079, Centre-Ville Station, Montreal, Quebec, H3C 3A7, Canada
Phone: 514 340-3240 ext. 4685; Fax: 514 340-3240

Abstract

Wireless Sensor and Actor Networks (WSANs) are made up of a large number of sensing devices which are resource-impoverished nodes and powerful actuation devices: both are equipped with computation and communication capabilities. These devices cooperate to manage sensing and perform acting tasks. Numerous work conducted in the field of WSANs assumes the existence of addresses and routing infrastructure to validate their proposals. However, assigning addresses and delivering detected events in these networks remains highly challenging, specifically due to the sheer number of nodes. To address these issues, this paper proposes SubCast, a novel distributed address assignment and routing scheme based on a Topic Clustering System and fractal theory Iterated Function Systems. In order to minimize data delivery costs among actors, the proposed architecture first builds an actor overlay network before allocating addresses to network nodes. Location information in the allocated addresses allows establishing data delivery paths. Simulation results confirm that the proposed system efficiently guarantees the allocation of unique

*Corresponding author. Phone:+1 514 340 4711 x 2127
E-mails: therence.houngbadji@polymtl.ca (T. Houngbadji), Samuel.Pierre@polymtl.ca (S. Pierre).

addresses and performs efficient data delivery while reducing communication costs, delays as well as the impact of imprecise locations.

Keywords: Wireless Sensor and Actor networks; Distributed Address Allocation; Iterated Function Systems, Publish/Subscribe; Routing.

4.1 Introduction

Recent innovations in wireless communication technologies have raised interest in Wireless Sensor and Actor Networks (WSANs), which refers to a group of sensors and actors linked through a wireless medium to perform distributed sensing and actuation tasks. Sensors gather information from the physical world (data regarding temperature, pressure, movement, light, etc.), while actors make decisions and perform relevant actions upon the environment by means of actuators, which allow remote, automated interactions with the environment. Actors are resource-rich devices with networking-related functionalities and extended battery life. They are more expensive than sensors associated with low cost, inadequate power and short transmission range. Furthermore, the number of sensor nodes deployed in a target area may be in the order of hundreds or thousands, while such a dense deployment is usually unnecessary for actor nodes [2].

The collaborative sensing and acting tasks of WSAN nodes enables sensors and actors to establish data paths. Also, a coordination mechanism between actors allows them to both estimate the characteristics of events and select actors or subsets of actors which are best suited to perform a given action [70, 71]. As such, WSANs can be regarded as a self-organized, event-sensing, communicating and decisive action loop. Aside from the WSANs coordination framework, its network layer also deserves special attention. Usually, node identifiers are assumed to exist. However, the way addresses are allocated remains an open issue. Indeed, addresses can be used to establish a node position in the physical world, which is useful to identify the physical source of sensed data, especially with mobile nodes or unplanned node placements. The fact that controlled communication is considered more effective than flooding makes address

allocations an outstanding routing support. That was the case for Directed Diffusion [5, 72] and TAG [73], for which gradient and Tree-based addressing and routing were used. Addressing is required to prevent redundant ad-hoc communication between sensor nodes. Reducing intra-network transmission can save node energy, thus extending network lifetime. During operations and control, specific nodes need to be addressed in order to update software modules, calibrate sensors and perform localization tasks.

The special data reporting behavior and the need to select a given actor to perform an actuation task makes WSAN address assignment problems significantly different than those of traditional networks. For applications in which events take place in different locations, it may also be necessary to send events to a set of actors which may be far from or even totally outside the event area. The closest actor in the event area may not be the most appropriate one to handle properly the required action if commands from suitable actors that can perform the set of tasks to be accomplished in the event area are lacking. In such a context, event dissemination to actors can be trivially implemented by flooding all events to all network actors. Although such a solution may seem trivial, its cost is very expensive as all actors are affected every time an event is detected. This raises the issue of delivering events to the most appropriate network actors.

In order to tackle the issue of addressing and data delivery in WSANs, this work proposes a distributed addressing and routing system. For that purpose, fractal theory *Iterated Function Systems* (IFSs) [74] is used to discretize the network area into micro-scale areas, referred as cells that offer a distributed low-cost address allocation to network nodes. In order to solve event dissemination issues among actors, they become clustered according to their own interests' subscription. Actors are grouped into topic segments that form what is referred as the *Actor Overlay Network* (AON). Called *SubCast*, the overall architecture guarantees unique identifications for every node, as well as routing information for all possible packet delivery. The remainder of this paper is organized as follows. Section 4.2 provides an overview of existing work pertaining to

node addressing. Section 4.3 offers an architectural overview of SubCast and Section 4.4 details its architecture functionalities. Section 4.5 presents results ensuing from a performance evaluation, a simulation result analysis. In Section 4.6 conclusion sheds light on future work.

4.2 Related Work

Although a few recent papers delve specifically into nodes addressing, the literature pertaining to addressing in Wireless Sensor Networks (WSNs) is extremely limited, and that of addressing in Wireless Sensor and Actor Networks remains quasi non-existent, in spite of being described as an open research issue at the network layer in [2]. A number of solutions have been proposed to allocate addresses dynamically. The best-known approach is the Dynamic Host Configuration Protocol (DHCP) [35] used in IPv4 and IPv6 networks, which consists of a client-server protocol composed of two major building blocks: a protocol to deliver specific parameters to the client and an IP address selection mechanism. Although the dynamic profile of this protocol is adequate for WSANs, it is hindered by the huge overhead needed to allocate addresses. Signaling overhead represents a major source of misused energy. In addition, the centralized address management scheme used by this protocol cannot scale well in highly dense WSANs. Although a DHCP-Like addressing structure was proposed for WSNs, it cannot be qualified as perfect. The addressing systems in [37-39, 75] are not scalable nor energy efficient. Their allocation process includes the exchange of numerous messages that cause serious energy burdens. More particularly, Yao et al.'s proposal [39], a round-based allocation process, deals with duplicate detection of allocated addresses, causing the exchange of numerous messages. The DHCP-like addressing approach can be appealing for low scale WSNs, yet it must be adapted for large networks, such as WSANs, in order to generate lower signaling overhead. In [76], the authors propose a dynamic address allocation scheme for query-based sensor networks. A temporary network-wide unique address is allocated on demand, solely to sensor nodes that report data in response to an explicit query from the sink. The approach splits the geographic

area into blocks and 2D logical grids, so that each node lies within a single grid. Unique addresses are assigned locally to sensors positioned within a cell of a predetermined size. This approach fails to lighten the heavy overhead used for allocating addresses. In addition, the central approach adopted to allocate active addresses to nodes may create bottlenecks as the network expands.

Others addressing systems [40, 42, 43, 77] divide the network into layers, and the sensor nodes add a suffix to addresses of nodes in last layer to form their own addresses. In such a layered allocation process, addresses become increasingly longer as the network expands. More particularly, the TreeCast addressing scheme [78], a global addressing and stateless routing architecture for sensor networks, adopts this strategy. In this scheme, nodes are organized in a tree structure and addresses are assigned according to node positions. Child nodes randomly generate their own addresses, which are subsequently approved by parent nodes. TreeCast does not scale well in large networks as the parents' address strictly consists of the prefix of their children's address, thus increasing address lengths as node levels grow. Following the same design approach, the *ZigBee* network layer [43] provides a distributed address assignment while routing management handles tree-routing mechanisms as well as AODV-like reactive routing mechanism. Before forming a network, the ZigBee coordinator determines the maximum number of children for a single router, the maximum number of child routers for a single router, as well as the network depth. Devices' addresses are assigned in a top-down manner. For the coordinator, the whole address space is logically partitioned into blocks: some blocks are assigned to the coordinator's child routers while others are reserved for the coordinators' own child end devices. The hierarchical block addressing mechanism of ZigBee protocol is subject to the waste of the 16-bit address space (especially for small networks), the limitation of the network expandability (limitation on the maximum number of children) and the lack of device mobility. Other works offer a new design avenue using cluster-based address allocation. Such a strategy is scalable

and thus deserves a closer look. Works in [45, 46, 79] have adopted this design philosophy.

4.3 Design Overview

By examining the existing work listed above, it appears that address assignment in WSANs remains largely an open issue. This section provides the design rationale of the components used by the proposed addressing system. As pointed out in [2], in certain applications, when events take place in different locations, it may also be necessary to pass events to the set of actors which may be far or outside the event area when the event is detected, but rather to the set of actors which is closer to the event when it is notified. Indeed, without commands from suitable actors that can issue the necessary set of tasks, the closest actor in the event area may not be able to handle properly the required action. In such a context, the event disseminated to a set of actors can be trivially implemented by flooding each event among actors, so that they can filter out events that do not match their interests' subscription. However, flooding among actors is costly, as each actor is affected every time an event is detected. To address such issues, an Actor Overlay Network (AON) is built, based on the *Publish-Subscribe* (PS) communication paradigm. Indeed, the PS paradigm was proven to let information propagate from publishers to interested subscribers in an anonymous and decoupled fashion [80]. In order to deliver events that match the actors' specific subscriptions while keeping *interest clustering* in mind, it is necessary to use a system with traffic confinement to build an AON. Actors that share common interests are clustered so that once an event reaches a cluster member, its dissemination can be limited to the cluster itself. This way, the cluster member that can act properly is immediately selected.

The second design goal of SubCast is to allocate addresses to network nodes. As discussed above, previous works on address allocation focuses only on address allocation in WSNs without integrating how the allocation process can affect the moving actor nodes in WSANs. Furthermore, numerous studies conducted in WSANs [70, 71],

[81, 82] assume the existence of node identifiers and routing protocol to validate their proposal. To solve the issue, this paper proposes a new addressing and routing protocols for WSANs, based on fractal theory Iterated Function System (IFS) [74]. The usability in mobile or indoor environments of GPS-enabled devices appears to be costly in real deployments even if the solution is attractive. Location approximation using localization algorithm can help achieve low-cost address allocation despite their lack of precision. The conjecture of this study is that partitioning the network into micro-scale areas with a minimal amount of physical location information, even if inaccurate, can offer real-world decentralized addressing system in large scale WSANs regarding the analysis of existing approaches. That is where the use of IFS becomes a valuable choice to discretize the network area with zero overhead. Indeed, it offers the possibility of using approximate or relative location information to determine node areas. Accordingly, it is more robust to errors and imprecisions in location measurements and estimates compared to schemes that depend on exact location information. As an outstanding feature, IFS generate an explicit area labeling based on a predefined coded alphabet. The partitions obtained provide a dampening factor that reduces the effects of mobility. Local node movements, meaning movements within their own regions, do not affect the SubCast addressing structure. Finally, in addition to allowing the design of a distributed address allocation scheme, the use of IFS enables a new generic event routing strategy for WSANs. Thus, the following design objectives are met:

(a) Low-cost address allocation: less signaling overhead is generated;
(b) A unique address is allocated to each node in the network;
(c) Event routing is performed by using only the node's neighborhood; path discovery to the destination and routing table maintenance are not needed.

Figure 4.1 gives an overview of the building blocks that compose the SubCast Network Layer, as well as their respective functionalities. In the following section, details are provided on the way the illustrated functionalities are built.

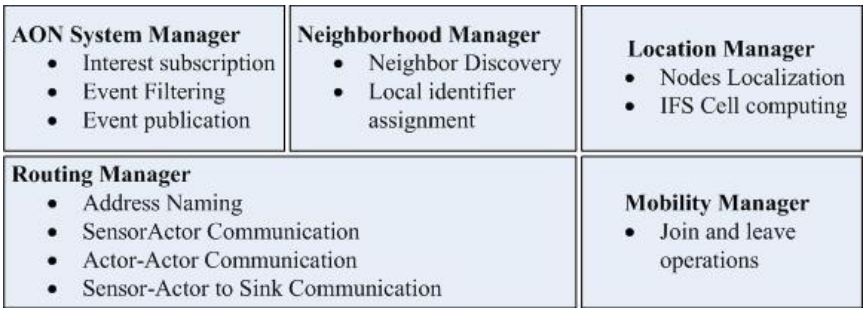


Figure 4.1 SubCast Components and their Functionalities

4.4 System Model

The assumptions pertaining to the SubCast addressing system are the following: the WSAN is deployed over two-dimensional areas and the architectures described in [2] are considered: a *semi-automated* architecture where sensors route data back to the sink, that, in turn, may issue action commands for actors. The *automated architecture* allows sensors to send their readings to the actors which, in turn, can trigger appropriate actions. It is assumed that sensor nodes are static and aware of their own approximate location through localization algorithms [83, 84]. Note that such an assumption is not specific to our proposal, as it appears in other WSAN works [81, 82]. For example, actor nodes can cooperate with the sink in order to help locate a node, if need be. Sensor nodes can reach actors and sink through single or multi-hop communication. Actors can move freely and are assumed to have the same transmission range as sensors, although they have greater computation resources. It is assumed that every node is associated with a serial number as soon as it is manufactured.

4.4.1 The Actors' Overlay Network (AON)

This section presents the design of the AON components. To this end, as mentioned above, the well-known communication paradigm called Publish-Subscribe is considered [80]. The PS paradigm consists of a collection of clients that interacts by publishing messages and subscribing to classes of messages that perk their interest. Interested parties specify the events in which they are interested by using the *subscribe* function

that lists their topics of interest. Objects of interest publish events notifications via the *publish* function. Publishers do not keep subscribers' references and vice versa, as an entity, called ''the broker'', coordinates interactions, making sure that matching events are delivered to the relevant parties. PS systems are classified according to how their data description styles are expressed: either *topic-based* or *content-based* systems [80]. Content-based PS systems are more expressive, although they often require a complex implementation due to the nature of the filters. However, AON adopts on *topic-based systems,* a relatively simple class of PS system. In topic-based PS systems, users subscribe to *topics* in order to be notified of matching events. The subscription syntax is limited to a simple test on a specific field of the event.

4.4.1.1 Topic-based PS System for WSANs

The actor nodes which represent *subscribers* transmit their subscription messages "Subscribe" to the sink node. Such messages are composed of identifiers that specify the actor's selected topics of interest. Subscriptions are registered with the sink, which computes the topic segments by gathering actors that share the same interests. Sensors, which represent the event sources, generate data in response to variable changes that they monitor in the real world, such as temperature changes, voice degradation, etc, thus representing the publishers. ''Publish'' messages include the sensor address, the topic identifier, the event data, as well as a sequence number. The event *broker,* an agent located on each sensor node, acts as an interface between the event publisher and the subscriber agent for the actor nodes. More specifically, the objective consists of clustering actors which share common interests. Given this perspective, a mathematical formulation of the interest clustering problem is presented. The following section defines certain notations. Let S_A denotes the set of actors ($n_a = |S_A|$) and T the set of topics. Each actor $a \in S_A$ subscribes to n_s topics, represented by a binary vector $[x_{a,1}, x_{a,2}, \cdots x_{a,n_s}]$ where:

$$x_{a,t} = \begin{cases} 1, & if\ the\ actor\ a\ subscribes\ to\ the\ topic\ t \\ 0, & otherwise \end{cases}$$

Let subscriptions for two different actors a and b, represented by the vectors $[x_{a,1}, x_{a,2}, \cdots x_{a,n_s}]$ and $[y_{b,1}, y_{b,2}, \cdots y_{b,n_s}]$, respectively. The similarity measure between them can be defined as the total matches of corresponding topics for both vectors. The matching similarity measure between both actors' subscriptions is defined as follows:

$$d(a,b) = \sum_{t=1}^{n_s} \delta(x_{a,t}, y_{b,t}) \qquad (4.1)$$

where

$$\delta(x_{a,t}, y_{b,t}) = \begin{cases} 1, & x_{a,t} = y_{b,t} \\ 0, & x_{a,t} \neq y_{b,t} \end{cases}, \qquad 1 \leq t \leq n_s \qquad (4.2)$$

The lower the number of mismatches, the most similar both objects. Let subset $Z \subseteq S_A$ of actors represent the clusters centers, and a subset $D \subseteq S_A$ of actors must be assigned to such centers. Furthermore, if actor nodes are clustered according to their interests, one cluster can happen to be much bigger than the others, raising the costs of event delivery among cluster members. To solve this issue and favor clusters of similar sizes, clustering entropy is taken into account. Indeed, entropy quantifies the amount of uncertainty present in the state of a variable or, in this application, uncertain topic clustering. The entropy of Z is defined as $H(Z) = -\sum_{i \in Z} p_i log(p_i)$, where p_i denotes the percentage of nodes in the cluster represented by its center z_i. Let n_c denote the number of clusters ($n_c = |Z|$). Entropy reaches its maximal value $log(n_c)$ when all clusters are equally probable, meaning of identical size. The relative entropy function [85] is defined as follows:

$$H_r(Z) = -\frac{\sum_{i \in Z} p_i log(p_i)}{log(n_c)}, \qquad 0 < p_i \leq 1 \qquad (4.3)$$

and has values between [0, 1]. When the value of this function is close to 1, cluster sizes are similar to one another. In fact, the objective of the problem on hand, denoted as

Topic Clustering Problem (TCP), consists of finding the subset Z composed of n_c centroids and the connection vector W that maximizes: (a) the cost of logical connections across actors and (b) the relative entropy. The mathematical model is thus presented as follows:

$$Max\ F(W,Z) = H_r(Z) + \sum_{i\in Z}\sum_{j\in D} w_{ij}d(i,j) \qquad (4.4)$$

Subject to:

$w_{ij} \in \{0,1\},\ \ \forall i \in Z, \forall j \in D$ (4.4a)

$\sum_{i\in Z} w_{ij} = 1,\ \ \forall j \in D$ (4.4b)

$0 < \sum_{i\in Z} z_i \leq n_a$ (4.4c)

$W = [w_{ij}]$ depicts a $n_c - by - n_a$ binary matrix and $Z = [z_1, z_{2,} \cdots z_{n_c}]$ denotes the cluster centers' vector. Equation (4.4) represents the objective function of the model. Expressions (4.4a), (4.4b) and (4.4c) translate the model constraints: Constraint (4.4a) defines the scope of variables w_{ij}; Constraint (4.4b) ensures that each actor belongs to a single cluster; and Constraint (4.4c) specifies the bounds of the number of clusters.

4.4.1.2 Algorithms and Complexity

For large scale networks composed of numerous actors, such a clustering issue cannot be solved in polynomial time. Obviously, TCP belongs to the class of Facility Location Problems known to be NP-hard [65]. In the literature, *K-means* algorithm (KM) is usually considered to solve this type of problem, as is the case when mining data from large sets of data clustering [86]. The pseudo-code of the KM algorithm appears hereafter.

Table 4.1 K-means Algorithm

1. Choose an initial $\boldsymbol{Z^{(1)}}$ and maximize $\boldsymbol{F(W, \mathbf{Z}^{(1)})}$ to obtain $\mathbf{W}^{(1)}$. Set $\boldsymbol{t = 1}$.

2. Let $\widehat{\mathbf{W}} = \mathbf{W^t}$ and solve $\boldsymbol{F(\widehat{W}, \mathbf{Z})}$ to obtain $\mathbf{Z^{(t+1)}}$.

2.1. If $\boldsymbol{F(\widehat{W}, \mathbf{Z^t}) = F(\widehat{W}, \mathbf{Z^{t+1}})}$, output $\widehat{\mathbf{W}}, \mathbf{Z^t}$ and stop; otherwise goto 3.

3. Let $\hat{\mathbf{Z}} = \mathbf{Z^{t+1}}$ and solve $\boldsymbol{F(\mathbf{W}, \hat{\mathbf{Z}})}$ to obtain $\mathbf{Z^{(t+1)}}$.

3.1. If $\boldsymbol{F(\mathbf{W^t}, \hat{\mathbf{Z}}) = F(\mathbf{W^{t+1}}, \hat{\mathbf{Z}})}$, output $\mathbf{W^t}, \hat{\mathbf{Z}}$ and stop; otherwise let $\boldsymbol{t = t + 1}$ and goto 2.

Note that with the KM, although the number of clusters is predefined, it has the shortcoming of converging to local optima. To escape such local optima and find a more suitable solution, Tabu Search (TS) algorithm becomes an interesting alternative. The tabu search proposed by Glover [87] is a meta-heuristic that guides the solution search procedure to explore the solution space beyond local optimality. TS uses a local or neighborhood search procedure to iteratively move within S's neighborhood, from Solution S to Solution S1, until a certain stopping criterion is satisfied. As our main objective consists of building clusters composed of actors that share common interests, the higher the clustering costs, the better the algorithm. TS's ability to find a better solution compared to KM is the ultimate reason that motivated its selection when building the AON. The proposed TS algorithm, called *Tabu-TCP* for the problem on hand, adopts as stopping criterion a certain number of iteration, Δ, which is controlled by the network operator. The larger the number of iterations, the longer the search, and possibly the most cost efficient solution. Moreover, the designed Tabu-TCP algorithm uses a greedy algorithm to build the initial solution from which the search starts. The entire pseudo-code is illustrated in the Tabu-TCP algorithm below. Once clusters are computed, the sink broadcasts the results, composed of the members of each cluster and their associated topic vectors. As such, each actor becomes an *Access Point* (AP) for the event that occurs in its action area, which matches a topic that belongs to its cluster, while sensor nodes publish the matching data to the nearest AP.

Cluster computations make it possible to deliver events to actors according to the type of detected event. The issue to be addressed consists of setting the communication infrastructures used to deliver notifications among actors, while ensuring interactions remain possible between all nodes involved: sensors, actors, and sinks. The parts of this infrastructure, namely the distributed address assignment and the routing strategy, are detailed in the next sections.

Table 4.2 Tabu-TCP Algorithm

1. $(\mathbf{W}, \mathbf{Z})$, the current solution;
2. $(\mathbf{W}^*, \mathbf{Z}^*)$ the best-known solution and $\boldsymbol{F}(\mathbf{W}^*, \boldsymbol{Z}^*)$ its value;
3. $\mathbf{N}(\mathbf{Z})$ the neighborhood of $\boldsymbol{Z}$;
4. $\widetilde{\mathbf{N}}(\mathbf{Z})$, the admissible subset of $\boldsymbol{N}(\boldsymbol{Z})$ (i.e., non-tabu);
5. $\mathbf{T} = \emptyset$ The tabu list
6. Compute initial solution $(\mathbf{W^0}, \mathbf{Z^0})$ with a greedy algorithm. Set $\boldsymbol{Z} = \boldsymbol{Z^0}, \boldsymbol{Z}^* = \boldsymbol{Z^0}$ and $\mathbf{F}^*(\mathbf{W}^*, \mathbf{Z}^*) = \mathbf{F}(\mathbf{W^0}, \mathbf{Z^0})$;
7. While the termination criterion is not met, do

   7.1. Select $(\mathbf{W}, \mathbf{Z}) = \mathbf{argmax}_{\mathbf{Z}' \in \widetilde{\mathbf{N}}(\mathbf{Z})}[\mathbf{F}(\mathbf{W}, \mathbf{Z}')]$**;**

   7.2. If $\mathbf{F}(\mathbf{W}, \mathbf{Z}) > \mathbf{F}^*(\mathbf{W}^*, \mathbf{Z}^*)$**,** then set $\mathbf{F}^*(\mathbf{W}^*, \mathbf{Z}^*) = \mathbf{F}(\mathbf{W}, \mathbf{Z})$ and, $(\mathbf{W}^*, \mathbf{Z}^*) = (\mathbf{W}, \mathbf{Z})$;

   7.3. Record tabu for the current move in **T;**

   7.4. Update **T;**
8. End

4.4.2 Distributed Address Assignment

4.4.2.1 Basic Idea

In geometric design, IFSs are defined as subdivision methods that construct fractals by uniting several copies of themselves ad infinitum[74]. Each copy is transformed by a set of functions which stand for a finite number of *M* transformations, defined as follows:

$$\gamma_u(x, y) = (a_u x + b_u y + e_u, c_u x + d_u y + f_u) \quad 0 \le u \le M - 1 \qquad (4.5)$$

where $a_u, b_u, e_u, c_u, d_u, f_u \in \mathbb{R}$. For a given initial form A (e.g. square, triangle, etc.), the small affine copies $\gamma_1(A), \gamma_2(A) \cdots \gamma_M(A)$ are produced and turned into a new form. The mapping process computes recursively $A_k = \cup_{u=1}^{M} \gamma_u(A_{k-1})$, $\forall\, k > 1$. The sequence $\{A_k\}$ converges to a final set called the IFS attractor and also define a code space Γ composed of the affine transformations indexes u. A sequence of indexes defines the address of each part of the resulting attractor. An example of this kind of construction appears in Figure 4.2. This illustration depicts what is called a Sierpinski triangle. The set of IFS describing the fractal process of the Sierpinski triangle maps a base triangle into three children by splitting each edge of the base triangle in half. Three affine transformations

are applied repeatedly with each new triangle obtained, until the size of the triangle falls below the desired resolution. Thus, at each step, smaller triangles are created by joining the midpoints of the triangles created in the previous iteration. Figure 4.2 shows the results of applying this process to a base triangle five times. A detailed formalization of IFS and fractals is presented in [74].

Figure 4.2 A Sierpinki triangle [74]

4.4.2.2 Address Allocation

To apply the IFS to a WSAN area, four linear affine transformations are defined to represent the mapping functions as follows:

$$\begin{cases} \gamma_{00}(x,y) = \left(\frac{x}{2}, \frac{y}{2}\right) \\ \gamma_{01}(x,y) = \left(\frac{x}{2}, \frac{y}{2} + \frac{L}{2}\right) \\ \gamma_{10}(x,y) = \left(\frac{x}{2} + \frac{L}{2}, \frac{y}{2} + \frac{L}{2}\right) \\ \gamma_{11}(x,y) = \left(\frac{x}{2} + \frac{L}{2}, \frac{y}{2}\right) \end{cases} \quad (4.6)$$

The initial set $A_0 = \{(0,0),(L,0),(L,L),(0,L)\}$ represents the WSANs area and these affine transformations denote an area contraction by a factor of 1/2, followed by a translation. The indexes of these transformations define a code space $\Gamma = \{00, 01, 10, 11\}$. A square cell belonging to the resulting form is represented by a sequence of code $\phi = \phi_1\phi_2\phi_3 \cdots (\phi_u \epsilon\, \Gamma)$. The address allocation essentially follows a three-step process: (a) cell computing, (b) neighborhood discovery, and (c) identifier assignment.

a. Cell Computing

By network wide flooding, the sink starts the allocation process by broadcasting the set A_0. Each node recursively computes the cell to which it belongs, by applying the defined transformations and comparing its coordinates with the bounds of obtained sub-square. Since sensor nodes cannot determine their cell addresses using an infinite alphabet

sequence of the code space Γ, the value of $k_{order,}$ which defines the number of IFS iterations and the cell's address length, are chosen finite and bounded by taking into account the sensors' transmission range *r*. Indeed, to allow node communication within a cell, the worst case scenario is considered: in a cell, the distance between two nodes equals the cell's diagonal, so that the cell's side l_{cell} is bounded as $l_{cell} \leq \frac{r}{\sqrt{2}}$. Since $l_{cell} = L/2^{k_{order}}$, the number of sequences k_{order} in the cell address can be chosen so that:

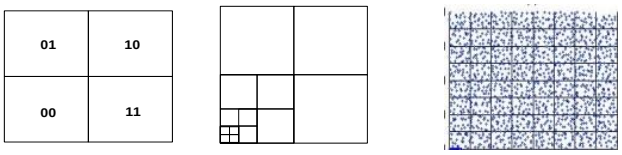
$$k_{order} \geq \frac{1}{2} + log_2\left(\frac{L}{r}\right) \quad (4.7)$$

The cell's address thus consists of the first part of the node's address. An address ϕ_{cell} of any cell consists of a finite sequence of elements of the code space Γ and is defined as follows:

$$\phi_{cell} = \phi_1\phi_2\phi_3 \cdots \phi_{k_{order}}, \qquad \phi_u \epsilon\, \Gamma \quad (4.8)$$

Figure 4.3 illustrates the construction process for r = 40 m and L = 200 m. Sub-squares represent the cells, and points deployed sensor nodes in the resulting attractor (Figure 4.3-C). With such physical parameters, the recursion order amounts to k_{order}=3 and the cell's address length equals 6 bits. Figure 4.3-A shows a one-order attractor while Figure 4.3-B provides an overview of the same attractor at a four-order level in Sub-square "00".

With this mechanism, nodes cell's addresses become $\phi_{\text{cell}} = 000001, 100011, \cdots 111111.$



A B C

Figure 4.3. (A) One-order attractor, (B) Four-order cells of attractor,
(C) Nodes spanned in the attractor cells

b. Neighborhood Discovery

When it finishes computing its cell address, the node initiates a neighborhood discovery process by including its serial number and the computed cell's address in a "Hello"

messages. The cost inherent to this neighborhood discovery approach is not specific to our technique, since all solutions which allow node interactions must pay neighbor discovery costs. Nodes that share a same cell address are thus aware of one another.

c. Identifier Assignments

Within each cell, the node with the lower serial number becomes the AC. The AC is responsible to maintain the global addressing states in the cell: it assigns an identifier for each sensor node that belongs to its cell. This is performed with the simple broadcast of a control message that specifies the cell's address and the list of both <serial number, assigned ID>. By doing so, each node that belongs to the cell obtains an identifier, rendering the node's complete address as:

$$\phi: \phi_{id}\ , \quad 0 \leq \phi_{id} < n_{cell} \tag{4.9}$$

where n_{cell} denotes the number of nodes in the cell of address ϕ and ϕ_{id} the node's identifier in that cell. Note that the AC assigns itself $\phi_{id} = 0$. Periodic refresh messages issued by the AC trigger updates of address pools by removing the IDs of unavailable nodes. The pseudo-code of the IFS algorithm appears below.

Table 4.3 IFS Algorithm

1. Inputs:
2. $\boldsymbol{A_0} = \{(\boldsymbol{x}, \boldsymbol{y}), (\boldsymbol{x}+\boldsymbol{d}, \boldsymbol{y}), (\boldsymbol{x}+\boldsymbol{d}, \boldsymbol{y}+\boldsymbol{d}), (\boldsymbol{x}, \boldsymbol{y}+\boldsymbol{d})\}$: Deployment corner points;
3. $\boldsymbol{p_0} = (\boldsymbol{x_0}, \boldsymbol{y_0})$: node coordinates.
4. $\boldsymbol{k_{order}}$: order of mapping; M: number of affine transformations
5. Output: ϕ_{cell}
6. Begin
7. For i=0 to $\boldsymbol{k_{order}}$ do
  a. For $\boldsymbol{u} = \boldsymbol{0}$ to M do
    i. For each point $\boldsymbol{p} \in \boldsymbol{A_{i-1}}$ do $\boldsymbol{A_i} = \boldsymbol{A_i} \cup \{\boldsymbol{\gamma_u}(\boldsymbol{p})\}$; End For p
    ii. $\phi_i = \phi_i + "\boldsymbol{u}"$; $\boldsymbol{Sq} = \boldsymbol{Sq} \cup \{(\boldsymbol{A_i}, \phi_i)\}$
  b. End For $\boldsymbol{u}$
8. End For $\boldsymbol{i}$
9. For each Square $\boldsymbol{s} \in \boldsymbol{S}_q$ do
  a. If $\boldsymbol{p_0} \in \boldsymbol{S_q}$ then return ϕ_q;
10. End For $\boldsymbol{s}$
11. End

The above described address allocation guarantees unique addresses since the discretization of the network area allows the ACs to act in a disjoint manner. ACs control the identifiers in their respective cell by maintaining the cell's global state, ensuring that a node address is not assigned twice in the network. The following section provides the design rationale that motivated using an Iterated Function System to allocate node addresses.

4.4.2.3 Design Rationale

Three keys features motivated the use of the IFS discretization fractal theory to allocate node addresses in WSANs.

a. Location Information: the feasibility of determining location by means of a localization algorithm is enhanced by the presence of actor nodes in the network. Indeed, actors are resource rich nodes and can easily be involved in localization algorithm such as those proposed for ad hoc networks in [83, 84]. However, the delivery of detected events can be affected significantly by the nodes' imprecise location information. The usage of IFS offers the possibility of using approximate or relative location information to determine nodes' cells. Accordingly, it is more robust to errors and imprecisions when measuring and estimating locations than schemes that rely on exact location information. Indeed, with imprecise location information, when location errors are unimportant, the cell computed with an IF can be the same as the one computed with the exact location information. This fact becomes salient upon viewing simulation results, shown in Section 4.5.
b. Allocation Costs: each node can determine its cell without interacting with its neighbors. The cell identifier assignment is performed with a single broadcast advertisement, thus reducing allocation costs in terms of signaling overhead. The costs of $O(m_{cell})$ is lower than those of DHCP-Like or hierarchical allocations, as shown with a comparative table provided in Section 4.5.
c. Routing: one of the targeted objectives consists of ensuring that the sensor nodes play the same role when routing data to an actor or the sink, to avoid bottlenecks or a

single point of failure. The shortage of hierarchical structure is that the task of the cluster head is heavy as it is responsible to manage and coordinate within the cluster nodes and may create network bottlenecks. For that purpose, the usage of IFS provides each node with the ability of sending data to any destination throughout its neighborhood, without referring to a dedicated node. This is accomplished by using the location information embedded in the cell's address.

Now that features have been made explicit, the following section details how the allocated addresses serve the Routing Manager purposes through the *Fractal Cell Routing* protocol.

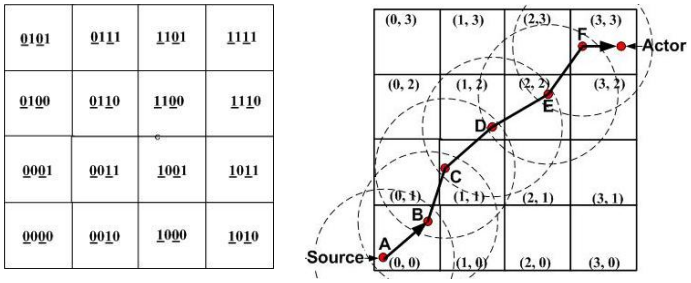
4.4.3 Routing

The proposed addressing system takes into account the two kinds of WSANs architectures described in [2]: the *Automated Architecture* and the *Semi-Automated Architecture*. Indeed, when a phenomenon is detected with the automated architecture, sensors transmit their readings to the actor nodes which then process all incoming data and initiate appropriate actions. A semi-automated architecture is used when data are routed back to the sink, which then issue action commands to actors. Such kind of communication architecture between sensors and actors is well-described in [70, 71], where sensor-actor communication is performed through one or multiple hops. For these two kinds of architecture, this section presents the *Fractal Cell Routing* (FCR) strategy that uses the allocated addresses to carry data packets from a source to its destination.

4.4.3.1 Fractal Cell Routing

When a node must send data to a specific destination, it employs cell position information that is embedded in the cell's address ϕ to compute the inter-cell distance between source and destination. Let $\underline{c}_0 c_1 \underline{c}_2 c_3 \underline{c}_4 c_5 :: c$ denote the address assigned to a given node. Its cell coordinates are given by $\left(\left(\underline{c}_0 \underline{c}_2 \underline{c}_4\right)_2, \left(c_1 c_3 c_5\right)_2\right)$. As such, from the addresses of any pair of cells u and v, the inter-cell distance d_{uv} is equivalent to the Euclidian distance between their coordinates. From the neighborhood discovery process,

each node acquires its neighbors' cell addresses and can compute the inter-cell distance between themselves and the destination. Note that this neighborhood, called *routeLet,* is composed of nodes in adjacent cells as well as nodes within the same cell. Upon receiving a packet to be forwarded, the node selects, among the forwarding candidates set, the nodes in its routeLet with the minimum inter-cell distance with the destination. Then, from that set, a node is selected randomly as forwarder. A node can have neighbors only in its own cell or have the same inter-cell distance with a neighbor in an adjacent cell. To break the tie, in a first instance, a node is selected randomly and, the second time around, a node is selected in the most densely populated cell in the routeLet. Figure 4.4 illustrates the FCR strategy between source and destination. Indeed, the source node S with address $\phi_{\text{cell}}^{\text{src}} :: \phi_{\text{id}}^{\text{src}} = \underline{0}\underline{0}\underline{0}0 :: 1$ must send a packet to the destination node (an actor for example) whose address is $\phi_{\text{cell}}^{\text{dest}} :: \phi_{\text{id}}^{\text{dest}} = \underline{1}\underline{1}\underline{1}1 :: 2$. Their respective cells' coordinates are $(\underline{0}\,\underline{0}, 00)$ and $(\underline{1}\,\underline{1}, 11)$, equivalent in decimal base to $(1, 3)$ and $(3, 3)$. A possible path is composed of nodes selected in the cells, as follows: $\underline{0}\underline{0}\underline{0}0 :: 1 \rightarrow (0,0) \rightarrow (1,1) \rightarrow (1,2) \rightarrow (2,2) \rightarrow (3,3) \rightarrow \underline{1}\underline{1}\underline{1}1 :: 2$.



A. Cell map B. Data forwarding

Figure 4.4 Fractal cell routing strategy

4.4.3.2 Sensor-Actor Communication

Actor nodes move freely from one point to another in order to perform a given task, following a path on which many acting points are found. At each acting point, it collects data from sensors and ensures coordination with other actors according to such data.

Actor coordination and task assignment mechanisms are beyond the scope of this paper. However, a coordination mechanism can be incorporated as an AON functionality to select the best suited actor for the actuation task. Every time an actor moves to a new area, it requests a new address from the neighborhood ACs (if the movement is directed towards a new cell). It selects the address allocated by the closest AC and announces itself by specifying its new address and interests in a broadcast announcement beacon. The announcement is finally described by the tuple $< \phi_{actor} : \phi_{id}, [t_1, t_2, \cdots t_r] >$. Sensor nodes that belong to a covered area create and configure a network interface according to the topics to which advertising actors have subscribed to. This configuration includes the actor's address and interests. When an event occurs, the node uses FCR to deliver the sensed data to the closest actor, even if the detected event fails to match any of the actor's interest subscription. Sometimes, when the source node belongs to an overlapping area, the node reports data according to the actors' interests subscription.

4.4.3.3 Actor-Actor Communication

When an actor node receives data reported after an event is detected, either of two situations occur: (1) the event reported does not match its interest: in this case, the actor node selects a partner within its own cluster that subscribes to such interest; otherwise, the event is reported to the sink using the FCR strategy; (2) the reported event matches the actor's subscription: in this case, the associated data is disseminated by using a minimum Spanning Tree (mST), rooted on the Access Point Actor (APA). Indeed, the APA node is the first actor contacted by the sensor node that generates the event. The metric used to compute this mST is the inter-cell distance between adjacent actor nodes. Note that the actor holds a list of tuples $< \phi_{actor} : \phi_{id}, [t_1, t_2, \cdots t_r] >$ of its partner in the same cluster. From one actor to another in the mST, FCR is used to carry data. Figure 4.5 illustrates both communication cases.

4.4.3.4 Sensor/Actor to Sink Communication

Communication with the sink takes place in either of these cases: (a) the event does not match any interest in the cluster of the APA, which subsequently reports the event to the sink using FCR; (b) for the semi-automated architecture suggested in [70], sensors report event data to the sink using FCR. The same communication pattern is used by the sink to issue commands for actors.

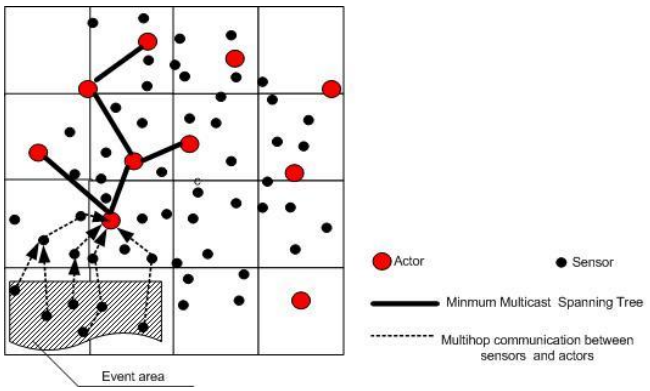


Figure 4.5 Sensor-actor communication

4.4.4 Handling Topology Changes

Topology changes in WSANs occur when nodes become unreachable, due to battery depletion, or when additional nodes are deployed during network operations. Another important topology change in WSANs pertains to actors' mobility. Although sensor nodes are considered static most of the time, actors are destined to navigate throughout the network in order to fulfill their coverage mission. Both of these specific cases are thus considered in this study.

4.4.4.1 Joining Nodes

When a node, or a group of nodes, becomes unavailable in a given cell, the AC updates the address pool accordingly. A newly arrived node in a given cell triggers a three-step process to get an address: (1) it waits for an AC notification message, (2) upon receiving the AC notification message, the node issues an address request to the closest AC in

terms of Radio Signal Strength Indicator (RSSI), and (3) the AC responds by allocating its cell address and an identifier. A newly arrived node senses the AC notification before sending a request, since blind solicitation often leads to multiple responses, which are energy inefficient. In addition, a moving actor that obtains an address immediately notifies its cluster partners so that they can update the relevant profiles.

4.4.4.2 Departing Nodes

The addresses assigned to nodes that become unreachable, due to a migration or eventual deaths, are recovered by the AC when its notification fails to generate a response from these nodes. AC notifications take place periodically. Two departure scenarios are considered: departures of regular nodes or ACs. A regular node departure is detected once the AC notification does not generate a node acknowledgement. Indeed, failures to respond cause departure detection, and the ID of the departed node is set in the pool for future reuse. The AC departure is detected when its cell nodes fail to receive AC notifications. Once an AC departure is detected, a re-configuration phase is reinitiated among the nodes that share a common cell address. The pool of the newly elected AC is updated according to the number of nodes that share the same cell address.

4.5 Performance Evaluation

This section reports on the performance evaluation of SubCast through extensive simulations using the *Qualnet* simulator [67]. Prior to presenting simulation results and assessing the accuracy of SubCast, experimental parameters are listed in Table 4.4. Note that Melodia et al. [70] have used the same parameter settings for their communication architecture in WSANs. When conducting these experiments, key metrics depicting the behavior of the proposed scheme are measured. The results presented reflect an average of 15 simulations over different random topologies with a confidence interval of 95%.

Table 4.4 Simulation Parameters

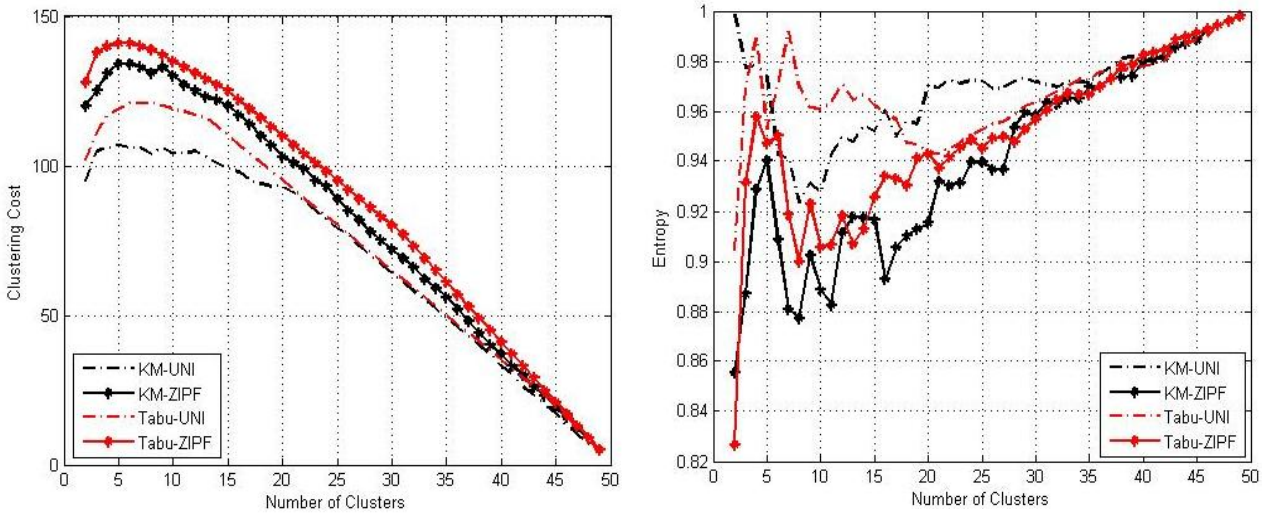
Parameter	Value
Network area	200 m x 200 m
Sensors	200
Transmission range	40 m
bandwidth	250 kbps
Packet size	128 bytes
Simulation time	300 sec
Actors	variable
Number of topics	20
Number of subscriptions	8
Reporting rate	1 packet/sec

To evaluate the clustering efficiency of the AON, its building cost is measured using the KM algorithm and compared to the proposed Tabu-TCP algorithm. KM and the Tabu-TCP algorithms are both implemented on the sink. The termination criterion for the Tabu-TCP is set at $\Delta = 50$ iterations. The workload is constructed as follows: given the set of topics *T*, each topic $t \in T$ is associated with probability q_t, $\sum_{t \in T} q_t = 1$, so that each node subscribes to t with a probability q_t. The value of q_t is distributed according to a Zipf distribution[1] or a uniform distribution. The Zipf factor is set at θ = 0.5 according to studies provided in [28], based on the subject popularity. Each actor subscribes to n_s = 8 of 20 available topics in *T*.

Figure 4.6 (a) depicts the AON clustering cost for various numbers of clusters, using the Tabu-TCP and KM algorithms. The number of clusters varies from 2 to 48 and a very large AON of $n_a = 50$ actors is considered. Tabu-UNI and KM-UNI denote that actors subscribe to their interest in a random and uniform distribution, while Tabu-ZIPF and KM-ZIPF denote a Zipfian topic subscription. As illustrated, regardless of the subscription access, the TS algorithm achieves a better clustering cost than KM. This is due to the fact that KM algorithms terminate at local minimum and, therefore, possibly

[1] The Zipf distribution is a well-known high-skewed distribution. If $\theta \geq 0.9$, it is very highly skewed; when $\theta = 0$ the distribution is uniform.

not providing optimal solutions to clustering problems. TS outperforms KM since it has the ability to provide means to escape from local optimal solutions by exploring the solution space beyond local optimality and attempts to find global optimal solutions. More particularly, at the maximal clustering cost, the optimal number of clusters that composes the AON is reached. For the case on hand (50 actors), the maximum cost is reached for $n_c = 7$ for Tabu-ZIPF, $n_c = 9$ for KM-ZIPF, $n_c = 8$ for Tabu-UNI and $n_c = 6$ *or* 9 for KM-UNI. The Zipfian access costs are more appealing as actors share a greater number of common interests compared to the uniform access. When a tie occurs, for the same values of maximum clustering cost, the optimal number of clusters is selected as the one with the maximum relative entropy. That is the case for KM-UNI where an identical maximum cost is obtained for $n_c = 6$ and $n_c = 9$. The value of $n_c = 6$ should thus be selected as its relative entropy is greater (0.94), as illustrated in Figure 4.6(b). Indeed, this figure shows the relative entropy variations as the number of clusters increases. This value increases with the number of clusters as clusters sizes are similar.



(a) Clustering Cost (b) Variation of the Relative Entropy

Figure 4.6 Topic clustering accuracy

Using the above described methodology (clustering costs and relative entropy) to determine the optimal number of clusters in the AON, the number of topics was varied in order to measure its influence on the communication costs in the AON. For that

purpose, events are randomly generated from sensor nodes according to uniform distribution and Zipfian distribution over the number of topics. Note that for all experiments, the reporting rate equals 1 packet/sec, from 100 sensor nodes randomly selected in the network.

Figure 4.7 depicts the comparative results of AON architecture with the Simple Clustering architecture (SC) in terms of communication costs. In the SC architecture (similar to the automated architecture described in [2]), sensors report events to the nearest actor, and when the reported event fails to match the actor's subscriptions, the actor forwards the events to the sink, which has the ability to handle such situations. Figure 4.7 shows that communication costs and number of topics increase proportionally. This is explained by the increasing number of topics that the closest actor cannot handle in the SC case. In the AON case, growth is explained by the increasing number of events that fail to catch interest in clusters, before being forwarded to the sink. For a larger number of topics, communication costs increase as actors share too few common interests, leading to a large number of clusters, causing forwarded reported events, as with the SC architecture. Conversely, for the zipfian interest accesses, communication costs are always better in the AON architecture due to the large number of common interests shared by actors.

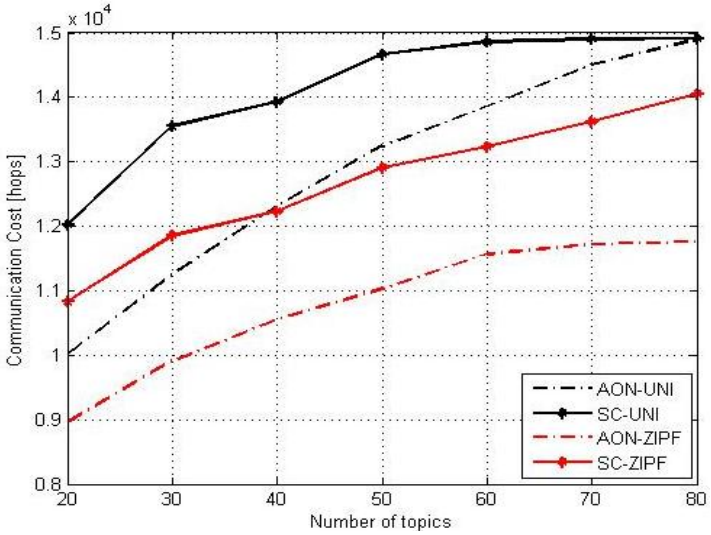


Figure 4.7 Communication Costs

Figure 4.8 illustrates the load distribution on sink and actors. It clearly shows that the sink load and the number of topics increase proportionally, regardless of the distribution of the actors' subscriptions. The event load on actor nodes is more significant when they share more common interests, as is the case in the zipfian distribution, especially with the proposed AON architecture. This highlights the ability of actors to handle most events occurring in the network, as well as the benefit of using the AON architecture.

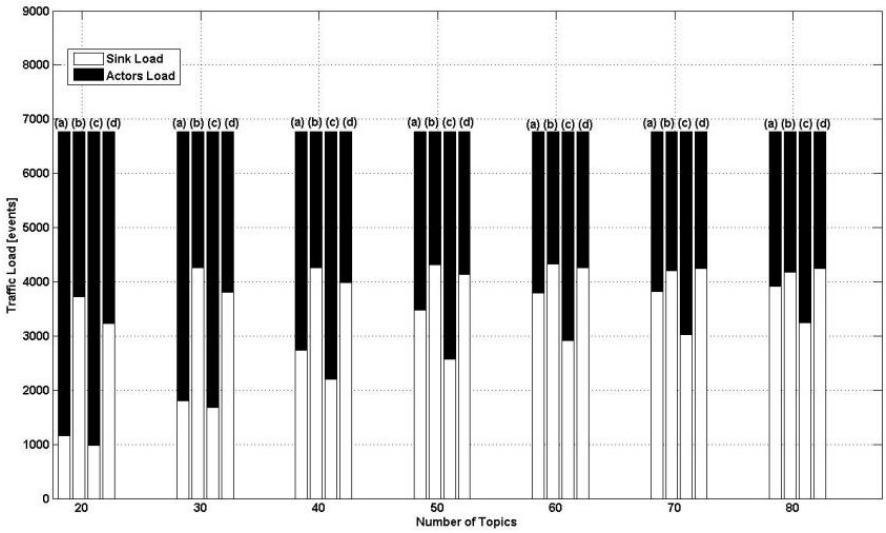


Figure 4.8 Events load distribution (a) AON-UNI, (b) SC-UNI, (c) AON-ZIPF and (d) SC-ZIPF

To evaluate the Fractal Cell Routing strategy (FCR) designed for SubCast, comparisons are made with the existing ZigBee routing protocols [43] for tree and mesh networks in terms of Packet Delivery Ratio (PDR) and End-to-end Delay. Figure 4.9 depicts the packet delivery ratio and the End-to-end Delay with an increasing number of reporting nodes randomly selected in the network. As expected, it is clear that the FCR strategy outperforms the ZigBee routing protocols, since FCR is based on sensors cell's coordinates and its forwarding strategy is quite similar to a classical geographic forwarding routing. More particularly, prior to forwarding data packet with the FCR strategy, the node does not refer to any cluster head as is the case with the ZigBee

routing protocols. With a growing number of reporting nodes, the PDR decreases abruptly with the ZigBee routing protocols, which is not the case with FCR as paths used to send packets are more disjoints compared to those in ZigBee routing protocols, which almost cross one another. Given such characteristics, it is obvious that end-to-end delays follow the same trend.

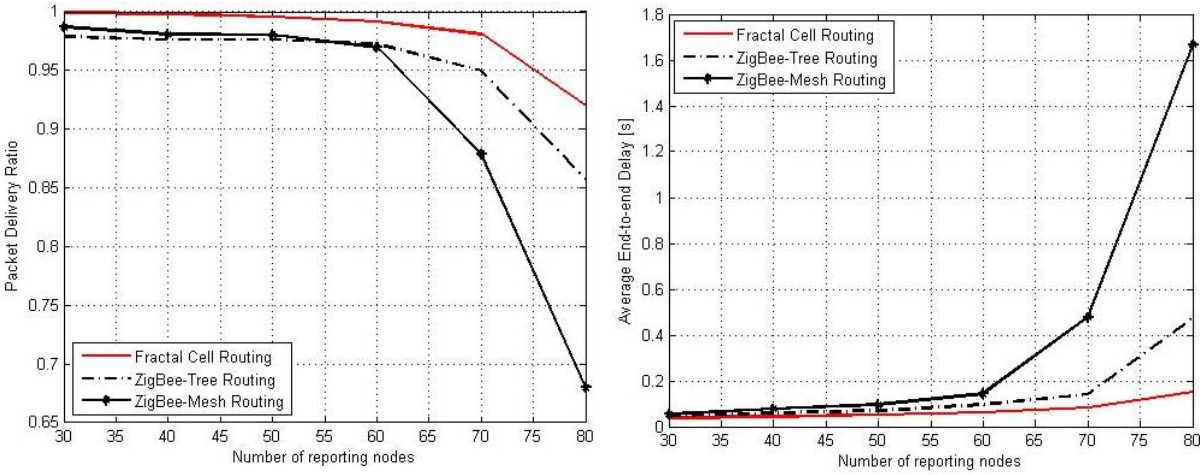


Figure 4.9 FCR versus ZigBee Routing protocols

Since the SubCast architecture is based on the assumption that nodes are aware of their own location through a localization algorithm, the impact of localization errors on routing performance is evaluated. For that purpose, comparisons are made with the Geographic Forwarding strategy (GF) [88] which forward data based on the knowledge of nodes coordinates. To highlight the pertinence of the results, a sink is deployed at the upper right corner of the area, no actors are deployed in this scenario and all detected events are reported to the sink. Figure 4.10 shows the impact of the location error on the PDR and the end-to-end delay. Indeed, the location error is generated from a Rayleigh distribution, by generating a pair $(\Delta x_i, \Delta y_i)$ where both Δx_i and Δy_i are selected from a zero-mean Gaussian distribution $N(0, \sigma_{loc}^2)$. With the average location error $\mu = \sigma_{loc}\sqrt{\pi/2}$, one can generate a location error distribution by fixing σ_{loc}. When μ varies from 0 up to 30%, the FCR routing strategy is less sensitive to location errors compared to the GF strategy, which uses node coordinates to deliver data to the sink. When

location errors increase, the PDR with GF decays significantly compared to the FCR strategy. This leads to higher end-to-end delay as depicted in Figure 4.10(b). The delay increases as the location error induces the forwarding strategy to choose the wrong nodes and then, increases the number of hops to be crossed in order to reach the sink node. The performance of FCR over GF can be explained by the fact that the same cells are always selected as forwarding cells to deliver data to the sink, and despite the location error, almost nodes belong to the same cell.

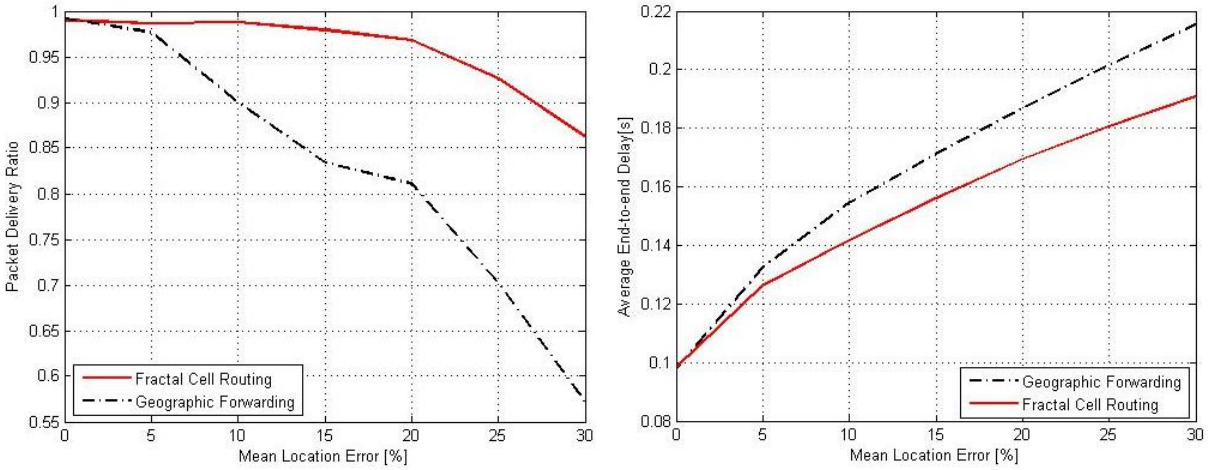


Figure 4.10 Impacts of localization errors on the routing strategy

To evaluate the impact of actor node mobility, various numbers of actors are deployed, navigating in the network following a random waypoint mobility model. Their speed is set in the range $[0, 2m/s]$ with a 30-second break. As previously, AON communication costs are measured and compared to:

(a) The Simple Clustering (SC), described above, in which sensors are clustered with their closest actor. Events reported to the closest actor with mismatched subscription are delivered to the sink node using FCR;

(b) The Simple Infrastructure less Clustering architecture (SIC), in which the event reported to the closest actors with a mismatched subscription is broadcasted among the actor nodes in order to find the one that does match the reported event. Note that actors do not broadcast events when subscriptions match.

Figure 4.11 compares results pertaining to the communication costs of such configurations. Obviously, the SIC scheme is only efficient for a small number of deployed actors while it turns out to be more costly with a greater number of actors. Communication costs remain constant with the SC scheme while they decay with the AON infrastructure, which is due to the following two reasons: (1) the clustering increases the likelihood of find matching actors, and (2) actors' mobility favors the proximity of sensor nodes and actors belonging to clusters with matching events. Moreover, with a growing number of actors, additional clusters help keep dissemination costs low compared to SC and SIC. This would not be the case if the number of clusters was fixed, as is the case for the KM algorithm.

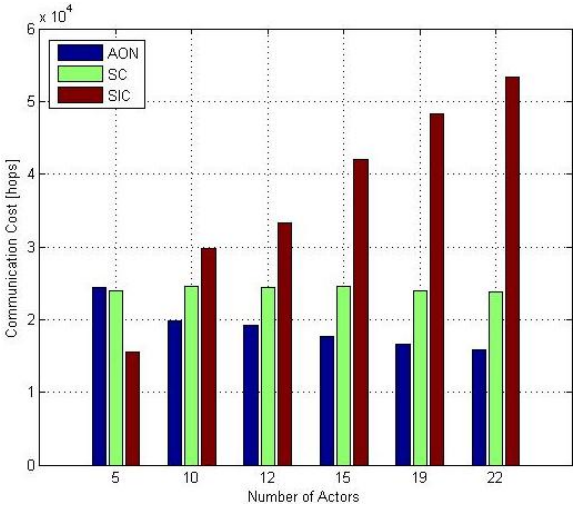


Figure 4.11 Communication costs for different event delivery infrastructure with moving actors

Table 4.2 provides a qualitative comparison of the proposed addressing system with some related works reported in the literature. In a cluster-based addressing scheme, the address length is expressed as $C_{cb} = log_2(n_a) + log_2\left(\frac{n}{n_a}\right)$, assuming that clusters headed by actors have an identical number of sensors. The closed form of the address length is then $C_{cb} = log_2(n)$, which is equivalent to the address length in the network-wide addressing scheme [38], where the address length also equals $log_2(n)$. The address

length in SubCast system is expressed as $C_{SubCast} = 2k_{order} + log_2\left(\frac{n}{2^{2k_{order}}}\right)$ and its closed form is $C_{SubCast} = log_2(n)$. Thus, with the simulation parameters, the address length in SubCast equals 8 bits. Assume that there are m types of message to allocate addresses to the network nodes. To complete the allocation process, m_{cell} messages are needed in SubCast and $m \cdot n$ for the other cases.

Table 4.5 Qualitative Analysis of Address Allocation Schemes

Properties	SubCast	TreeCast	ZigBee	DHCP-Like
Uniqueness	Yes	Yes	Yes	Yes
Allocation overhead	$O(m_{cell})$	$O(mn)$	$O(mn)$	$O(mn)$
Allocation mode	Distributed	Centr./Distributed	Centr./Distributed	Centralized
Address length [bits]	$log_2(n)$	$\geq log_2(n)$	16	$log_2(n)$
Routing	Flat	Tree	Tree and Mesh	Flat

4.6 Conclusion

This paper presents an addressing and routing system for large scale WSANs based on topic clustering among actors, and fractal area discretization to allocate addresses. To this end, actors are clustered according to their interest. The overall clustering system forms the Actors Overlay Network that uses a multicast tree communication pattern to deliver reported event to interested actors. Such a process of assigning node addresses in the networks depends on the nodes' physical location, which may be prone to localization errors. The fractal theory Iterated Function System appears to deal efficiently with such location errors and provides a low-cost distributed allocation that guarantees that unique addresses are assigned to network nodes. The proposed architecture was tested with simulations and the results were compared with certain traditional schemes found in the literature. Results demonstrate SubCast robustness and, in fact, contribute to making it an attractive addressing system for WSANs. Future research avenues include plans to grant the task allocation process to actor nodes according to the reported events, a constituent of the actors' overlay networks.

CHAPITRE 5

QoSNET: AN INTEGRATED QoS NETWORK FOR ROUTING PROTOCOLS IN LARGE SCALE WIRELESS SENSOR NETWORKS

Therence Houngbadji*, Samuel Pierre

Department of Computer Engineering, École Polytechnique de Montréal

E-mail: { Therence.Houngbadji, Samuel.Pierre }@polymtl.ca

P.O. Box 6079, Centre-Ville Station, Montreal, Quebec, H3C 3A7, Canada

Abstract

Numerous QoS routing strategies focus on end-to-end delays to provide time constrained routing protocols in Wireless Sensor Networks (WSNs). With the arrival of wireless multimedia sensor networks, traffic can be composed of time sensitive packets and reliability-demanding packets. In such situations, some works also take into account link reliability to provide probabilistic QoS. The trade-off between the guarantee of the QoS requirements and the network lifetime remains an open issue, especially in large scale WSNs. This paper proposes a promising multipath QoS routing protocol based on a separation of the nodes into two sub networks: the first part includes specific nodes that are occasionally involved in routing decisions, while the remaining nodes in the second sub-network fully take part in them. The QoS routing is formulated as an optimization problem that aims to extend the network lifetime subject to QoS constraints. Using the percolation theory, a routing algorithm is designed to solve the problem on the respective sub-networks. Simulation results show the efficiency of this novel approach in terms of average end-to-end delays, on-time packet delivery ratios and network lifetime.

*Corresponding author. Phone:+1 514 340 4711 x 2127

E-mails: therence.houngbadji@polymtl.ca (T. Houngbadji), Samuel.Pierre@polymtl.ca (S. Pierre).

Index Terms—Sensor networks, Iterated Function System, Routing, Quality of Service, Percolation.

5.1 Introduction

Wireless Sensor Networks (WSNs) consist of an emergent technology deployed for a large range of solutions, spanning military, civilian, environmental and commercial applications. They consist of a large number of low-cost sensing devices equipped with wireless communication and computation capabilities. Given the recent advances in WSNs, it is expected that video sensors will be supported in such networks, for applications such as battlefield intelligence, security monitoring, emergency response, and environmental tracking. For example, multimedia sensors may monitor the flow of vehicular traffic on highways and retrieve aggregate information such as average speed and number of cars. Given the physically small nature of sensors, and since multimedia applications typically produce a huge volume of data that require high transmission rates and extensive processing, power consumption becomes a fundamental concern in WSNs. More particularly, transmitting a video stream using the shortest path will drain the node of its energy along the path, thereby shortening the network lifetime, for example. Most proposals for routing in WSNs are based on a flat, homogenous architecture in which every sensor has identical physical capabilities and can interact only with its neighboring sensors. However, flat topologies are not always best suited to handle the amount of traffic generated by multimedia applications, including audio and video [23]. Therefore, developing new routing strategies to maximize the network lifetime, while satisfying the QoS requirements, represent a critical problem to be addressed.

Some works attempt to address the issue through single path routing strategies which only guarantee delay constraints. Compared to multipath routing, single path routing algorithms in WSNs are simpler and they consume less energy. Moreover, aside from meeting delay constraints, multipath routing is also reliable. Reliability can be

characterized by way of a packet delivery ratio, which is defined as the ratio between the number of unique packets successfully received by the sink over the number of packets generated by source nodes. In WSNs, multipath routing is used to establish multiple paths between all source–sink pairs in order to increase the likelihood of reliable data delivery. Multiple copies of data along different paths [89] are sent to reduce data delivery delays by sharing transmission delays among the different paths available from the source to the destination.

Thus, end-to-end delays, reliability requirements as well as the residual energy of nodes in WSNs are considered to build a new multipath routing strategy over a discretized network area. This perspective considers how underlying communication networks can achieve hard QoS constraints while efficiently utilizing network resources. This work complements our previous work [90] on addressing systems in large scale WSANs. It extends routing capabilities by providing QoS for large scale WSNs that possibly include powerful devices such as actors or mobile robots. In a first step, the network area is discretized into cells using the fractal theory Iterated Function System (IFS). In each cell, a Cell Controller (CC) is elected to control the signaling operation (address allocation, statistics collection, etc.). These CCs form the Cell Controller Network (CCN) which is also expected to integrate powerful devices such as actors and mobile robots. The remaining nodes compose the Simple Sensor Network (SSN) which maintains the primary data routing role of sensor nodes. The hard QoS constraints are fulfilled by using a switching routing mechanism between the two sub-networks in order to determine a forwarding set. The latter consists of the selected nodes where the incoming packet will be forwarded. This switching mechanism is accomplished by applying a *percolation* theory to the discretized network area. The remainder of this paper is organized as follows: Section 5.2 discusses related works and Section 5.3 provides an in-depth look at the proposed routing strategy. Performance evaluations are detailed in Section 5.4 followed by a Conclusion in Section 5.5.

5.2 Related Work

Routing techniques are divided into three categories based on their underlying network structures: flat, hierarchical, and location-based [6]. Furthermore, these protocols can be classified into multipath-based, query-based, negotiation-based, QoS-based, and coherent-based. This section presents an overview of the existing QoS-based routing protocols used in WSNs. QoS-based protocols allow sensor nodes to balance energy consumption and certain predetermined QoS metrics before they deliver data to the sink node. One of the first routing protocols for WSNs, Sequential Assignment Routing (SAR) [47], introduces the notion of QoS in routing decisions according to three factors: energy resources, QoS on each path, and the priority level of each packet. In order to avoid single route failures, a multipath approach is coupled with localized path restoration. SAR maintains multiple paths from nodes to a base station. Although, this ensures fault-tolerance and easy recovery, the protocol suffers from high overhead to maintain tables and states for all sensor nodes, especially in the presence of a large number of nodes. SPEED [91], another QoS routing protocol for WSNs, provides soft real-time end-to-end guarantees, yet it also maintains a desired delivery speed across sensor networks with a two-tier adaptation method included to divert traffic at the networking layer, and locally regulating packets sent to the MAC layer. MMSPEED [49], an extension of SPEED, supports service differentiation and probabilistic QoS guarantee. For delivery timeliness, multiple network-wide packet delivery speed options are provided for different types of traffic according to their end-to-end deadlines. As is the case with SPEED, since all MMSPEED mechanisms work locally without global network state information and end-to-end path setup, they become scalable and adaptive to network dynamics. Both SPEED and MMSPEED share a common deficiency: they fail to consider energy consumption metrics. Several delay-aware routing techniques have been proposed [50-53] to improve delays. A routing algorithm proposed by [54] uses the global knowledge of node queue sizes by the gateway in order to calculate end-to-end delays, lowest-cost paths and to generate routing tables. A heuristic solution to

find energy-efficient paths for delay-constrained data in WSNs has been designed in [55]. They employ topology control and a model of contention delays caused by the MAC layer. Paths between source and sink nodes are identified and indexed according to the amount of energy they consume. End-to-end delays are estimated along each of the ordered paths and the one with the lowest index that satisfies the delay constraint is selected. Multi-constrained multipath routing is proposed in [56, 57], where a node to sink packet delivery is achieved based on the so-called Soft-QoS constraints expressed in terms of reliability and delays. This model addresses the issue of multi-constrained QoS in WSNs by considering the unpredictability of network topology and attempting to minimize energy consumption. However, with these approaches, packets may get routed to a node that is highly congested and/or whose energy is critical. Simulation results conducted for a small network size do not guarantee the same performance for large scale WSNs in which the number of selected paths can increase and become a source of wasted energy. A Real-time Power-Aware Routing (RPAR) protocol [92] is proposed to achieve application specified communication delays at low energy cost, by dynamically adjusting transmission power and routing decisions. In [58], a protocol called ReInForM is proposed to deliver packets with the desired reliability level by sending multiple copies of each packet along multiple paths between the source and the sink. The number of paths used is dynamically determined depending on the channel error probabilities. Instead of using disjoint paths, GRAB [59] uses a path interleaving technique to achieve a high level of reliability. To date, all multipath routing techniques address the QoS routing in terms of the end-to-end delays and reliability in small WSNs, made up of homogeneous nodes whose capabilities are almost identical. Furthermore, it seems that there is a need to address the same issues in large scale WSNs.

5.3 The System Model

WSNs with randomly deployed nodes in a two-dimensional network area are being considered. Nodes are powered by battery and they are assumed to be aware of their own location. Indeed, nodes can find their own approximate location coordinates using

triangulation or multilateration methods. A grid network architecture is considered, as it discretizes the network into two-dimensional micro-scale areas that favor operation distributed over the network. Other forms such as triangles or hexagons could be used to discretize the network area; yet the grid representation is selected for its simplicity. Current network representations using grid structures [93] usually lead to an underutilization of the radio coverage areas, as pointed out by [94]. This is due to the fact that all nodes within a grid must be able to reach all nodes in adjacent grids. Actually, nodes are forced to cover less than half the distance allowed by the radio range. To deal with this issue, the fractal grid representation, using a square form, discretizes the network area by using the entire radio coverage of nodes. The proposed system model follows the same IFS design strategy used in [90] for WSAN addressing systems. IFSs are simple methods to construct fractals built by uniting several copies of themselves ad infinitum[74]. Each copy is transformed by a set of functions which stand for a finite number of *M* transformations $\{\gamma_j;\ 0\leq j\leq M\text{-}1\}$.

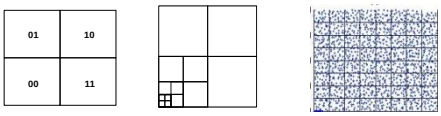
$$\gamma_j(x,y) = \left(a_j x + b_j y + e_j, c_j x + d_j y + f_j\right) \tag{5.1}$$

where $a_j, b_j, e_j, c_j, d_j, f_j \in \mathbb{R}$. For a given initial form S, the small affine copies $\gamma_1(S), \gamma_2(S) \cdots \gamma_M(S)$ are produced and lead to a new form $W(S) = \bigcup_{j=1}^{N} \gamma_j(S)$.The mapping process starts with a set S_0 and computes recursively $S_k = W(S_{k-1})\ , \forall\ k > 1$. The sequence $\{S_k\}$ converges to a final set called the IFS attractor. Node coordinates are used by the IFS to determine the cell where the node is located. In order to apply IFS to the network area, four linear affine transformations that represent the mapping functions are defined as follows:

$$\begin{cases} \gamma_{00}(x,y) = \left(\frac{x}{2}, \frac{y}{2}\right) \\ \gamma_{01}(x,y) = \left(\frac{x}{2}, \frac{y}{2} + \frac{a}{2}\right) \\ \gamma_{10}(x,y) = \left(\frac{x}{2} + \frac{a}{2}, \frac{y}{2} + \frac{a}{2}\right) \\ \gamma_{11}(x,y) = \left(\frac{x}{2} + \frac{a}{2}, \frac{y}{2}\right) \end{cases} \tag{5.2}$$

The initial set $S_0 = \{(0,0), (a,0), (a,a), (0,a)\}$ represents the network area and these

affine transformations represent a contraction of the area by a factor $s = 1/2$, followed by a translation. The indexes of these transformations define a code space $\Gamma = \{00, 01, 10, 11\}$. Indeed, $\phi = \phi_1\phi_2\phi_3\cdots$ denotes an element of the attractor ($\phi_i \epsilon\, \Gamma$). Figure 5.1(c) illustrates the resulting attractor. Sub-squares represent the cell and the dots show the deployed sensor nodes. The sensor node recursively computes the cell to which it belongs by applying the defined transformations and it then compares its coordinates with the obtained sub-square bounds.



(a) (b) (c)

Figure 5.1 (a) One-order attractor, (b) Four-order cells of attractors, (c) Sensor nodes spanned in the attractor cell

Figure 5.1(a) shows a one-order attractor while Figure 5.1(b) provides an overview of the same attractor at a four-order level, in sub-square "00". Note that nodes are aware of the dimensions of the deployment area, as such information can be gathered by means of network wide flooding. Since sensor nodes cannot use an infinite alphabet sequence of the code space Γ to determine their cell address, the k_{order} value that defines the number of iteration of the IFS, as well as the length of the cell address chosen, are finite and bounded. An address ϕ_{cell} of any cell is a finite sequence of elements of Γ such as:

$$\phi_{cell} = \phi_1\phi_2\phi_3\cdots\phi_{k_{order}}, \qquad \phi_i \epsilon\, \Gamma \qquad (5.3)$$

To allow communication between a sensor node and other nodes in its cell, a worst case scenario is considered, in which the distance between two nodes in any cell equals the cell's diagonal so that the cell's side l_{cell} is bounded as $l_{cell} \leq \frac{r}{\sqrt{2}}$. Since $l_{cell} = a/2^{k_{order}}$, the number of sequences k_{order} in the cell address can be chosen so that:

$$k_{order} \geq \frac{1}{2} + \log_2\left(\frac{a}{r}\right) \qquad (5.4)$$

where r denotes the sensor node radius. Thus, the cell address consists of the first part of the node address. Within each cell, nodes elect a dedicated entity that represents the Cell Controller (CC), to maintain the global states within the cell. The only function of CCs

is to be signaling points within the cell. To discriminate sensors in the same cell, the CC assigns an ID to each of them. Consequently, the complete address of a node can be written as:

$$\phi: \phi_{id}, \quad 0 \leq \phi_{id} < n_{cell} \qquad (5.5)$$

where n_{cell} denotes the number of nodes in the cell of address ϕ and ϕ_{id}, the node's identifier in that cell. A periodic refresh, issued by the CC, triggers an update of the address pool by removing the IDs of unavailable nodes.

5.3.1 The QoS Network Model

With the assigned address, each sensor node can participate in the network routing operations. As indicated above, each cell is equipped with a CC that manages the cell operations. Numerous studies address QoS routing for WSNs that aims to allow each node to select a path based on QoS requirements. Although this seems to be an interesting approach, we believe that, in general, the large scale of WSNs must allow using certain nodes to build a separate network used to signal operations, and allow others to perform regular network operations, such as routing. The resulting sub-network that includes the CCs can also integrate powerful entities such as the actor overlay networks design presented in [90] or any other resource-rich nodes. Doing so should almost free each node to periodically perform a dedicated role, as is the case in most WSN clustering schemes. Such network is called Cell Controller Networks (CCNs). The remaining nodes in the network, i.e., those which are neither elected CCs nor categorized as an actor or mobile robot, make up the second type of sub network, which mainly performs routing operations.

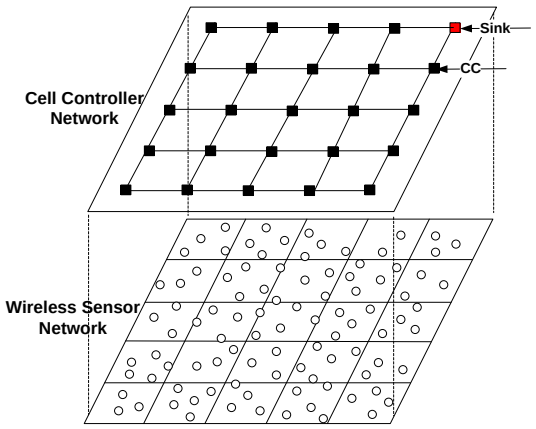


Figure 5.2 Two-layer overview of the WSN

Figure 5.2 illustrates both layers of the network plan. The lower level plan represents the entire network, while the higher level plan is exclusively composed of CCs. Our conjecture is that by performing only network signaling operations and occasional routing decisions, the CCs will support the same burden as a simple sensor node whose only role consists of performing data forwarding. Obviously, given the large number of nodes deployed, it is assumed that it is not wasteful for sensor nodes to be exclusively dedicated to performing signaling tasks. Before delving into the details of how QoS requirements are guaranteed, let us explicitly define the parameters selected as QoS metrics.

5.3.2 End-to-end QoS Parameters

Traditional networks specify QoS in a rigid manner, using metrics such as the average bandwidth, bounded delays and jitter requirements. In dynamic environments such as mobile wireless networks, it is difficult to provision and support rigid QoS. Works by [56], [57], which are similar to our proposal, deal with the so-called soft-QoS to fulfill QoS requirements by selecting end-to-end delays and path reliability as metrics.

In our proposal, we still strive to achieve hard-QoS constraints. For that purpose, and in order to integrate the energy consumption issue, End-to-End Path Battery Costs (EEPBCs) are introduced as additional metrics. In fact, our hypothesis is that, extending the lifetime of individual nodes will prevent early failures, thus helping to maintain the

hard QoS constraints. As a result, the proposed routing scheme, in addition to the end-to-end delay and path reliability, considers the residual energy of the forwarding nodes. Let $\mathrm{b_i(t)}$ denote the battery capacity of a sensor s_i at time *t*. The EEPBC $B_{s_o s_d}(\wp)$ between the source node $s_o$and the destination node s_d is defined as:

$$B_{s_o s_d}(\wp) = \sum_{p \in \wp} \sum_{s_i \in p} b_i(t) \qquad (5.6)$$

The rationale for choosing the residual energy in this proposal is that the information provided by the battery is easy to acquire and it offers an accurate level of energy consumed in the network. Since battery capacity is directly incorporated into the routing protocol, this metric prevents nodes from being overused, thereby increasing their lifetime.

Data can be reported at different rates depending on the type of application used or the heterogeneity of the nodes in the network. The readings generated by detecting nodes can be subject to various time deadline constraints. Let $d(s_i, s_{i+1})$ denote the delay between the sensor node s_i and neighbor s_{i+1}. The end-to-end delay between any source node s_o and the destination node s_d over the set of paths $\wp$ is given by:

$$D_{s_o s_d}(\wp) = \min_{p \in \wp} \left\{ \sum_{s_i \in p} d(s_i, s_{i+1}) \right\} \qquad (5.7)$$

Indeed, $D_{s_o s_d}(\wp)$ is the minimal achievable delay when the generated data are routed along the set of paths between s_o and s_d. The delay $d(s_i, s_{i+1})$ between two nodes consists of the elapsed time for successfully transmitting a packet after having received it. Thus, it includes queuing time, contention time, as well as transmission, retransmission and propagation times. Since our goal aims to allow CCs to participate in occasional routing decisions, one can expect that the delay experienced to route packets in CCNs will be lower than in SSNs, as they are subject to less congestion.

Reliability, often computed to assess the probability of transmission failures, can be characterized by a packet delivery ratio, defined as the ratio of the number of unique packets successfully received at the destination over the number of packets generated by

source nodes. The multipath end-to-end reliability between a source node s_o and a destination node s_d on a set of paths $\wp$ can be written as follows:

$$R_{s_o s_d}(\wp) = 1 - \prod_{p \in \wp} \big(1 - r(p)\big) \qquad (5.8)$$

where $r(p)$ is the reliability of a path p, equal to $\prod_{s_i \in p} r(s_i, s_{i+1})$ and $r(s_i, s_{i+1})$ denotes the reliability of the link (s_i, s_{i+1}). Since reliability is multiplicative, a variation in any links on p or an increasing number of hops on p will remarkably alter end-to-end reliability. That explains why single path routing cannot always meet reliability requirements. Once QoS parameters have been identified, one must define how such requirements are met. The following section defines the strategy used to put together these QoS parameters in an optimization problem.

5.3.3 Switching QoS Routing

In heterogeneous WSNs, a wireless node may also contain different sensors such as audio, video and scalar sensors. As the priority of such heterogeneous traffic differs, considering service differentiation appears to be an additional issue. There are two types of traffic: real-time traffic has hard time constraints, such as delays, although it is more tolerant of packet losses; non-real time traffic usually exists in networks and produces thousands of packets which are generated synchronously or asynchronously. In order to remain generic and not delve into the details of priority levels assigned to the different types of traffic, a general formulation of the QoS routing problem is considered. The joint goal of reducing energy consumption and end-to-end delays, while maximizing path reliability, appears to be very challenging. The routing objective thus consists of finding a set of paths $\wp(s_{o,}s_d)$ between source and destination that meet the specified QoS requirements at data source, while maximizing the EEPBC. The QoSNet routing problem can be formulated as follows:

QoSNet: Given the tuple (D_{req}, R_{req}) of QoS requirements specified by the source node, that represents the end-to-end delays and the reliability level, find

the set of paths $\wp(s_{o,}s_d)$ from the source node s_o to the destination node s_d that maximizes the end-to-end battery power, subject to deadline and reliability constraints.

$$max\, B_{s_o s_d}(\wp) \qquad (5.9)$$

Subject to:

$$D_{s_o s_d}(\wp) \leq D_{req} \qquad (5.9a)$$

$$R_{s_o s_d}(\wp) \geq R_{req} \qquad (5.9b)$$

Expression (8) maximizes the residual energy of the selected nodes, while constraints (8a) and (8b) express the delay and reliability requirements of the data source node, respectively. The problem of determining a QoS route that satisfies multiple constraints is known to be NP-hard [56]. The problem definition requires exact information regarding nodes on their paths to the destination, which is almost impossible to obtain in WSNs. Hence, to solve this challenge, each source node periodically collects information from its one-hop neighborhood. One-hop link metrics are much easier to acquire. Furthermore, such a strategy is scalable to the network size. By uniformly partitioning the QoS requirements at all downstream hops, the overall QoS requirements can be met. A node can satisfy the hop requirement by selecting next hop nodes based on the link condition. As in [56], the link delay L_i^d, the reliability L_i^r requirements and the local battery costs L_i^b for each node, considering the hop count h_i, are expressed as follows:

$$L_i^d = \left(D_{req} - D_i\right)/h_i \qquad (5.10)$$

$$L_i^r = \sqrt[h_i]{R_i} \qquad (5.10a)$$

$$L_i^b = \sum_{j \in F_w(s_i)} b_j(t) \qquad (5.10b)$$

where D_i is the actual delay experienced by a packet at node s_i from the source node, and R_i indicates the portion reliability requirement assigned to the path through node s_i, as decided by the upstream node of s_i. Moreover, $F_w(s_i)$ denotes the forwarding set to

be determined by the node. Based on the above definitions, the **QoSNet** routing problem can be reformulated for each node as follows:

QoSNet:

$$max\ L_i^b \tag{5.11}$$

$$\sum_{j \in N(s_i)} x_j log\ (1 - R_{ij}) \leq log(1 - L_i^r) \tag{5.11a}$$

$$x_j D_{ij} \leq L_i^d \tag{5.11b}$$

Expression (5.11) maximizes the residual energy of the forwarding set. Constraints (5.11a) express the fact that the selected links in the forwarding set must all together satisfy the local reliability requirement. Constraint (5.11b) expresses that the delay of the selected links must be lower than the local delay requirement. The forwarding set $F_w \subseteq N(s_i)$ which consists of the solution to this problem, is determined by the set of neighbors $N_{ssn}(s_i)$ in the SSN, or the set $N_{ccn}(s_i)$ in the CCN, so that:

$$\begin{cases} N_{ssn}(s_i) \cup N_{ccn}(s_i) = N(s_i) \\ N_{ssn}(s_i) \cap N_{ccn}(s_i) = \emptyset \end{cases} \tag{5.12}$$

The following section describes how nodes select the set that is subsequently considered to solve QoSNet. A typical forwarding scenario is described in Figure 5.3. Feasible paths are found in the SSN when the detected event is forwarded from the source node S, while node A, when solving the problem, finds feasible paths only in the CCN. When the node fails to find a feasible solution either in $N_{ssn}(s_i)$ or $N_{ccn}(s_i)$, the packet is dropped. Each forwarding node solves QoSNet and the packet is forwarded until it reaches its destination. Intermediate nodes might receive duplicate packets for the same session. For that purpose, each node maintains a table that records the tuple $\langle \phi{:}\,\phi_{\mathrm{id}}, seq, D_i \rangle$ of a data reporting session. As mentioned above, $\phi{:}\,\phi_{\mathrm{id}}$ identifies the reporting node and seq the associated sequence number. This table is emptied periodically and the node keeps a time stamp of its last clean out in order to maintain coherent routing timing. The rationale behind this is guided by the fact that a forwarding node which receives a packet twice, forwards the latter only when the previous experienced delay to forward this packet is greater than the actual delay. Indeed, let

D_i^{prev} denote the delay experienced by the packet previously received by node s_i for the same session. The node solves QoSNet to determine F_w only when the inequality $D_i < D_i^{prev}$ holds. So far, we have shown how a packet is forwarded, but how the QoSNet problem is actually solved has yet to be explained. In the following section, a promising strategy is developed in order to solve this issue, by applying site percolation to the discretized area.

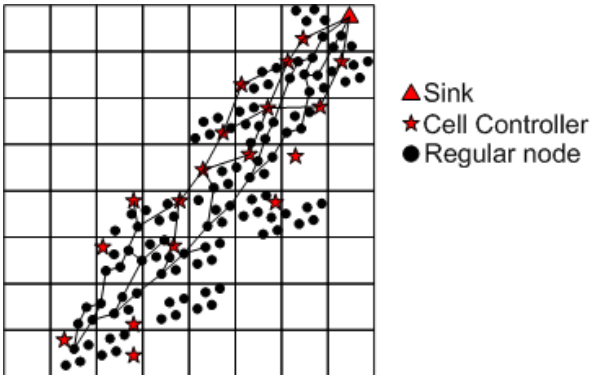


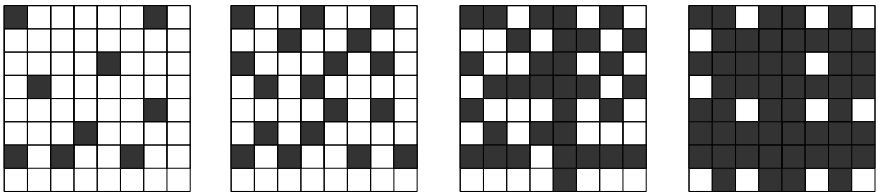
Figure 5.3 Data forwarding strategy in QoSNet

5.3.4 Mapping QoSNet Resolution to Site Percolation

The resolution of QoSNet is limited to the node's neighborhood $N(s_i)$, and can be achieved in polynomial time $O(|N(s_i)|)$. However, if the nodes in all cells determine their forwarding set by considering both their SSN and CCN neighbors, they will spend additional energy resources to those of the nodes in the CCN as they are already involved in the network signaling operations. To occasionally allow nodes in certain cells to consider both their SSN and CCN neighbors to solve QoSNet in such a situation, should help save energy resources and extend the network lifetime. One way of addressing this issue is to take a probabilistic approach based on the percolation theory. Introduced by Broadbent & Hammersley [95], the percolation theory has been widely applied in various fields, including economics, biology, sociology and communication. In

the field of wireless networks, this theory has been used in the past to study node connectivity.

Site percolation is a kind of random graph in which the edges are formed only between open neighboring sites. In site percolation, each site can be either open, with probability p_c, or closed, with probability $1 - p_c$. An edge exists only where two adjacent open sites are connected. All sites are connected and become a 'cluster' if any pair of open sites is connected. A cluster that spans from one side of the plane to the other is said to be an infinite cluster. A rigorous mathematical study of site percolation has been conducted in [74]. To model the QoSNet resolution to site percolation, consider the square grid in Figure 5.4, in which each cell represents a site. A site is considered *open* when the nodes in that site solve the QoSNet problem by considering their neighbor in the SSN, and in the CCN when they fail to find a solution with the SSN neighbors. A *closed* site corresponds to the situation when the node that has a packet to forward solves the QoSNet problem by using only its SSN neighbors. Obviously, when the value of the percolation probability is $p_c = 1$, all network sites are opened to solve the QoSNet problem in both SSN and CCN networks. Figure 5.4 illustrates the way network nodes solve the problem. A dark square represents the cells that determine their forwarding set by considering both their SSN or CCN neighbors exclusively, while the blank boxes represents the squares that solve it using the SSN neighbors only. First, the node solves the problem by considering its SSN neighbors, and if a feasible solution is not found, the node attempts to solve it by considering the CCN neighbors if their cell is open.



$p_c = 0.14$ $p_c = 0.28$ $p_c = 0.51$ $p_c = 0.8$

Figure 5.4 Percolation stages for QoSNet resolution

So far, we have shown how to model the QoSNet resolution by way of site percolation. However the way the various cells are selected has yet to be defined. Indeed, let $n(t)$ depict the number of instances the nodes in a given cell must solve QoSNet, to determine the forwarding set F_w during a period of time t; $n_{ssn}(t)$ the number of successful attempts to find a feasible solution when the nodes consider their SSN neighbors. These parameters are collected by the CC on a periodic basis. The CC activates its cell by notifying its members to also consider the CCN neighbors for QoSNet resolution. The probability p_c for a CC to be activated is computed as follows:

$$p_c = 1 - \frac{n_{ssn}(t)}{n(t)} \qquad (5.13)$$

Illustration: assume nodes in cell $\phi = 100001$ altogether have successfully found feasible solutions $n_{ssn}(t) = 100$ during the elapsed period of time t, while receiving $n(t) = 120$ to forward. The CC of that cell will activate its cell with the probability $p_c = 0.16$ to allow the nodes in that cell to consider the neighbors in $N_{ccn}(s_i)$ for the subsequent time period. If the cell is activated, its CC will notify its member, in order to use this new setting to process the incoming packets to be forwarded. Table 1 shows the QoSNet pseudo-code executed by the node when it has packets to forward.

Table 5.1 QoSNet Routing Algorithm

(A)When the node receive a packet to forward

1. $T(i) \leftarrow N_{ssn}(i);\ Forwarding\ set\ F_w(i) = \emptyset$
2. $candidate\ set\ C_w(i) = \{l_{ij}/h_j < h_i\ and\ D_i + d_{ij} <\ D_{req}, j \in T(i)\}$
3. **If** $C_w(i) \neq \emptyset$
4. compute $L_i^r = \sqrt[h_i]{R_i}$;
5. **do**
6. Select the node $v \in C_w(i)$ with the maximum residual energy b_v;

7. $F_w(i) \leftarrow F_w(i) \cup \{v\}$;
8. **until** constraint (5.9a) is met;
9. **else if** ($C_w(i) = \emptyset$ or $F_w(i) = \emptyset$) then
10. **If** the cell is activated then
11. $T(i) \leftarrow N_{ccn}(i)$; goto 2;
12. **else**
13. update statistics;
14. discards the packet and return;
15. **end if**
16. **end if**

(B) When the node is a **Cell Controller**

1. At the expiration of the timer $T_{percolation}$ do
2. **collect** the statistics from cell members;
3. **compute** the percolation probability p_c;
4. **activate** the cell with probability p_c and send notification to members;

End

5.4 Performance Evaluation

This section shows the effectiveness of the proposed routing scheme in terms of on-time packet delivery ratio, average end-to-end-delays, and the network lifetime. For that purpose, extensive simulations were conducted to assess the performance of QoSNet. Baseline comparisons were made with the MCMP [56] and the God routing. In the God routing, each node knows the state of all network links and nodes. This scheme can be viewed as an upper bound of a QoS routing algorithm with end-to-end delays and reliability requirements. The average delivery delay equals the latency experienced by successfully received packets. On-time packet delivery ratio means the number of successfully received packets that satisfy the QoS requirements over the total number of generated packets.

Our simulations assess the worst-case performance where link delays and reliability levels always change suddenly at any transmission instant, making them unpredictable. The success probability of each transmission is randomly picked from [0.8, 0.9], which implies that the link reliability ranges from 0.8 to 0.9. Link delays are also randomly distributed over the range of [1, 50] ms. Link delays consist of the time elapsed to

successfully transmit a packet after it was received. It thus includes queuing time, contention time, as well as transmission, retransmission and propagation times. The sink is located on the upper right corner of the sensor field. Over time, energy is dissipated at regular intervals until a routing hole appears after time T, which is known as the disconnection time or the network lifetime. Let T_i denote the disconnection time of an individual node. The network lifetime equals the time when the residual energy at any one of the sensor node reaches zero, and is given as:

$$T = \min(\{T_i\}_{i=1}^{n}) \qquad (5.14)$$

The simulation implemented in Qualnet follows the parameter settings shown in Table 5.2. The delay constraints are taken in the range of [60, 120] ms and the default reliability threshold is set at 0.7. For the MCMP routing, both parameters α and β are set at 95%. A data packet contains 128 bytes. Different random seeds were applied to generate different network configurations and all simulations lasted 1,000 sec. The results show an average of over 15 simulation runs.

Table 5.2 Simulation Parameters

Parameter	Value
Network area	200 m x 200 m
Sensors	256
Transmission range	40 m
Bandwidth	250 kbps
Packet size	128 bytes
Simulation time	1000 sec
Reliability	[0.8, 0.9]
Reporting rate	1 packet/sec
Percolation period	50 sec

To highlight how percolation affects the performance indexes, a scenario was simulated where each CC activates its cell with a given percolation probability. Figures 5.5, 5.6, and 5.7 respectively, illustrate the on-time packet delivery ratio, the average end-to-end delay, and the network lifetime for various percolation probabilities. Indeed, allowing the participation of CNN nodes in routing decisions helps improve the QoS requirements. However, there is a price to pay for such performance, as evidenced by the

network lifetime which decreases when many CCN nodes are involved in routing decisions.

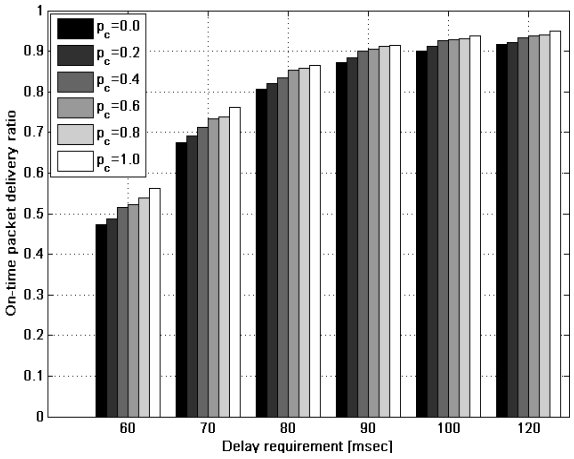


Figure 5.5 Effect of percolation on the delivery ratio

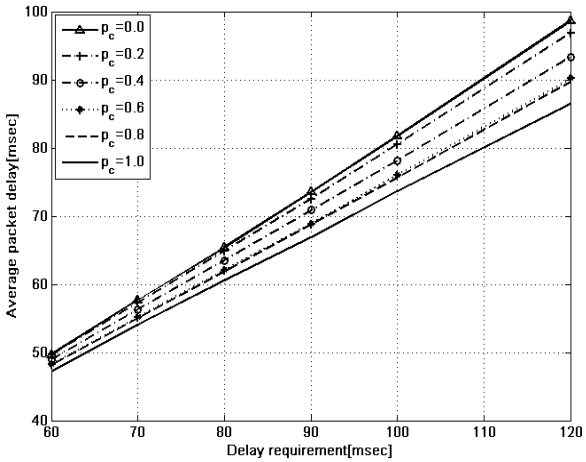


Figure 5.6. Effect of percolation on the average packet delay

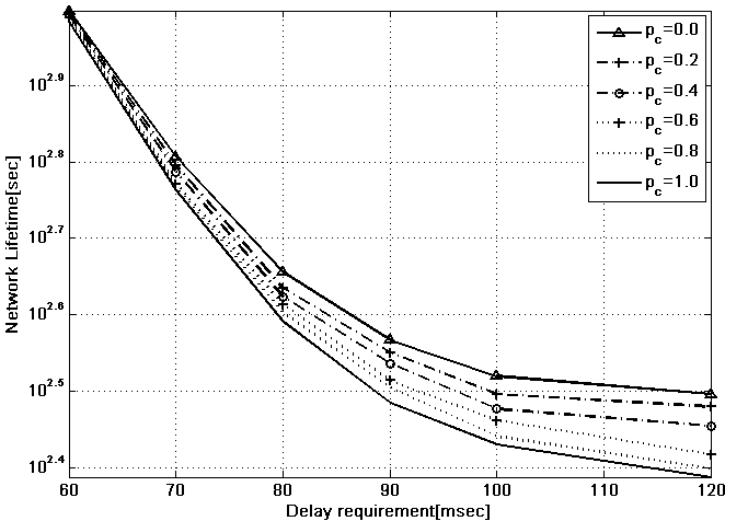


Figure 5.7 Effect of percolation on network lifetime

Figures 5.8 and 5.9 respectively present simulation results by comparing other routing schemes for on-time delivery ratio and the average latency experienced by the delivered data packets. The QoSNet routing scheme performs better in terms of delivery ratio when compared to the MCMP scheme, since it takes the residual energy of the routing nodes into account. This performance can be explained by the fact that some nodes in the MCMP scheme are frequently chosen and soon run out of energy. Obviously the

God routing scheme outperforms all of the studied schemes due to the fact that all network nodes are aware of the state of all network links.

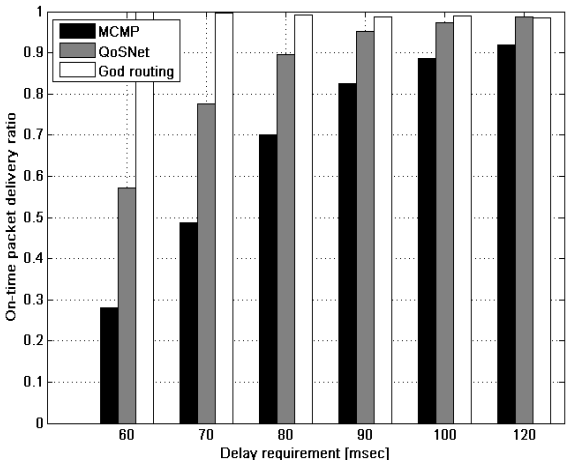


Figure 5.8 Average delivery ratio vs. delay requirement

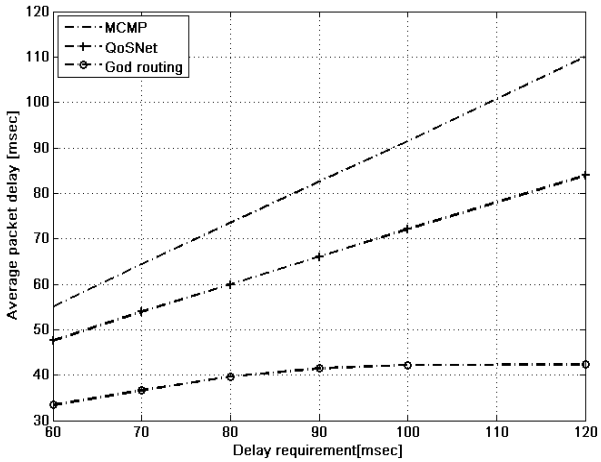


Figure 5.9 Average packet delay vs. delay requirement

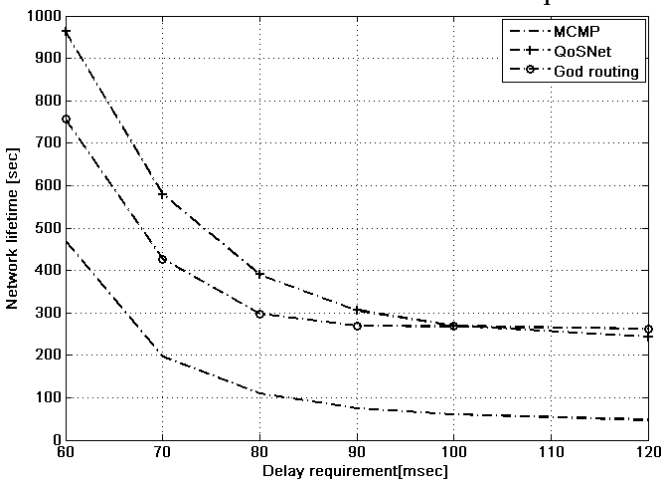


Figure 5.10 Network lifetime

The QoSNet scheme is more akin to the God routing as the nodes know the state of the local links, while the knowledge is global in the God routing scheme. Figure 5.10 illustrates the performance of the respective schemes in terms of network lifetime when the delay requirement varies. As expected, since the QoSNet scheme takes into account the residual energy of the nodes for path selection, it seems that the network lives longer than that of the MCMP scheme, which only minimizes the number of paths in order to save energy. The QoSNet scheme outperforms the God routing scheme for low delay

requirements as in God routing scheme, the node selects links with the least delays, potentially including nodes with low residual energy. However, the QoSNet scheme only selects links with nodes endowed with high residual energy: this explains why it outperforms God routing in terms of low delay requirements.

Figures 5.11 and 5.12 illustrate the results of the respective schemes when the value of the reliability requirement varies. As for the QoSNet scheme, the delivery ratio is tied to the God routing scheme and does not influence the performance indexes. This situation can be explained by the fact that the paths selected satisfied together the reliability requirements. However, the gap between the respective schemes in terms of the delivery ratio is due to the fact that when the nodes do not find a feasible solution, they drop the packet to be forwarded; MCMP thus drops more packets than other schemes.

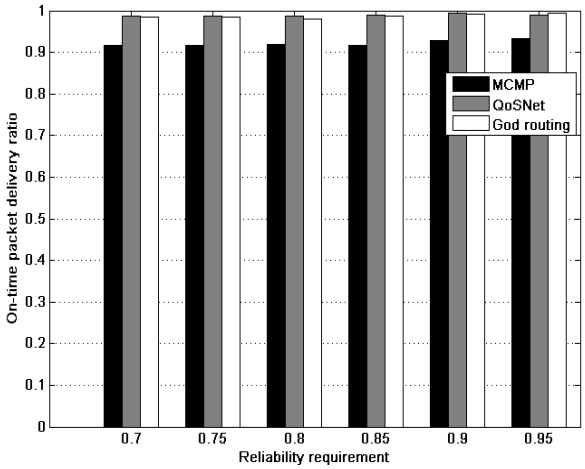


Figure 5.11 Average delivery ratio vs. reliability requirement

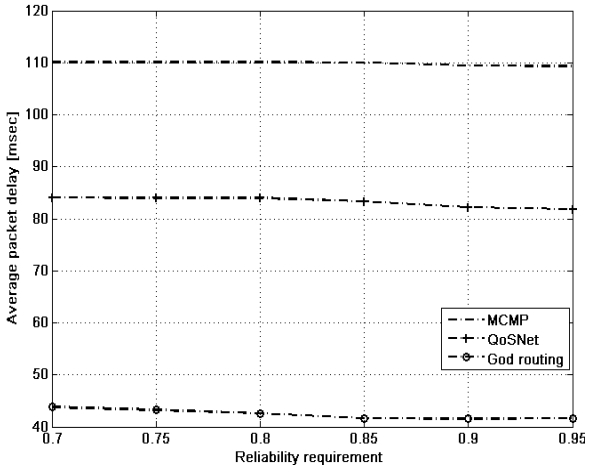


Figure 5.12 Average packet delay vs. reliability requirement

5.5 Conclusion

This paper presents a new routing strategy to provide QoS in large scale WSNs. Existing routing algorithms that aim to provide QoS, focus only on end-to-end delays and link reliability in relatively small wireless networks. To deal with the scale factor, the proposed scheme splits WSNs into two sub-networks composed of nodes spread over a

discretized area. The first includes cell controllers and possibly other powerful nodes such as actors, while the second is composed of the remaining sensor nodes. Considering the QoS metrics such as end-to-end delays and reliability, the proposed routing strategy is designed for large scale WSNs whose goal consists of extending network lifetime. By using the percolation theory, a switching mechanism allows the node that has a packet to forward to select forwarding nodes among its neighbors in the two sub-networks. Simulation results validate our scheme by showing its efficiency in terms of mean end-to-end delays, on-time packet delivery ratio, and most importantly, extended network lifetime.

CHAPITRE 6

DISCUSSION GÉNÉRALE

Ce chapitre fait une synthèse de nos travaux et contributions en regard de nos objectifs de recherche. Nous discutons l'approche méthodologique considérée et nous analysons les résultats obtenus ainsi que leurs portées.

6.1 Synthèse des travaux

Nous avons commencé cette thèse par une analyse critique de l'état de l'art dans le domaine. Ainsi, nous avons consacré le Chapitre 2 à une revue de littérature exhaustive. Cet état de l'art nous a permis de mettre en évidence les faiblesses et les lacunes des mécanismes existants, afin de dégager quelques problèmes ouverts pertinents qui ont servi comme orientations de recherche. La recherche menée dans cette thèse a donné lieu à cinq articles principaux de revue, dont trois présentés comme chapitres de cette thèse. Un de nos articles de revue est déjà publié tandis que les autres sont actuellement en cours d'évaluation.

Dans le premier volet de nos objectifs, nous avons considéré un MANET formé de nœuds mobiles susceptibles d'initier à tout instant des requêtes de données auprès d'un serveur distant possiblement connecté à un réseau filaire. Les nœuds mobiles peuvent subir des déconnexions intermittentes et disposent d'un espace mémoire dédié pour le stockage et le partage de données. Ces données sont supposées cohérentes durant une période de temps limitée (Time-to-Live). Dans ce contexte, nous avons abordé la problématique de l'accessibilité aux données en tenant compte du caractère dynamique de la topologie du MANET. Sachant que dans la littérature, les techniques traitant de l'accessibilité aux données dans de tels réseaux se basent soit sur la mise en cache, soit sur la réplication de données, nous avons proposé une approche hybride combinant ces

deux techniques. Pour mettre en évidence l'efficacité de la nouvelle approche mixte, nous avons conçu en premier lieu un mécanisme de mise en cache coopérative CoonCache utilisant le voisinage à un saut du nœud mobile. Dans ce mécanisme de mise en cache homogène, une infrastructure virtuelle dynamique, la NCT, effectue une liaison entre les espaces de stockage des nœuds voisins. En tenant compte de paramètres des données stockées tels que la taille, la durée de vie, ainsi que la fréquence d'accès, nous avons dérivé une expression analytique de la fonction d'utilité des données à purger pour assurer le renouvellement de la mémoire cache. Par la suite, nous avons introduit le mécanisme de mise en cache hybride qCache, qui combine CoonCache avec une technique de réplication partielle de données sur des nœuds clés (QR) sélectionnés à partir de leur ressource énergétique. Les QR stockent des copies de données fréquemment accédées par les nœuds mobiles et, contrairement aux nœuds mobiles simples (MUs), ils n'utilisent aucun mécanisme de remplacement de données. Pour effectuer la réplication partielle des données sur les nœuds clés, nous avons modélisé notre problème sous forme d'un problème de programmation linéaire en nombres entiers, que nous avons prouvé NP-complet. Ensuite, nous avons proposé une heuristique distribuée pour sa résolution en explorant également les techniques de réplication uniforme, proportionnelle, et racine-carrée. Ce volet a donné lieu à deux articles de revue dont seulement un est présenté dans cette thèse.

Le deuxième volet de nos contributions porte sur les RCSF formés de nœuds déployés aléatoirement à grande échelle, pour surveiller une zone donnée. Les capteurs détectent les événements et les acteurs exécutent des tâches en se basant sur l'analyse et la nature des événements rapportées par les capteurs. Plusieurs travaux conduits sur les RCSF assument l'existence d'un système d'adressage et d'une infrastructure de routage pour valider leurs propositions. Pourtant, l'identification des sources d'événements ainsi que l'acheminement des événements détectés constituent un défi de recherche partiellement exploré. Pour aborder ce défi, nous avons proposé un système distribué d'adressage et de routage des événements, basé sur un mécanisme de mise en grappe d'intérêts, et la

théorie fractale des systèmes de fonction itérés. Dans le but de minimiser les coûts d'acheminement des événements aux acteurs associés, un réseau virtuel surcouche (AON) composé d'acteurs est construit en prenant en compte les affinités des acteurs en termes d'intérêts aux événements. Les acteurs souscrivent préalablement à des intérêts culminant les événements auxquels ils sont intéressés. Puis les capteurs, une fois l'événement détecté, publient ce dernier aux acteurs intéressés susceptibles d'effectuer les actions appropriées. Il s'agit donc d'un système PS à partir duquel les acteurs sont regroupés dans l'AON suivant la similarité de leur souscription. À cet effet, nous avons développé un modèle mathématique générique de construction de l'AON que nous avons résolu avec une métaheuristique de recherche taboue pour l'obtention des différentes grappes d'intérêts. Pour permettre une allocation décentralisée d'adresses aux nœuds du réseau, l'aire de déploiement est discrétisée par chaque nœud à l'aide d'un système de fonction itéré. Les cellules issues de cette discrétisation sont identifiées chacune par une adresse représentant l'identité régionale de ses membres. Les CC allouent aux membres de leurs cellules des identifiants qui permettent ainsi à un nœud d'obtenir une adresse complète unique servant aussi à identifier la source des événements. L'information de localisation contenue dans l'adresse allouée, permet l'établissement des chemins de transit des événements détectés. Nous avons aussi démontré que le fait de discrétiser l'aire de déploiement des nœuds, permet d'atténuer les effets sur le routage, des erreurs de localisation. Ce volet a donné lieu à deux articles de revue dont seulement un est présenté dans cette thèse.

Le dernier volet de nos travaux porte sur le routage avec QdS des données issues des événements détectés. En effet, avec l'hétérogénéité des capteurs déployés et le caractère dynamique du trafic généré, le besoin d'acheminer les données d'un événement détecté dans des délais courts et borné à travers des liens de transit fiables posent tout un défi. En particulier, nous avons tenté de répondre à la question suivante : comment garantir, dans un réseau de capteurs hétérogène de grande taille, l'acheminement des données dans des délais cours et bornés, à travers des liens fiables tout en minimisant les coûts

énergétiques? Pour ce faire, nous avons développé un nouveau système de commutation entre deux réseaux virtuels issue de la discrétisation produite par les systèmes de fonctions itéré. Le CCN composé des CCs et le SSN composé de simples capteurs permettent d'assurer la QdS dont les métriques retenues sont le délai de bout-en-bout et la fiabilité des chemins. Pour garantir l'usage efficace de l'énergie à la disposition des nœuds, nous avons aussi inclus le EEPBC qui représente l'énergie résiduelle des nœuds dans les routes empruntées par les données jusqu'à la destination. D'une formulation sous forme de problème d'optimisation sous contrainte globale, nous avons proposé une reformulation du problème sous contrainte locale à chaque nœud. La formulation du problème sous contrainte locale, permet d'abord à un nœud ayant une information à transférer, de résoudre le problème en tenant compte des nœuds de son voisinage figurant uniquement dans le SSN. Puis, en l'absence de solution faisable, le problème est résolu en considérant les nœuds du voisinage qui se retrouvent dans le CCN.

6.2 Méthodologie

La proposition de nouveaux mécanismes et algorithmes doit être validée par une preuve de concept. Nous avons choisi, afin d'évaluer les performances des mécanismes proposés, d'utiliser une validation par simulations. Pour ce faire, le simulateur pour réseaux sans fil QUALNET, le logiciel MATLAB ainsi que l'environnement de développement Visual C++.Net ont été utilisés. De plus, plusieurs batteries de tests ont été effectuées pour la validation des algorithmes proposés suivant un plan d'expérience clair et assez complet.

6.3 Analyse des résultats

La validation par simulations des mécanismes et algorithmes proposés montre que les résultats obtenus sont satisfaisants. Pour affirmer la crédibilité de nos travaux, nous avons procédé à des analyses comparatives pour les différents algorithmes et mécanismes proposés. Les mécanismes auxquels nous nous sommes comparés étaient généralement proposés dans des publications scientifiques récentes. En effet, dans notre

premier article, les résultats de simulation ont montré que la coopération entre nœuds mobiles par l'intermédiaire de la NCT pour le partage des données, offre de meilleures performances comparées à des mécanismes existants. De plus, la combinaison des techniques de mise en cache coopérative et de réplication partielle de données, auparavant jamais essayée dans les MANET, démontrent de par les résultats satisfaisants obtenus, de l'intérêt d'utiliser cette technique dans l'implantation des MANET. Dans notre second article, les résultats des simulations démontrent de l'efficacité du nouveau système d'adressage et de routage pour les RCSF. En particulier la réduction des coûts de communication obtenus lors de l'acheminement des événements détectés, démontre de la nécessité de construire des grappes d'intérêts regroupant les acteurs ayant les mêmes centres d'intérêts. Par ailleurs, les résultats obtenus en mesurant l'impact des erreurs de localisation sur le routage, montrent clairement l'importance et l'efficacité de la discrétisation effectuée avec les IFS. Dans notre troisième article, les simulations montrent que notre nouvel algorithme de routage multi chemin basé sur la commutation entre SSN et CCN, et prenant en compte les métriques de QdS définis offre de bien meilleures performances en terme de délai d'acheminement des données et du taux de réception des paquets. En outre, la prise en compte de l'énergie résiduelle des nœuds dans une route permet, à travers les résultats obtenus, une prolongation de la durée de vie du réseau, comparativement à un algorithme de routage avec QdS récent de la littérature.

CHAPITRE 7

CONCLUSION

7.1 Sommaire des contributions

Cette thèse a donné lieu à trois principales contributions:

– La proposition de nouvelles stratégies de partage et de stockage de données dans les MANET. En effet, l'accessibilité aux données dans les MANET se base en général sur l'usage des deux techniques traditionnelles de stockage et de partage de données que sont la mise en cache et la réplication des données. Celles-ci sont utilisées exclusivement l'une par rapport à l'autre dans ces réseaux. Cela ouvre la voie à une nouvelle avenue offerte par la technique hybride que nous avons explorée. En particulier, pour mettre en évidence les performances de la technique hybride, nous avons proposé une stratégie homogène de mise en cache de données basée sur une infrastructure virtuelle dynamique de liaison des espaces de stockage des nœuds dans le voisinage à un saut. À cet effet, une technique de remplacement des données en cache, basée sur la coopération entre les nœuds du même voisinage et d'une fonction d'utilité dérivée analytiquement, a été mise en place. Celle-ci tient compte de la taille de la donnée, de sa durée de vie, et de sa fréquence d'accès. Par la suite, nous avons conçu le mécanisme hybride en combinant la technique de mise en cache coopérative précédente avec une réplication partielle de données sur une fraction de nœuds du réseau. Cette nouvelle technique peut servir de référence dans la mise en œuvre d'applications pour les MANET.

– La proposition d'un système d'adressage et d'une stratégie de routage des données d'événements dans les RCSF de grande taille. En effet, si les similitudes entre les

RCSF et les MANET sont importantes, ils diffèrent en particulier par le facteur d'échelle qui fait que les RCSF sont beaucoup plus peuplés de nœuds. Les problématiques relatives aux techniques d'adressage et de routage doivent donc êtres approchées autrement. À cet effet, nous avons conçu un réseau logique surcouche composé de grappes d'acteurs ayant des intérêts convergents. Le paradigme de communication PS a servi à la modélisation des interactions entre les entités du réseau. Le réseau surcouche est construit en maximisant la convergence des intérêts des acteurs, et en favorisant la construction de grappes de tailles similaires. Pour allouer des adresses de façon distribuée, nous avons conçu une stratégie de discrétisation de l'aire de déploiement du réseau en cellules dotées d'adresses, et d'un contrôleur qui alloue une adresse complète à chacun de ses membres. L'information de localisation contenue dans l'adresse allouée au nœud, permet le routage des données d'événements, tout en résistant aux impacts des erreurs de localisation.

- La proposition d'un protocole de routage multi-chemins intégrant des mécanismes de QdS. En effet l'hétérogénéité du trafic et en particulier la présence de capteurs multimédia induit la nécessité de garantir une certaine QdS pour l'acheminement des données. Pour répondre à ce besoin, le protocole de routage multi-chemins avec QdS, proposé effectue une séparation du réseau en deux sous-réseaux virtuels : le SSN et le CCN. En formulant le problème du routage avec QdS dans les RCSF comme un problème d'optimisation sous les contraintes de délais et de fiabilité des liens, nous avons conçu un mécanisme de commutation de la résolution du problème posé, basé sur la percolation, en considérant les voisins du nœud dans chacun des sous-réseaux.

Toutes les solutions proposées ont été validées par des simulations. Cette validation a permis d'avoir des résultats qui montrent une amélioration des performances par rapport aux mécanismes de référence de la littérature.

7.2 Limitations des travaux

Malgré les contributions énoncées ci-dessus, notre travail présente certaines limitations que nous pouvons citer :

Dans le chapitre 3, une limitation pourrait venir de la façon dont la cohérence des réplicas alloués aux QR est maintenue. En effet, dans notre approche du problème d'accessibilité aux données dans les MANET, lorsqu'un réplica n'est plus cohérent, sa mise à jour se fait seulement sur une demande, provenant potentiellement d'autres nœuds, ou du nœud lui-même. Cela a pour conséquence que des réplicas qui sont populaires, et qui ont une durée de vie (TTL) faible, feront fréquemment l'objet de demande de mise à jour auprès du serveur. Cela aura pour conséquence une augmentation du temps de réponse et une consommation des ressources du réseau. Toutefois, une piste de solution pour surmonter cette limitation serait de permettre au serveur de disséminer des mises à jour asynchrones de ces réplicas auprès des QR qui les hébergent.

Dans le chapitre 4, la limitation proviendrait de la granularité avec laquelle les sujets d'intérêt entre acteurs sont prédéfinis. Une trop fine granularité signifie que beaucoup de sujets d'intérêt ne permettraient pas la construction de grappes homogènes d'acteurs. Cela a pour conséquences : (a) d'engendrer des grappes de faible taille sans beaucoup d'intérêts en commun; (b) la dissémination des événements détectés deviendrait quasiment une diffusion à tous les acteurs du réseau. Une avenue de solution explorable serait de définir des sujets principaux d'intérêts avec une granularité fine, telle une ontologie du style de celle proposée dans [96].

Finalement, dans le chapitre 5, la collecte périodique des statistiques sur les liens pour permettre la décision de routage constitue toujours une source de consommation d'énergie. Les travaux existants sur le routage multi-chemin n'en font pas mention.

Pourtant ceci demeure une limitation à cause du fait qu'un minimum d'information soit nécessaire pour permettre de garantir la QdS. Tout dépend finalement de la périodicité avec laquelle ces informations sont collectées et de la façon dont leur imprécision peut affecter le mécanisme de routage proposé. Une possibilité de résolution de ce problème est la prédiction de l'état des liens dans le voisinage du nœud se basant sur des rapports périodiques produits de façon asynchrones.

7.3 Indication des travaux futurs

Les réseaux mobiles ad hoc (MANETs) et les réseaux de capteurs sans fil (WSNs) étant encore dans une phase de conception, plusieurs problèmes restent encore non-abordés. Les avenues pour des travaux futurs à cette thèse sont multiples. Nous en présentons quelques-unes qui s'inscrivent dans sa continuité logique.

Une première piste, qui découle directement de nos travaux, serait d'améliorer le mécanisme de réplication partielle conjointe à la mise en cache. En effet, dans la solution que nous avons proposée, seul le serveur détient l'originale des données partagées dans le réseau et est capable de les mettre à jour. Pourtant, il est d'usage que des nœuds mobiles détiennent aussi des données d'intérêt pour leurs pairs dans le réseau, des données qu'ils mettent eux-mêmes à jour. Une investigation de la stratégie de mise en cache et de celle de réplication partielle dans ces conditions est aussi nécessaire.

Une autre piste de recherche serait d'étendre le système d'adressage proposé pour les RCSF, afin qu'il intègre aussi un système de coordination et d'allocation de tâches aux acteurs dépendamment des événements rapportés. Cette étude consisterait concrètement à fusionner, dans l'AON du système d'adressage, le mécanisme de coordination et d'allocation des tâches aux acteurs. De même, la définition d'une ontologie des sujets d'intérêt pour les acteurs est une piste à investiguer pour permettre d'améliorer le mécanisme de formation de grappe des acteurs dépendamment de leurs intérêts.

Le protocole de routage multi-chemins avec QdS que nous avons proposé est plutôt générique et ne tient pas compte de la spécificité du trafic véhiculé dans le réseau. Une piste de recherche future serait d'effectuer une classification du trafic afin d'y allouer les ressources nécessaires à la garantie de la QdS. La mise en œuvre réelle des solutions proposées dans un déploiement réel de RCSF : cela permettrait de faire un test en grandeur nature des mécanismes proposés et attesterait par la même occasion de leur réalisme.

CONTRIBUTIONS À LA RECHERCHE

Le travail mené lors de cette thèse a donné lieu à plusieurs contributions scientifiques. Nous donnons ci-dessous la liste des différents articles :

Articles de revues publiés

T. Houngbadji, S. Pierre, “SubCast: A Distributed Addressing and Routing System in Large Scale Wireless Sensor and Actor Networks,” publié dans *Computer Networks (Elsevier).*

Articles soumis à des revues scientifiques

T. Houngbadji, S. Pierre, A. Quintero, “Joint Data Caching and Replication Schemes in Mobile Ad Hoc Networks”. Soumis pour publication à la revue *Ad Hoc Networks (Elsevier).*

T. Houngbadji, S. Pierre, A. Quintero, “On the Data accessibility in Mobile Ad hoc Networks”. Soumis pour publication à la revue *Wireless Communications and Mobile Computing (Wiley).*

T. Houngbadji, S. Pierre, “Address Allocation Strategies in Wireless Sensor Networks”. Soumis pour publication à la revue *Ad Hoc and Ubiquitous Computing (InderScience).*

T. Houngbadji, S. Pierre, “QoSNet: An Integrated QoS Network for Routing Protocols in Large Scale Wireless Sensor Networks,” Soumis pour publication à la revue *Computer Communications (Elsevier).*

BIBLIOGRAPHIE

[1] I. F. Akyildiz, S. Weilian, Y. Sankarasubramaniam, et E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, pp. 102-14, 2002.

[2] I. F. Akyildiz et I. H. Kasimoglu, "Wireless sensor and actor networks: research challenges," *Ad Hoc Networks*, vol. 2, pp. 351-367, 2004.

[3] P. Guillaume, S. Maarten van, et S. T. Andrew, "Dynamically Selecting Optimal Distribution Strategies for Web Documents," *IEEE Transaction on Computers*, vol. 51, pp. 637-651, 2002.

[4] S. Bakiras et T. Loukopoulos, "Increasing the Performance of CDNs Using Replication and Caching: A Hybrid Approach," in *Proceedings of the 19th IEEE International Symposium on Parallel and Distributed Processing*, 2005, pp. 92b-92b.

[5] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, et F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 2-16, 2003.

[6] J. N. Al-Karaki et A. E. Kamal, "Routing techniques in wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 11, pp. 6-28, 2004.

[7] B. David et E. Deborah, "Rumor routing algorithm for sensor networks," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, Atlanta, Georgia, USA, 2002, pp. 22-31.

[8] M. Robert, J. John, K. Frans, L. Jinyang, et D. Douglas, "CarNet: a scalable ad hoc wireless network system," in *Proceedings of the 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system*, Kolding, Denmark , 2000, pp. 61-65.

[9] Y. Liangzhong et C. Guohong, "Supporting cooperative caching in ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 5, pp. 77-89, 2006.

[10] C. Chee-Yee et S. P. Kumar, "Sensor networks: evolution, opportunities, and challenges," in *Proceedings of the IEEE*, vol. 91, pp. 1247-1256, 2003.

[11] M. Alan, C. David, P. Joseph, S. Robert, et A. John, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications* Atlanta, Georgia, USA, 2002, pp. 88-97.

[12] CMOP, "Projet CORIE," consulté le 02 Aout 2009, tiré de: http://www.ccalmr.ogi.edu/, 2009.

[13] E. P. Charles, T. M. Jari, W. Ryuji, N. Anders, et J. T. Antti, "Internet connectivity for mobile ad hoc networks," *Wireless Communications and Mobile Computing*, vol. 2, pp. 465-482, 2002.

[14] J. Wieselthier, G. Nguyen, et A. Ephremides, "Resource-Limited Energy-Efficient Wireless Multicast of Session Traffic," in *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, IEEE Computer Society, 2001.

[15] T. Hara, "Effective replica allocation in ad hoc networks for improving data accessibility," in *Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, 2001, pp. 1568-1576

[16] H. Takahiro, "Replica allocation methods in ad hoc networks with data update," *Mobile Network Application*, vol. 8, pp. 343-354, 2003.

[17] H. Takahiro, M. Norishige, et N. Shojiro, "Replica allocation for correlated data items in ad hoc sensor networks," *SIGMOD Record*, vol. 33, pp. 38-43, 2004.

[18] C. Edith et S. Scott, "Replication strategies in unstructured peer-to-peer networks," *SIGCOMM Computing Communication Review*, vol. 32, pp. 177-190, 2002.

[19] W. Zhijun, K. D. Sajal, et C. Hao, "A Scalable Asynchronous Cache Consistency Scheme (SACCS) for Mobile Environments," *IEEE Transaction on Parallel and Distributed Systems*, vol. 15, pp. 983-995, 2004.

[20] L. Sunho, L. Wang-Chien, C. Guohong, et R. D. Chita, "Cache invalidation strategies for internet-based mobile ad hoc networks," *Computer Communications*, vol. 30, pp. 1854-1869, 2007.

[21] L. Wenzhong, E. Chan, W. Yilin, et C. Daoxu, "Cache Invalidation Strategies for Mobile Ad Hoc Networks," in *Proceedings of the International Conference on Parallel Processing*, 2007, pp. 57-57.

[22] I. Chalermek, G. Ramesh, et E. Deborah, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th*

annual international conference on Mobile computing and networking, Boston, Massachusetts, United States: ACM, 2000.

[23] F. A. Ian, M. Tommaso, et R. C. Kaushik, "A survey on wireless multimedia sensor networks," *Computer Networks*, vol. 51, pp. 921-960, 2007.

[24] C. Guohong, Y. Liangzhong, et C. R. Das, "Cooperative cache-based data access in ad hoc networks," *Computer*, vol. 37, pp. 32-39, 2004.

[25] N. Chand, R. C. Joshi, et M. Misra, "An efficient caching strategy in mobile ad hoc networks based on clusters," in *Proceedings of the IFIP International Conference on Wireless and Optical Communications Networks*, 2006, pp. -5.

[26] S. Lim, W.-C. Lee, G. Cao, et C. R. Das, "A novel caching scheme for improving Internet-based mobile ad hoc networks performance," *Ad Hoc Networks*, vol. 4, pp. 225-239, 2006.

[27] D. Yu, K. S. G. Sandeep, et V. Georgios, "Improving on-demand data access efficiency in MANETs with cooperative caching," *Ad Hoc Networks*, vol. 7, pp. 579-598, 2009.

[28] T. Bin, H. Gupta, et S. R. Das, "Benefit-Based Data Caching in Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 7, pp. 289-304, 2008.

[29] Y. Liangzhong et C. Guohong, "Balancing the tradeoffs between data accessibility and query delay in ad hoc networks," in *Proceedings of the 23rd IEEE International Symposium on Reliable Distributed Systems*, 2004, pp. 289-298.

[30] P. Bellavista, A. Corradi, et E. Magistretti, "REDMAN: a decentralized middleware solution for cooperative replication in dense MANETs," in *Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications*, 2005, pp. 158-162.

[31] P. Bellavista, A. Corradi, et E. Magistretti, "Comparing and evaluating lightweight solutions for replica dissemination and retrieval in dense MANETs," in *Proceedings of the 10th IEEE Symposium on Computers and Communications*, 2005, pp. 43-50.

[32] L. Jun, H. Jean-Pierre, et E. Patrick Th, "PAN: providing reliable storage in mobile ad hoc networks with probabilistic quorum systems," in *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and computing*, Annapolis, Maryland, USA: ACM, 2003, pp. 1-12.

[33] K. Chen et K. Nahrstedt, "Integrated data lookup and replication scheme in mobile ad hoc networks," in *Optoelectronic and Wireless Data Management, Processing, Storage, and Retrieval*, Denver, CO, USA, 2001, pp. 1-8.

[34] P. Prasanna, G. Le, V. Anita, et A. Mohammed, "A survey of data replication techniques for mobile ad hoc network databases," *The VLDB Journal*, vol. 17, pp. 1143-1164, 2008.

[35] R. Droms, "Dynamic Host Configuration Protocol," *IETF*, RFC2131, 1997.

[36] R. C. Doss, D. Chandra, L. Pan, W. Zhou, et M. Chowdhury, "Lease Based Addressing for Event-Driven Wireless Sensor Networks," in *Proceedings of the 11th IEEE Symposium on Computers and Communications*, 2006, pp. 251-256.

[37] T. Rui, M. Hiroyuki, A. Tomonori, et Z. Bing, "Network-layer and MAC-layer Address Autoconfiguration in Self-organized Sensor Networks," in *Proceedings of the 6th International Conference on ITS Telecommunications*, 2006, pp. 1005-1010.

[38] W. Fang, Y. Liu, J. Wu, et D. Qian, "An On-demand Address Allocation Scheme for Query based Sensor Networks," in *Proceedings of the Eighth International Symposium on Autonomous Decentralized Systems*, 2007, pp. 173-179.

[39] Y. Zheng et D. Falko, "Dynamic Address Allocation for Management and Control in Wireless Sensor Networks," in *Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, 2007, pp. 292b-292b.

[40] C. Santashil Pal, S. Du, A. K. Saha, et D. B. Johnson, "TreeCast: a stateless addressing and routing architecture for sensor networks," in *Proceedings of the 18th International on Parallel and Distributed Processing Symposium*, 2004, p. 221.

[41] Y. Li-Hsing et T. Wei-Ting, "Flexible Address Configurations for Tree-Based ZigBee/IEEE 802.15.4 Wireless Networks," in *Proceedings of the 22nd International Conference on Advanced Information Networking and Applications*, 2008, pp. 395-402.

[42] P. Meng-Shiuan, F. Hua-Wei, L. Yung-Chih, et T. Yu-Chee, "Address Assignment and Routing Schemes for ZigBee-Based Long-Thin Wireless Sensor Networks," in *IEEE Vehicular Technology Conference*, 2008, pp. 173-177.

[43] S. Ramon "ZigBee Specification: ZigBee Document 053474r06, Version 1.0," *ZigBee Alliance Inc*, 2005.

[44] R. Ashwini, P. Shruti, V. Muthukumar, et A. Ajith, "HAUNT-24: 24-bit hierarchical, application-confined unique naming technique," in *Proceedings of the 5th International Conference on Intelligent Systems Design and Applications*, 2005, pp. 375-380.

[45] J.-L. Lu, F. Valois, D. Barthel, et M. Dohler, "Low-Energy Address Allocation Scheme for Wireless Sensor Networks," in *Proceedings of the 18th International Symposium on Personal, Indoor and Mobile Radio Communications*, 2007, pp. 1-5.

[46] D. Zhigao, Q. Depei, S. Stanczak, R. Heras, et L. Yi, "Auto-configuration of Shared Network-layer Address in Cluster-based Wireless Sensor Network," in *Proceedings of the IEEE International Conference on Networking, Sensing and Control*, 2008, pp. 148-153.

[47] K. Sohrabi, J. Gao, V. Ailawadhi, et G. J. Pottie, "Protocols for self-organization of a wireless sensor network," *IEEE Personal Communications*, vol. 7, pp. 16-27, 2000.

[48] T. He, J. A. Stankovic, T. F. Abdelzaher, et C. Lu, "A spatiotemporal communication protocol for wireless sensor networks," *Transactions on Parallel and Distributed Systems*, vol. 16, pp. 995-1006, 2005.

[49] E. Felemban, L. Chang-Gun, et E. Ekici, "MMSPEED: multipath Multi-SPEED protocol for QoS guarantee of reliability and. Timeliness in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 5, pp. 738-754, 2006.

[50] B. Najet et S. Yeqiong, "A New Routing Metric for Satisfying Both Energy and Delay Constraints in Wireless Sensor Networks," *Journal of Signal Processing and Systems*, vol. 51, pp. 137-143, 2008.

[51] P. Dario, M. Tommaso, et F. A. Ian, "Routing algorithms for delay-insensitive and delay-sensitive applications in underwater sensor networks," in *Proceedings of the 12th annual international conference on Mobile computing and networking* Los Angeles, CA, USA, 2006, pp. 298-309.

[52] E. Sinem Coleri et V. Pravin, "Energy efficient routing with delay guarantee for sensor networks," *Wireless Networks*, vol. 13, pp. 679-690, 2007.

[53] R. M. Mahmood, G. Sathish, et C. M. L. Victor, "Multiobjective Routing for Simultaneously Optimizing System Lifetime and Source-to-Sink Delay in Wireless Sensor Networks," in *Proceedings of the 29th IEEE International Conference on Distributed Computing Systems*, 2009, pp. 123-129.

[54] K. Akkaya et M. Younis, "Energy and QoS Aware Routing in Wireless Sensor Networks," *Cluster Computing*, vol. 8, pp. 179-188, 2005.

[55] P. K. Pothuri, V. Sarangan, et J. P. Thomas, "Delay-Constrained, Energy-Efficient Routing in Wireless Sensor Networks Through Topology Control," in *Proceedings of the IEEE International Conference on Networking, Sensing and Control*, 2006, pp. 35-41.

[56] H. Xiaoxia et F. Yuguang, "Multiconstrained QoS multipath routing in wireless sensor networks," *Wireless Networks*, vol. 14, pp. 465-478, 2008.

[57] B. B. Antoine et G. M. Kuzamunu, "Energy Constrained Multipath Routing in Wireless Sensor Networks," in *Proceedings of the 5th international conference on Ubiquitous Intelligence and Computing* Oslo, Norway, 2008, pp. 453-467.

[58] B. Deb, S. Bhatnagar, et B. Nath, "ReInForM: reliable information forwarding using multiple paths in sensor networks," in *Proceedings of the 28th Annual IEEE International Conference on Local Computer Networks*, 2003, pp. 406-415.

[59] Y. Fan, Z. Gary, L. Songwu, et Z. Lixia, "GRAdient broadcast: a robust data delivery protocol for large scale sensor networks," *Wireless Networks*, vol. 11, pp. 285-298, 2005.

[60] C. Jiannong, Z. Yang, X. Li, et C. Guohong, "Consistency of cooperative caching in mobile peer-to-peer systems over MANET," in *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*, 2005, pp. 573-579.

[61] C. Guohong, "On improving the performance of cache invalidation in mobile environments," *Mobile Network Application*, vol. 7, pp. 291-303, 2002.

[62] M. K. H. Yeung et K. Yu-Kwong, "Wireless Cache Invalidation Schemes with Link Adaptation and Downlink Traffic," *IEEE Transactions on Mobile Computing*, vol. 4, pp. 68-83, 2005.

[63] J. Shim, P. Scheuermann, et R. Vingralek, "Proxy cache algorithms: design, implementation, and performance," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, pp. 549-562, 1999.

[64] L. Qin, C. Pei, C. Edith, L. Kai, et S. Scott, "Search and replication in unstructured peer-to-peer networks," in *Proceedings of the 16th international conference on Supercomputing*, New York, New York, USA, 2002, pp. 84-95.

[65] O. Kariv et S. L. Hakimi, "An Algorithmic Approach to Network Location Problems II: The p-Medians," *SIAM Journal on Applied Mathematics*, vol. 37, pp. 539-560, 1979.

[66] B. Urgaonkar, A. G. Ninan, M. S. Raunak, P. Shenoy, et K. Ramamritham, "Maintaining mutual consistency for cached Web objects," in *Proceedings of the 21st International Conference on Distributed Computing Systems*, 2001, pp. 371-380.

[67] Qualnet, "Qualnet Simulator," consulté le 02 Juin 2007, tiré de: http://www.qualnet.com.

[68] L. Breslau, C. Pei, F. Li, G. Phillips, et S. Shenker, "Web caching and Zipf-like distributions: evidence and implications," in *Proceedings of the Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies* 1999, vol.1, pp. 126-134.

[69] S. Lei, G. Zhi-Min, T. Yong-Cai, W. Lin, et S. Yun, "Modeling Web objects' popularity," in *Proceedings of the International Conference on Machine Learning and Cybernetics*, 2005, pp. 2320-2324.

[70] T. Melodia, D. Pompili, et I. F. Akyildiz, "A Communication Architecture for Mobile Wireless Sensor and Actor Networks," in *Proceedings of the 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, 2006, pp. 109-118.

[71] T. Melodia, D. Pompili, V. C. Gungor, et I. F. Akyildiz, "Communication and Coordination in Wireless Sensor and Actor Networks," *IEEE Transactions on Mobile Computing*, vol. 6, pp. 1116-1129, 2007.

[72] C. Intanagonwiwat, R. Govindan, et D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th ACM annual international conference on Mobile computing and networking*, Boston, MA, USA, 2000, pp. 56-67.

[73] M. Samuel, J. F. Michael, M. H. Joseph, et H. Wei, "TAG: a Tiny AGgregation service for ad-hoc sensor networks," *SIGOPS Operating System Review*, vol. 36, pp. 131-146, 2002.

[74] H.-O. Peitgen, J. Hartmut, S. Dietmar, *Chaos and Fractals: New Frontiers of Science*, 2nd ed., New York: Springer-Verlag, 2004.

[75] R. C. Doss, D. Chandra, L. Pan, W. Zhou, et M. Chowdhury, "Lease Based Addressing for Event-Driven Wireless Sensor Networks," in *Proceedings of the 11th IEEE Symposium on Computers and Communications*, 2006,pp. 251-256.

[76] W. Fang, Y. Liu, J. Wu, et D. Qian, "An On-demand Address Allocation Scheme for Query based Sensor Networks," in *Proceedings of the Eighth International Symposium on Autonomous Decentralized Systems*, 2007, pp. 173-179.

[77] Y. Li-Hsing et T. Wei-Ting, "Flexible Address Configurations for Tree-Based ZigBee/IEEE 802.15.4 Wireless Networks," in *Proceedings of the 22nd International Conference on Advanced Information Networking and Applications*: IEEE Computer Society, 2008, pp. 395-402.

[78] C. Santashil Pal, S. Du, A. K. Saha, et D. B. Johnson, "TreeCast: a stateless addressing and routing architecture for sensor networks," in *Proceedings of the 18th International Symposium on Parallel and Distributed Processing*, 2004, p. 221.

[79] R. Ashwini, P. Shruti, V. Muthukumar, et A. Ajith, "HAUNT-24: 24-bit Hierarchical, Application-Confined Unique Naming Technique," in *Proceedings of the 5th International Conference on Intelligent Systems Design and Applications*: IEEE Computer Society, 2005, pp. 375-380.

[80] E. Patrick Th, A. F. Pascal, G. Rachid, et K. Anne-Marie, "The many faces of publish/subscribe," *ACM Computing Survey*, vol. 35, pp. 114-131, 2003.

[81] R. Vedantham, Z. Zhenyun, et R. Sivakumar, "Mutual Exclusion in Wireless Sensor and Actor Networks," in *Proceedings of the 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, 2006, pp. 346-355.

[82] R. Vedantham, Z. Zhuang, et R. Sivakumar, "Hazard avoidance in wireless sensor and actor networks," in *Proceedings of the 2nd International Conference on Broadband Networks*, 2005, pp. 690-702.

[83] N. Bulusu, J. Heidemann, et D. Estrin, "GPS-less low-cost outdoor localization for very small devices," *IEEE Personal Communications*, vol. 7, pp. 28-34, 2000.

[84] N. Bulusu, J. Heidemann, et D. Estrin, "Adaptive beacon placement," in *Proceedings of the 21st International Conference on Distributed Computing Systems*, 2001, pp. 489-498.

[85] J. Vesanto et E. Alhoniemi, "Clustering of the self-organizing map," *IEEE Transactions on Neural Networks*, vol. 11, pp. 586-600, 2000.

[86] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, et A. Y. Wu, "An efficient k-means clustering algorithm: analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 881-892, 2002.

[87] G. Fred et L. Fred, *Tabu Search*, Kluwer Academic Publishers, 1997.

[88] K. Brad et H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, Boston, MA, USA, 2000, pp. 243-254.

[89] L. Ye Ming et W. S. W. Vincent, "An energy-efficient multipath routing protocol for wireless sensor networks: Research Articles," *International Journal of Communication Systems*, vol. 20, pp. 747-766, 2007.

[90] T. Houngbadji et S. Pierre, "SubCast: A distributed addressing and routing system for large scale wireless sensor and actor networks," *Computer Networks*, doi:10.1016/j.comnet.2009.07.006, 2009.

[91] J. A. Stankovic, T. E. Abdelzaher, L. Chenyang, S. Lui, et J. C. Hou, "Real-time communication and coordination in embedded sensor networks," in *Proceedings of the IEEE*, vol. 91, 2003, pp. 1002-22.

[92] O. Chipara, Z. He, X. Guoling, C. Qin, W. Xiaorui, L. Chenyang, J. Stankovic, et T. Abdelzaher, "Real-time Power-Aware Routing in Sensor Networks," in

Proceedings of the 14th IEEE International Workshop on Quality of Service, 2006, pp. 83-92.

[93] Y. Xu, J. Heidemann, et D. Estrin, "Geography-informed energy conservation for ad hoc routing," in *Proceedings of the 7th ACM annual international conference on Mobile computing and networking*, Rome, Italy, 2001, pp. 70-84.

[94] G. Anastasi, M. Conti, M. Di Francesco, et A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 7, pp. 537-568, 2009.

[95] S. R. Broadbent et J. M. Hammersley, "Percolation processes I: crystals and mazes," in *Proceedings of the Cambridge Philosophical Society*, vol. 53, 1957, pp. 629-641.

[96] H. Yuheng, W. Zhendong, et G. Ming, "Ontology driven adaptive data processing in wireless sensor networks," in *Proceedings of the 2nd international conference on Scalable information systems* Suzhou, China: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007, pp. 1-2.

ANNEXES

ANNEXE A List of the Notations and Pseudo-codes used in Chapter 3

Notation	Description
d_k	Index of data item
TTL_k	Time-to-Live
λ_k	Update rate
λ_a	Request arrival rate
λ_k^a	Access rate of data item d_k
s_k	Size of the data item
p_k	Access probability to the data item
s_{req}	Size of a request
$H(u,v)$	hops counts between nodes u and v
B_w	Wireless Bandwidth
β_k	Read probability of data item
$c(d_k)$	communication cost to retrieve d_k
C	capacity of the MU cache space
$U(d_k)$	Utility function
C_{TOT}	Total capacity in the network
n_{QR}^{opt}	Optimal number of QRs
n_{MU}	Number of MU including the QRs
D	Data set owned by the server
$\boldsymbol{\gamma}$	Vector of the space capacity allocated to replicas
$\mathbf{n}$	Vector of the number of copies per replica
D_h	Hot data set owned by the server
m	Number of items owned by the server
D_c	Cold data set owned by the server
C_Q	Total storage capacity of the QRs
Q	QR set
M	MU set, including QRs
n_k	Number of copies of the data item d_k
c_{ij}	Communication cost of MU *i* accessing data on QR j
h_{jk}	Allocation cost of item d_k to the QR j
r_{jk}	the demand of item d_k from QR j

DqCA- Server

(A) Select the n_{QR}^{opt} most powerful MUs in terms of energy to build the QR set;

(B) Create the Hot and the Cold sets after sorting items in descending order of their β_k;

(C) Select a replication strategy to form the allocation vector **n;**

Repeat until the sharing space is filled

a. Randomly pick n_k QRs from the QR set, and assign one copy of the associate item d_k to each of them;

b. Add the new item to the QR item list;

(D) Broadcast the QR list with the associated items;

(E) **When** a request arrives

Update or Create the entry of the requester in the MUTable;

Send d_k to the requester;

(F) At the reallocation period

If QR expansion is needed

Select the most powerful MUs in term of energy;

Refresh the Hot and the Cold Sets by resorting items based on their β_k;

if changes occur in the Hot set

Refresh the replicas by assigning the new items that appears in the Hot set and by removing those which migrate from the hot to the cold set;

Broadcast the QR list with the associated items;

DqCA-MU

(A) **When** the MU is selected as QR

a. **When** the QR receives a request;
if the requested item exists in the *Store*
Send d_k to the requester;
else
Forward the request to the Server;

b. **When** the QR receives new replicas at the reallocation period
Remove the missing items from the *Store*;
Insert the fresh items in the *Store*;

c. **When** the QR receives a response after requesting a data item d_k
Refresh the data item d_k
Send d_x to the requester;

(B) **When** the MU is not selected as QR

a. **When** the MU has a request to send
if d_k exists in cache and its TTL does not expire
Send d_k to the requester application
else if d_k exists in NCT and its TTL does not expire
Extract the address *vAddr* of the corresponding one-hop neighbor ;
Forward the request to *vAddr*;
else if the TTL of d_k expires, or d_k does not exist in cache or NCT
if d_k exist in the QR list
hq ←return the hop count to the QR which has d_k,and the hop count to the server **hs**;
if hq < hs
Forward the request to the QR
else
Forward the request to the next hop node in the path toward the server;
end
end
end

b. **When** the MU receives the QR list from the server,
Create or Update an entry for each QR;

c. **When** a data item d_k arrives
if an old version of d_k exist in cache
Update the cached copy and send d_k to the requester application;
else if d_k meta-data exist in the NCT and cache d_k if the cache is not full
Remove d_k meta data in the NCT
else if the cache is full
Cache d_k by applying the replacement policy **Greedy-U**;
Send $d_k$to the requesting application;
end

ANNEXE B List of the Notations and Frame used in Chapter 4

Notations	Description
n	Number of nodes in the network
n_a	Number of actor nodes
Δ	Number of iterations for termination criterion
r	Transmission range
k_{order}	Number of recursions for IFS cell computing
n_{cell}	Number of nodes in a cell
n_c	Number of clusters
n_s	Number of actors' subscriptions
Φ_{cell}	Cell's address
Φ_{id}	Node identifier in a cell
L	Network area side
l_{cell}	Cell side
Γ	Addressing alphabet
S_A	Actors set
n_t	Number of topics
T	Set of topics
$H_r(Z)$	Relative entropy of the clustering vector Z
θ	Zipf factor
m_{cell}	Number of cells
M	Number of affine transformations
u	Transformation indexes

Frame used for all exchanges

Source Address	Next hop Address	Destination Address	Packet type	Sequence Number	Payload

Source Address: network address of the source node; empty when the source identification does not matter.

Next hop Address: network address of the next hop node; empty for Advertisements;

Destination Address: network address of the destination node; empty for Actor/Sink Advertisements.

Packet type: denotes the type of packet.

"0": Subscription Packet;

“1”: Data Packet;

”2”: Actor Advertisement Packet;

“3”: Sink Advertisement Packet.

Sequence Number: a number that uniquely identifies a notification.

Payload: the useful data.