Réseaux ad hoc aidés par satellites pour les communications d'urgence

Monia Hamdi

N° d'ordre : 2012telb0227

Sous le sceau de l'Université européenne de Bretagne

Télécom Bretagne

En habilitation conjointe avec l'Université de Rennes 1

Ecole Doctorale - MATISSE

Réseaux ad hoc aidés par satellites pour les communications d'urgence

---

Thèse de Doctorat

Mention : Informatique

Présentée par **Monia Hamdi**

Département : Micro-ondes

Directeur de thèse : Xavier Lagrange
Encadrant de thèse : Laurent Franck

Soutenue le 05 mars 2012

Rapporteurs :
Prof. AHMED, Toufik, enseignant-chercheur, ENSEIRB-MATMECA/LABRI-CNRS
Prof. GAYRAUD, Thierry, enseignant-chercheur, Université Paul Sabatier/LAAS-CNRS

Examinateurs :
Prof. BOUSQUET, Michel, enseignant-chercheur, ISAE
Prof. LAGRANGE, Xavier, enseignant-chercheur, Télécom Bretagne
Prof. FRANCK, Laurent, enseignant-chercheur, Télécom Bretagne
Mme MONIER, Mélanie, ingénieur, EADS Astrium Satellite

Abstract

The services offered by traditional wireless systems such as cellular networks depend on established infrastructures. In the aftermath of a disaster, communication infrastructures may be totally destroyed. Therefore, there is a need for mobile technologies being independent of infrastructure where network management is the result of cooperation among terminals. Ad hoc networks are a promising technology for emergency communications requiring easily deployable and dynamically adaptable topologies. However, ad hoc networks can experience severe impairments because of node motion in the case of Mobile Ad hoc NETworks (MANETs) or scarce node density in the case of Wireless Mesh Networks (WMNs).

We first consider a scenario of forest fire fighting operations where the nodes form a MANET. Due to node mobility, network partitioning may occur. The network is then split into unconnected groups. The limited availability of communications in this situation compromises the rescue and recovery operations. Satellites may help in this respect by setting up temporary back up links and bridging the unconnected islands. We particularly study the problem of selecting nodes that will provide access to the satellite capacity (i.e. gateway nodes). A key challenge in this scenario is that nodes are mobile resulting in topology changes. We address this challenge by extending the use of the clustering techniques initially designed to solve scalability issues in MANETs.

K-hop clustering techniques allow to form virtual groups called *clusters* and to select particular nodes called *clusterheads* providing access to the satellite communication. Applying this technique to our scenario raises two problems : the node mobility modeling and the cluster structure maintenance. For the first problem, we compare a specific topology model, *FireMobility* to two generic topology models *Random Walk* and *Reference Region Group Mobility* model. We show the importance of including dedicated behavioral patterns to represent the actual field activity and its impact on the accuracy of the simulation results. For the cluster maintenance problem, we propose a new maintenance policy allowing to minimize the overhead in signaling, so as to save bandwidth and energy.

Finally, we consider a scenario where the nodes form a WMN. Satellite links are set up to bridge the different parts of the network and to connect the disaster area with headquarters. For this scenario, we propose a selection procedure based on genetic algorithm.

Remerciements

Mes travaux de recherche ont pu être menés jusqu'à terme grâce à l'aide précieuse de certaines personnes auxquelles j'exprime toute ma gratitude et ma reconnaissance.

Je tiens donc à remercier M. Xavier Lagrange, mon directeur de thèse et M. Laurent Franck, mon encadrant pour leur encouragement, leur disponibilité et leurs conseils.

Je remercie également le jury pour l'intérêt qu'il a porté à mon travail. Grâce aux remarques et aux corrections apportées, ce manuscrit a pu être amélioré et enrichi.

Je ne saurais terminer sans remercier ma famille et mes amis pour leur présence et leur soutien durant toutes mes études.

Finalement, un pensée particulière à ma mère pour son encouragement continuel et pour son soutien inconditionnel.

Sigles

3

3hBAC	3-hop Between Adjacent Clusterhead

A

AODV	Ad hoc On Demand Distance Vector

C

CDS	Connected Dominating Set
CEMCA	Connectivity, Energy and Mobility Driven Clustering Algorithm
CSC	Connectivity-based Stretchable Clustering scheme

D

DARPA	Defence Advanced Research Agency
DCA	Distributed Clustering for Ad hoc networks
DGMA	Distributed Group Mobility Adaptive Clustering Algorithm
DKR	Distributed (K,R) dominating set
DMAC	Distributed and Mobility-Adaptative Clustering
DNDP	Discrete Network Design Problem
DSCAM	Distributed Scenario-based Clustering Algorithm for MANET
DSR	Dynamic Source Routing

K

KCMBC	K-hop Compound Metric Based Clustering
k-CONID	k-hop CONnectivity-IDentity based clustering algorithm

L

LCC	Least Cluster Change

M

MCDS	Minimal Connected Dominating Set
MIS	Maximal Independant Set
MOBIC	Mobility Based Metric for Clustering
MST	Minimum Spanning Tree

O

OLSR	Optimized Link State Routing Protocol

R

RdP	Réseau de Petri
RRGM	Reference Region Group Mobility model

S

SCPP	Satellite Capability Placement Problem

W

WACA	Weighted Application Aware Clustering Algorithm
WACHM	Weight Based Adaptive Clustering for Large Scale Heterogeneous MANET
WBACA	Weight Based Adaptive Clustering Algorithm
WCA	Weighted Clustering Algorithm
WMN	Wireless Mesh Network
W-GCP	Weight based Greedy Cluster Partitioning

Table des figures

Liste des tableaux

Table des matières

Chapitre 1

Introduction

Sommaire

1.1 Contexte général

Accéder à des services de communications quel que soit l'endroit, le moment et le type du terminal n'est plus un concept utopique. La prolifération croissante de terminaux mobiles avec de réelles capacités de calcul et de stockage (PDA, ordinateurs *ultrabook*, tablettes, *smartphones*, etc.) ouvre de nouvelles perspectives en termes d'architectures, d'applications et de technologies. La révolution technologique de l'Internet mobile a transformé la manière dont la société perçoit les communications et traite les informations. Elle a sans cesse besoin d'échanges rapides d'informations et l'informatique mobile gagne de plus en plus de popularité. Mais dans la plupart des systèmes mis en place par les opérateurs ce sont surtout les terminaux qui sont mobiles. Les services de communications sont encore basés sur une infrastructure fixe dont le déploiement nécessite du temps et engendre des coûts importants. Alors pourquoi ne pas imaginer un système de communications sans infrastructure, où la topologie peut évoluer constamment ? D'où l'émergence d'un nouveau mode de fonctionnement, le mode ad hoc.

La technologie des réseaux ad hoc a fait l'objet d'une attention particulière de par les

propriétés qu'elle possède en termes d'auto-organisation et d'auto-adaptation. Chaque terminal fait office de relais pour acheminer les informations de proche en proche. Après une configuration initiale, les terminaux découvrent et utilisent les ressources disponibles, partagent des données et des applications sans besoin d'une administration centralisée. La décentralisation, la rapidité et la simplicité de la mise en oeuvre ont ouvert de nombreuses perspectives pour les réseaux ad hoc, conçus au départ pour des applications militaires. Une application importante est la mise en place rapide d'un réseau de communications d'urgence, dans le cas d'une catastrophe sur des zones dépourvues d'infrastructure ou dans le cas où l'infrastructure existante est saturée, voire complètement détruite.

1.2 Motivations et Contributions

1.2.1 Motivations

Le mode de fonctionnement ad hoc n'est pas particulièrement lié une technologie de transmission ; les terminaux peuvent être équipés de puces IEEE 802.11 (*Wi-Fi*) ou IEEE 802.15 (*Bluetooth*). Mais les communications se basent essentiellement sur la transmission radio. Si l'environnement sans fil offre une grande flexibilité, il engendre de nombreux problèmes tels que les déconnexions fréquentes et les débits variables. De nouvelles techniques ont été développées pour pallier ces problèmes, mais certains problèmes persistent encore malgré les efforts de miniaturisation et de réduction des coûts. Ces problèmes sont essentiellement liés à la mobilité des terminaux où les changements topologiques peuvent engendrer l'indisponibilité des services de communications.
L'objectif de cette thèse est de traiter le problème de restauration de la connexité dans un réseau ad hoc déployé dans des situations d'urgence. Nous considérons dans un premier temps un scénario de lutte contre les feux de forêt, où les unités d'intervention forment un réseau ad hoc mobile. L'environnement est caractérisé par la mobilité des utilisateurs et la dynamique de la disponibilité des ressources. Dans un tel environnement, une des principales causes de la défaillance du système est le partitionnement du réseau. Or, dans les situations d'urgence et de secours, une telle interruption est intolérable car elle peut mettre en péril la vie des équipes de secours. La solution envisagée est d'utiliser un satellite dont la disponibilité n'est pas perturbée par les catastrophes naturelles et les incidents industriels. Ce satellite, fonctionnant en mode bidirectionnel, permet de rétablir la liaison entre les partitions du réseau. Ensuite, nous considérons un deuxième scénario où les noeuds forment cette fois-ci un réseau ad hoc *mesh*. Déployés sur une zone dépourvue de toute infrastructure de communications, les noeuds passent par un relais satellite pour se connecter au réseau dorsal (*backbone*). De la même manière que le premier scénario, nous nous intéressons particulièrement au choix des passerelles qui vont avoir accès satellite et fournir le service de communications.

1.2.2 Contributions

Les travaux réalisés dans le cadre de cette thèse consistent à étudier les mécanismes de réparation de connexité dans les réseaux ad hoc mobiles et *mesh*. Notre contribution se résume dans les points suivants :

- Analyse théorique des algorithmes permettant de structurer les réseaux ad hoc mobiles (chapitre 3) : en effet, notre problématique est de choisir un sous-ensemble de noeuds qui vont avoir accès aux communications satellite. Le but de cette analyse est de montrer les avantages et les limites de deux principales propositions : les ensembles dominants connectés et les ensembles dominants indépendants. Cette étude est fortement liée à notre cas de figure : un réseau ad hoc mobile où le réseau peut être temporairement partitionné. Cette étude a abouti à la sélection de l'algorithme qui correspond au mieux aux critères que nous avons imposés.

- Étude de la problématique de la modélisation de mobilité (chapitre 4) : grâce à une étude comparative entre les modèles de mobilité génériques et dédiés, nous avons expliqué comment le modèle et la manière dont les noeuds sont répartis sur le terrain peuvent influencer les résultats de simulation. Nous avons utilisé particulièrement un modèle de mobilité conçu spécialement pour un scénario de lutte contre les feux de forêt, *FireMobility*. Ce modèle a été comparé à deux autres modèles génériques : *Random Walk* où la répartition des noeuds est complètement aléatoire et *Reference Region Group Mobility model* qui introduit la notion de groupe. Nous avons montré qu'un modèle dédié est indispensable pour faire apparaître les caractéristiques et les propriétés du scénario étudié.

- Proposition d'une nouvelle procédure de maintenance (chapitre 5) : cette procédure a pour but de réduire le surcoût en signalisation. En effet, l'algorithme choisi pour résoudre le problème de sélection des noeuds crée une structure virtuelle appelé *cluster* où le *clusterhead* assure l'interconnexion de la partition avec les autres partitions du réseau grâce au lien satellite. La mobilité des noeuds et les changements topologiques du réseau nécessitent un mécanisme de *monitoring* qui permet de garantir à chaque instant une certaine qualité de *clustering*.

- Proposition de l'algorithme génétique pour résoudre le problème du choix des passerelles dans un réseau *mesh* (chapitre 6) : comme dans les réseaux ad hoc mobiles, les réseaux *mesh* peuvent être déployés dans des zones dépourvues de toute infrastructure de communications. Le satellite est proposé comme solution pour interconnecter les réseaux *mesh* au centre des opérations et de contrôle. Nous avons montré l'équivalence entre notre problème initial qui est le chois des passerelles satellite avec un autre problème

connu dans la littérature, le *Discrete Network Design Problem* pour justifier l'utilisation de l'algorithme génétique dont les performances sont ensuite comparées à celles de deux algorithmes basés sur la théorie des graphes (W-GCP et *Recursive*).

Publications

Les travaux de recherche élaborés durant cette thèse ont abouti à une contribution lors d'une école d'été et deux articles dans des conférences. Deux autres articles sont soumis dont un dans une revue, en attente d'évaluation.
La première contribution [1] couvre les aspects généraux de la thèse, notamment l'introduction de l'environnement des réseaux ad hoc mobiles et le problème de partitionnement. Dans cet article, nous avons présenté les différentes approches qui permettent de structurer les réseaux ad hoc : la construction de l'épine dorsale et la technique du *clustering*. La simulation de la mobilité des noeuds soulève le problème de la représentation mathématique des schémas de mobilité. La comparaison entre le modèle de mobilité dédié *FireMobility* et un modèles de mobilité générique, *Random Walk* a fait l'objet de l'article [2]. Le but de cet article est de montrer l'importance de développer des modèles dédiés pour en dégager les propriétés du scénario considéré. L'article montre l'impact du modèle sur les propriétés topologiques du réseau et les performances des protocoles. Pour remédier au problème du partitionnement des réseaux ad hoc mobiles, nous proposons de mettre en place un service de communications par satellite pour relier les différentes partitions. La technique de *clustering* et particulièrement l'algorithme KCMBC (*K-hop Compound Metric Based Clustering*) a été sélectionné pour résoudre le problème de sélection des noeuds qui vont avoir accès au satellite. Pour suivre la dynamique du réseau, un mécanisme de maintenance est nécessaire pour garantir la structure des *clusters*. Le surcoût en signalisation introduit par la procédure de maintenance fait l'objet de l'article [3] où nous proposons une nouvelle procédure de maintenance adaptée aux réseaux partitionnés.

Deux articles ont été soumis et en attente d'évaluation. Le premier article soumis [4] est la suite du travail publié dans [2], où nous introduisons un deuxième modèle de mobilité générique, *Reference Region Group Mobility* qui essaye de se rapprocher des modèles dédiés en définissant le concept de mobilité de groupe. Le dernier article soumis [5] analyse les problématiques liées à l'application de la technique de *clustering* à un réseau partitionné. En effet, cette technique a été originellement proposée pour résoudre le problème du passage à l'échelle dans des réseaux connectés.

1.3 Organisation du document

Ce document est organisé en 7 chapitres. Le premier chapitre, l'introduction, présente le contexte général de ce travail ainsi que nos motivations et contributions. Le chapitre 2 est un chapitre introductif qui présente les caractéristiques des réseaux ad hoc *mesh* et mobiles et les principales avancées techniques qui transforment les réseaux ad hoc d'un simple concept théorique à des réseaux fonctionnels. Nous présentons aussi les problématiques liées à leur déploiement pour les communications d'urgence, notamment le problème du partitionnement.

L'utilisation du satellite pour la restauration de la connexité du réseau soulève le problème du choix des noeuds qui vont avoir accès au segment spatial. Le chapitre 3 introduit les principales approches permettant de structurer le réseau ad hoc mobile. Nous détaillons principalement les propriétés de deux approches : la structuration du réseau sous forme d'un ensemble dominant connecté appelé aussi dorsale et d'un ensemble dominant indépendant appelé aussi *clusterheads*. Dans ce chapitre, nous discutons l'adéquation de chaque approche à la problématique du choix des noeuds et décrivons les aspects techniques liés à leur application dans un scénario de lutte contre les feux de forêt.

L'un de ces aspects est la modélisation mathématique des schémas de mobilité des noeuds. En effet, nous considérons dans un premier temps un scénario de lutte contre les feux de forêt. Le chapitre 4 présente les différents modèles de mobilité existants que nous classons en deux principales catégories : générique et dédié. A travers la description du mode de fonctionnement des différents modèles sélectionnés dans les deux catégories, nous démontrons l'inadéquation des modèles génériques à notre scénario. Cette conclusion est consolidée par simulation où nous comparons *FireMobility*, un modèle dédié, développé spécialement pour le scénario de lutte contre les feux de forêt à deux modèles génériques *Random Walk* et *Reference Region Group Mobility*. Grâce à la simulation, nous démontrons l'importance du développement d'un modèle dédié au scénario étudié.

Améliorer les performances du réseau grâce à la réduction du surcoût induit par les messages de signalisation est le sujet du chapitre 5. Dans ce chapitre, nous proposons une nouvelle procédure de maintenance que nous appelons *Passive Maintenance*. Cette procédure évite l'envoi périodique des informations sur l'état des *clusterheads*, en exploitant d'autres messages de signalisation envoyés par les noeuds dans KCMBC. Le fonctionnement de cette procédure est tout d'abord validé grâce à un réseau de Petri. Les performances de *Passive Maintenance* sont ensuite analysées et comparées à une autre procédure de maintenance largement utilisée dans la littérature, *Periodical Broadcast*. Les résultats de simulation montrent que *Passive Maintenance* permet de réduire considérablement le

surcoût en signalisation requis pour maintenir la structure des *clusters* ; ce qui permet de réduire la consommation en énergie et économiser de la bande passante.

Outre le scénario de réseaux ad hoc mobiles, nous considérons un deuxième scénario où les noeuds sont quasi-statiques. Le chapitre 6 discute la problématique du choix des noeuds dans un réseau *mesh*. Nous ramenons tout d'abord le problème à un autre problème connu dans la littérature, le *Discrete Network Design Problem* (DNDP) et nous appliquons ensuite l'algorithme génétique proposé initialement pour résoudre le DNDP, au problème de sélection des noeuds dans un réseau *mesh*. Les performances de l'algorithme génétique sont comparées à celles de deux algorithmes basés sur la théorie des graphes, W-GCP et *Recursive*.

Nous concluons cette thèse dans le chapitre 7 et donnons quelques perspectives de travail. Nous proposons d'évaluer les performances globales du réseau en termes de capacité et de durée de vie. Nous proposons également d'évaluer le changement de rôles de *clusterheads* sur les performances des algorithmes de routage.

Remarque :
Pour des raisons de cohérence, nous avons unifié les notations utilisées dans les différentes formules mathématiques. Sauf mention contraire, la signification de chaque notation est comme suit :

* $N(x)$: l'ensemble des noeuds voisins du noeud x.
* $\delta(x)$: le degré du noeud ; $\delta(x) = |N(x)|$.
* $d(x, y)$: la distance euclidienne entre les noeuds x et y exprimée en mètres.
* $h(x, y)$: la longueur du plus court chemin entre x et y.

Chapitre 2

Réseaux ad hoc : principes et caractérisation

Sommaire

2.1 Mode ad hoc : définition

Les systèmes de communication cellulaires sont basés essentiellement sur le mode infrastructure. Dans cette approche, les terminaux communiquent entre eux par l'intermédiaire de points d'accès appelés *Node B* dans le réseau UMTS. Les *Node Bs* sont contrôlés par des RNCs qui gèrent la répartition des ressources radio. Le routage et le transfert de

paquets dans le réseau coeur, s'effectuent par les commutateurs paquets (SGSN) et les passerelles vers le réseau extérieur (GGSN).

A l'inverse, les réseaux ad hoc sont des réseaux sans fil qui se forment spontanément et qui s'organisent automatiquement sans avoir besoin à une infrastructure existante. Ils sont capables de fournir des services de communication en dépit de fréquents changements de topologie et s'adapter aux différentes conditions de propagation et de trafic. Contrairement aux réseaux cellulaires, ce sont les hôtes eux mêmes qui assurent les fonctions de routage et de gestion de ressources.

Historiquement, les réseaux ad hoc ont été conçus dans les années 1970 pour une application militaire, dans le cadre du projet PRNet de l'agence DARPA du département de la Défense américaine [6]. Aujourd'hui, les réseaux ad hoc suscitent de plus en plus d'intérêts dans des circonstances caractérisées par une absence totale d'une infrastructure préexistante. Avec la baisse des coûts des équipements sans fil, ils permettent de mettre en oeuvre des solutions de communications pour plusieurs applications telles que les opérations de secours et les missions d'exploration.

Le mode ad hoc est un mode avantageux d'échange d'informations d'homologue à homologue ($peer-to-peer$) offrant plus de liberté aux terminaux. Le concept de communication multi-sauts permet de s'affranchir des problèmes de portées radio.

2.2 Réseaux ad hoc mesh et mobiles

Les MANETs et les WMNs sont deux types de réseaux ad hoc. Dans un MANET (*Mobile Ad hoc NETwork*), les noeuds sont mobiles. Les WMNs *(Wireless Mesh Networks)* sont des réseaux ad hoc maillés dans lesquels les noeuds formant la dorsale sont immobiles où à mobilité réduite.

Les MANETs et les WMNs partagent plusieurs points communs caractérisant de manière générale les réseaux ad hoc.

2.2.1 Points communs

Les réseaux ad hoc, *mesh* ou mobiles, possèdent plusieurs caractéristiques les différenciant des réseaux cellulaires classiques. Ces caractéristiques peuvent être soit des contraintes, soit des propriétés souhaitées.

2.2.1.1 Contraintes

- Topologie dynamique : la mobilité des terminaux et la versatilité des liens radio entraînent une topologie dynamique du réseau. Cette propriété constitue l'élément principal qui caractérise les réseaux ad hoc.

- Bande passante limitée : plusieurs communications sont actives simultanément ce qui pose un problème d'interférence et d'accès partagé au médium. Avec l'absence d'un contrôle centralisé, les ressources radio ne peuvent pas être gérées au préalable.

2.2.1.2 Propriétés souhaitées

- Auto-organisation : aucune entité centrale n'est nécessaire pour assurer le routage, la diffusion et la gestion des ressources radio. Les terminaux collaborent entre eux pour fournir les services réseaux et faire émerger, à partir d'interactions locales un comportement global.

- Adaptabilité : La structure s'adapte à l'environnement et réagit aux changements locaux. La réaction dynamique aux changements assure la robustesse du système face aux défaillances des noeuds et des liens dues à la mobilité et au manque d'énergie. Le système s'adapte et se reconstruit localement.

- Robustesse : Dans un réseau ad hoc, il n'y a pas de noeuds critiques. Les protocoles de communications sont implémentés dans l'ensemble des noeuds du réseau et sont basés sur des interactions locales. Donc, le système peut être réparé sans intervention extérieure.

- Passage à l'échelle : le système doit continuer à fonctionner efficacement même avec un grand nombre de noeuds. Le réseau ad hoc évite que le niveau de congestion soit proportionnel au nombre de noeuds.

2.2.2 Réseaux mesh

Pour les communications d'urgence, l'infrastructure à déployer doit être rapide à mettre en place et facile à configurer. Le réseau *mesh* est une nouvelle technologie sans fil émergente. Elle est la combinaison des réseaux ad hoc mobiles qui sont totalement dépourvus d'infrastructure et des réseaux cellulaires sans fil classiques qui sont complètement basés sur une infrastructure terrestre.

Les réseaux *mesh* sans fil forment un cas particulier des réseaux ad hoc. Ils ont tous les avantages des réseaux ad hoc : ils sont auto-configurables et auto-organisés, avec une topologie dynamique qui évolue au cours du temps avec la disparition ou l'apparition de nouveaux noeuds. Ils offrent aussi tous les avantages d'un réseau terrestre notamment en termes de débits disponibles grâce à l'organisation du réseau avec des noeuds à mobilité réduite, dédiés au routage et des noeuds clients mobiles [7].

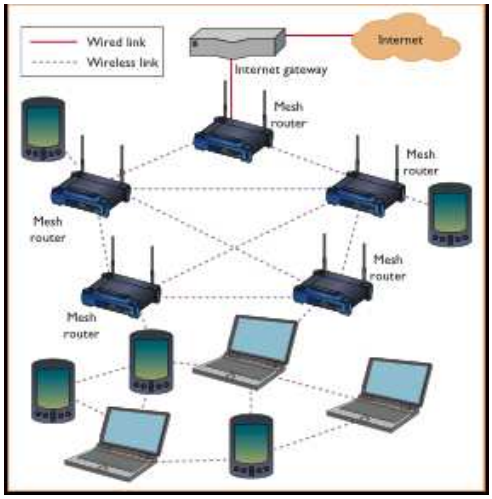


FIG. 2.1 – Exemple d'un réseau *mesh*.

L'architecture d'un WMN est composée de deux parties :

- Dorsale WMN : les routeurs *mesh* représentent la dorsale du WMN. Ils forment un réseau ad hoc et implémentent les fonctions de routage à multi-sauts dans un environnement sans fil. Ils peuvent être connectés via une passerelle (*gateway*) au monde IP extérieur.
- Clients WMN : les clients *mesh* communiquent entre eux et avec le monde extérieur en s'associant à un routeur *mesh*. Ils peuvent former entre eux un réseau ad hoc mobile.

Le groupe de travail TGs de l'IEEE a standardisé une norme appelée 802.11s pour améliorer les protocoles au niveau MAC et les adapter à un milieu sans fil maillé. Le TGs a défini un protocole de routage spécifique appelé HWMP (*Hybrid Wireless Network Protocol*) basé sur le protocole RM-AODV (*Radio Metric AODV*)[8].

Les réseaux *mesh* posent des problèmes inédits aux réseaux ad hoc mobiles notamment : quelle densité de routeurs *mesh* faut-il utiliser ? Où placer les routeurs passerelles ? Quelle

passerelle chaque noeud doit-il sélectionner pour communiquer vers le monde IP ?

Finalement, la différence entre un réseau ad hoc classique et un réseau *mesh* est la restriction des fonctionnalités de routage à un sous-ensemble du réseau formé par les routeurs *mesh*. Les WMNs diversifient les capacités d'un réseau ad hoc en introduisant une hiérarchie dans le réseau.

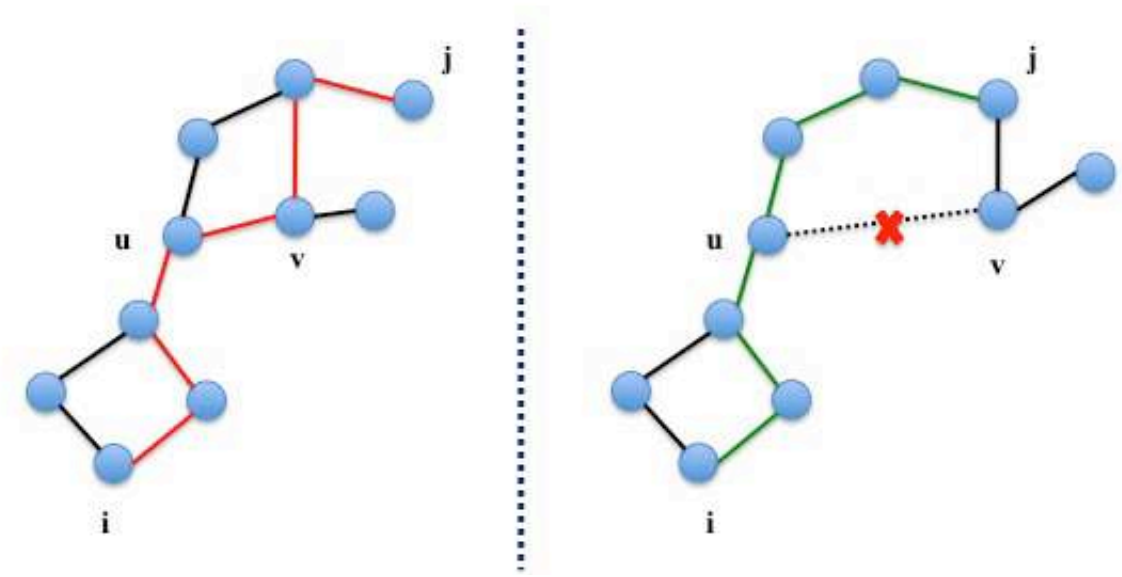
2.2.3 Réseaux ad hoc mobiles

La forte dynamique des réseaux ad hoc mobiles par rapport aux réseaux *mesh* introduit de nouveaux défis. La topologie du réseau est fortement variable à cause du changement fréquent des positions des noeuds qui peuvent se déplacer librement. Pour atteindre sa destination, un message passe par plusieurs noeuds relais. Donc les algorithmes de routage doivent s'adapter au changement de routes au fil du temps. Le mécanisme de maintenance de routes consiste à détecter les liens défaillants ; les routes qui utilisent ces liens deviennent invalides [9]. Dans la figure 2.2, à un instant t, le noeud i passe la route qui contient le lien (u, v) pour atteindre le noeud j. A un instant $t + \Delta t$, le lien (u, v) n'existe plus et la route précédemment établie devient invalide. L'algorithme de routage doit détecter cette défaillance et trouver une nouvelle route vers de i vers j.

Les applications ayant des différentes exigences de qualité de service sont de plus en plus nombreuses : la voix sur IP impose des contraintes sur les délais de transmissions et la vidéo à la demande sur la bande passante. Le support de la qualité de service dans le réseau exige la connaissance de la bande passante consommée au niveau MAC. Mais, la rupture de liens existants et la création de nouveaux liens de manière dynamique entraînent la versatilité de la qualité des liaisons radio. Un contrôle périodique est nécessaire afin de maintenir la qualité de service offerte aux flux acceptés. INSIGNIA [10] et BRuIT [11] sont deux protocoles de signalisation de la qualité de service utilisés les réseaux ad hoc mobiles.

2.3 Du concept théorique à la pratique

Mettre en oeuvre des réseaux ad hoc nécessite le développement de nouvelles techniques ou l'adaptation des protocoles implémentés dans les réseaux filaires et cellulaires. Nous présentons ici, trois techniques qui ont permis aux réseaux ad hoc de passer du concept théorique à des applications pratiques.



A un instant t A un instant $t + \Delta t$

FIG. 2.2 – Adaptation du routage à la mobilité des noeuds.

2.3.1 Routage

L'une des premières problématiques soulevées par les communications radio à multi-sauts est le problème de routage. L'absence d'infrastructure oblige les noeuds à assurer eux-mêmes le routage et l'acheminement des messages. Chaque terminal sert de relais en retransmettant un message qui ne lui est pas destiné, vers un autre terminal qui est hors de portée radio de l'émetteur initial de ce message. Les défis majeurs pour ce type de routage sont imposés par l'instabilité des chemins à cause de la mobilité des noeuds et les contraintes de ressources radio (bande passante limitée, débits variables et contraintes d'énergie).

L'une des questions soulevées est de savoir si on doit construire à l'avance un ensemble de routes entre chaque paire source - destination du réseau ou plutôt établir des routes uniquement vers les destinations demandées. Pour répondre à cette question, deux types de protocoles de routage ont été proposés : le routage proactif et le routage réactif. Si le premier type souffre de l'importance du volume du trafic de signalisation, le délai de construction et de reconstruction des routes limite la performance du deuxième. Le groupe

de travail MANET de l'IETF a retenu, en 2007, quatre protocoles de routage : deux protocoles proactifs (OLSR [12] et TBRPF [13]) et deux protocoles réactifs (AODV [9] et DSR [14]).

Les algorithmes de routage tels que OLSR et AODV utilisent le nombre de sauts comme métrique de routage. Or dans un environnement radio sensible aux interférences, cette métrique s'avère non optimale et peut créer des zones de congestion. Plusieurs métriques de routage alternatives ont été proposées :

1. ETX (Expected Transmission Count) : est basé sur l'estimation du nombre de transmissions, au niveau de la couche MAC, nécessaires pour transmettre avec succès un paquet sur un lien radio [15].
2. ETT (Expected Transmission Time) : est basé sur l'estimation du temps requis pour envoyer le paquet sur un lien. Il permet de prendre en considération la différence des débits de transmission [16].
3. MIC (Metric of Interference and Channel-switching) : prend en compte l'interférence inter-flux en calculant l'ETT et le nombre de noeuds qui peuvent être brouillés par la transmission sur un lien radio. Il prend en compte aussi l'interférence intra-flux en affectant un poids à un lien suivant qu'il utilise le même canal que le lien précédent ou pas [17].
4. CWB (Contention Window Based) : est composé de deux parties : la mesure du niveau de congestion *(Contention Window ou CW)* et la mesure du taux d'utilisation du canal [18]. Le paramètre CW est calculé en fonction du Frame Error Rate (FER) à la réception et de la période maximale de back-off avant la transmission dans la couche MAC. Le taux d'utilisation d'un canal est la fraction de temps où le canal est détecté comme occupé.

Les protocoles de routage assurent le transfert de messages de proche en proche sur des routes à plusieurs sauts. Comme les noeuds peuvent se déplacer librement, les liens radio se créent et se détruisent dynamiquement et fréquemment. Dans les réseaux ad hoc, les protocoles de routage s'adaptent à ces changements de routes.
Les réseaux ad hoc sont caractérisés par des ressources de capacité limitée et variable. Le routage s'adapte à ces variations en acheminant les messages vers les routes disposant des ressources nécessaires.

2.3.2 Contrôle de topologie

La notion de contrôle de topologie a été développée dans le contexte de réseaux soumis à des contraintes énergétiques. Le principe est de déterminer localement le rayon de trans-

mission minimal qui garde la connexité globale du réseau.

Dans la littérature, il existe plusieurs méthodes pour contrôler la topologie du réseau :

1. Structuration du réseau : la technique consiste à créer une hiérarchie dans le réseau. Une manière de mettre en place cette hiérarchie est de construire un ensemble dominant connecté. [19] propose de construire de manière distribuée le CDS (*Connected Dominating Set*) où les noeuds dominés ajustent leur puissance de transmission en fonction de leurs voisins dominants.
2. Réduction de graphe : les noeuds cherchent à éliminer les liens redondants. Dans l'algorithme CBTC *(Cone-Based Topology Control)* [20], chaque noeud v, sélectionne un sous-ensemble de voisins $\{w_1, w_2, ...w_k\}$. Il divise le disque de centre v en un ensemble de cônes d'arêtes vw_i avec un angle maximal de α. Il conserve un seul voisin par cône.
3. Élimination des voisins : l'élimination des voisins est l'approche adoptée par l'algorithme XTC présenté dans [21]. Chaque noeud classe ses voisins suivant la distance et la qualité du lien. Et en fonction du classement établi par ses voisins, il ajuste sa puissance de transmission pour ne garder qu'un sous-ensemble des noeuds voisins.

Le but principal du contrôle de topologie est transférer les messages sur des liaisons à faible puissance ce qui permet d'économiser de l'énergie et réduire les interférences.

2.3.3 Auto-organisation

L'organisation d'un système est la structuration du système en plusieurs entités pour faciliter leurs interactions. Donc l'auto-organisation fait référence à la collaboration de toutes les composantes du système pour établir et maintenir une structure qui répond à des besoins déterminés.

Un réseau ad hoc peut comporter des centaines mêmes des milliers de noeuds. Cette structuration dite "à plat" dégrade l'efficacité des protocoles de communication tels que le routage et la localisation des services. En effet, le volume de trafic de signalisation augmente exponentiellement avec la taille du réseau. Ce phénomène est connu sous le nom du problème des tempêtes de *broadcast*. Pour résoudre ce problème dans les réseaux ad hoc, les noeuds s'auto-organisent pour faire émerger à partir des interactions et des propriétés locales une structure globale et *virtuelle* du réseau. Cette structure virtuelle peut être formée par des dorsales (*backbones*) ou des *clusters*.

La fédération des noeuds autour d'une dorsale ad hoc est analogue à l'interconnexion des réseaux à travers des routeurs formant la dorsale Internet. Les noeuds de la dorsale servent

de relais pour la diffusion de messages de *broadcast*. Le contrôle du nombre de rediffusions réduit considérablement la redondance de transmission, la consommation d'énergie et la collision. Deux principales techniques issues de la théorie des graphe ont été utilisées pour la construction d'une dorsale dans les réseaux ad hoc : Connected Dominating Set ([22], [23] et [24]) et Minimum Spanning Tree ([25], [26] et [27]).

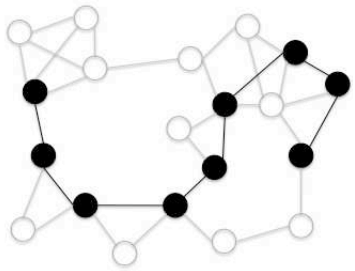


FIG. 2.3 – La construction d'un ensemble dominant connecté.

Nous traitons en détail la structuration du réseau en *clusters* dans le chapitre 3. Pour conclure, le mécanisme d'auto-organisation, permet au réseau de réagir aux changements et à la structure de s'adapter et se reconstruire localement. En effet, dans cette structure en *clusters* ou en dorsale, un changement local (disparition d'un noeud ou d'un lien) n'entraîne qu'une modification locale et n'impacte pas la structure sur son intégralité.

2.4 Communications d'urgence

Dans les situations d'urgence, le réseau déployé doit être disponible à tout instant. Les équipes de secours doivent recevoir en temps réel les instructions et les informations relatives à l'état du terrain, afin de mener à bien leur mission. Plusieurs projets ont été développés pour améliorer les moyens de communications mis à disposition des populations et des équipes de secours.

2.4.1 Exemple de projets

En passant de l'ouragan Katrina qui a frappé en 2005 les États-Unis au séisme de 2011 survenu au Japon, le monde doit faire face à de multiples dangers. Les catastrophes, naturelles ou induites par l'homme ont des conséquences sur, non seulement les vies humaines mais aussi l'infrastructure, notamment les réseaux de télécommunications qui peuvent être totalement endommagés. Or, l'accès à l'information et aux moyens de communica-

tions dans la zone touchée par la catastrophe aide à atténuer ces conséquences.

Plusieurs actions ont été initiées par les gouvernements pour développer de nouvelles solutions qui permettent de réduire la vulnérabilité des populations. Les télécommunications surviennent dans toutes les étapes de prévention et de gestion de catastrophe. La réglementation pourrait empêcher la mise en place rapide de systèmes de communication d'urgence, notamment l'obligation de licence d'utilisation des fréquences attribuées. Les obstacles d'ordre réglementaire ont été en partie résolus grâce à la convention de Tampere. Cette convention est un traité international entré en vigueur en 2005 qui régit la fourniture et la disponibilité des équipements de télécommunications lors des opérations de secours.

2.4.1.1 WISECOM

Les premières 24 heures qui suivent une catastrophe sont déterminantes pour les opérations de secours et de sauvetage. L'acheminement des aides et la coopération des équipes nécessitent des moyens de communications fiables et rapides. WISECOM[1] [28] est un projet européen qui s'intéresse au développement d'équipements de communications qui assurent la couverture de la zone sinistrée dès les premières heures suivant la catastrophe. Ce projet a été initié en 2006 et sa démonstration finale a eu lieu en 2008.

Le but de WISECOM est de développer un système qui intègre les différentes technologies radio de type GSM, UMTS, WiFi et WiMax permettant aux sinistrés et aux équipes de secours d'utiliser directement leurs téléphones et leurs terminaux. L'interconnexion avec l'infrastructure encore opérationnelle est assurée grâce à des liaisons satellite. La liaison satellite utilise le système Inmarsat BGAN (*Broadband Global Area Network*). La solution proposée est destinée à répondre aux besoins immédiats durant les premières heures qui suivent la catastrophe, en attendant qu'une infrastructure plus complexe et plus lourde soit mise en place.

2.4.1.2 CHORIST

Les forces de sécurité publique utilisent des réseaux de communications publics appelés PMRs (*Professional Mobile Radio*) qui offrent entre autres des services d'appels de groupe. CHORIST[2] [29] est un projet financé par la Commission Européenne qui modernise les réseaux PMRs en proposant une nouvelle architecture. CHORIST est un projet de 38 mois (juin 2006 - juillet 2009).

[1] *Wireless Infrastructure over Satellite for Emergency COMmunications*
[2] *integrating Communications for enHanced envirOnmental RISk management and citizens safeTy*

L'architecture proposée comprend trois modules :

Module 1 : est responsable de la collecte d'informations sur la zone sinistrée, l'évaluation des risques et la diffusion des messages d'alerte aux autorités. Les informations collectées peuvent provenir des capteurs installés ou des populations à travers les centres d'appels d'urgence.

Module 2 : est responsable de la diffusion des messages d'alerte par les autorités aux populations concernées par la catastrophe. Plusieurs moyens de communications peuvent être utilisés : réseaux téléphoniques, diffusion audio numérique (*Digital Audio Broadcasting* ou DAB) et diffusion vidéo numérique (*Digital Video Broadcasting* ou DVB).

Module 3 : constitue l'infrastructure de télécommunications déployée dans la zone sinistrée. Le système intègre deux technologies complémentaires : les réseaux *mesh* et les réseaux TETRA[3]. L'architecture proposée offre de nouveaux services tels que la vidéo et le transfert de fichiers qui procurent aux équipes de secours de nouveaux types de données tels que les images satellite et les cartes topographiques.

Les projets décrits dans le paragraphe précédent proposent d'intégrer différentes technologies pour construire de nouvelles architectures. Le projet WISECOM propose d'utiliser le satellite comme moyen d'interconnexion entre les zones sinistrées et les centres de contrôle (headquarters). CHORIST propose d'introduire les réseaux *mesh* pour offrir de nouveaux services à travers les réseaux PMR. A l'inverse, nous proposons une vision alternative des réseaux ad hoc et du satellite, en considérant les réseaux ad hoc comme des réseaux autonômes (*stand alone*), où le satellite intervient pour résoudre certains problèmes dont peut souffrir une telle architecture. Or, comme tout réseau de télécommunications, les ressources disponibles peuvent être insuffisantes pour garantir une bonne qualité de service à tout instant. Les deux principaux incidents qui peuvent causer l'indisponibilité du réseau sont la congestion et le partitionnement.

2.4.2 Congestion

2.4.2.1 Définition

Le premier problème concerne la congestion des liens. A un instant t, le graphe représentant le réseau peut être connecté, c'est à dire qu'il existe au moins un chemin entre toute paire de sommets. Mais l'étude des propriétés d'un réseau est loin d'être réduite à l'étude d'un simple graphe. L'existence d'un chemin sur le graphe entre toute paire de sommets ne

[3]TETRA (*TErrestrial Trunked RAdio*) est un standard ETSI pour les réseaux PMR.

garantit pas la disponibilité du service. Un lien radio est doté d'un débit maximal qui dépend de la technologie utilisée. Par exemple, la norme 802.11g prévoit un débit maximal de 54 Mbit/s sur un lien radio. A un instant donné, la demande en termes de volume de trafic peut être très élevée et peut dépasser la capacité disponible.
La figure 2.4(a) montre un cas où il existe au moins un chemin entre toute paire du réseau. En revanche, le lien entre i et j est l'unique lien entre les deux parties du réseau. Tout le trafic dont la source est située dans la partie 1 du réseau et dont la destination est située dans la partie 2 du réseau transite forcément par le lien (i,j). Par conséquent, si le volume de trafic entre les deux parties du réseau est au dessus de la capacité disponible, un goulot d'étranglement apparaît au niveau de ce lien.

En cas de congestion, les paquets subissent de longs délais de transmission et peuvent même être rejetés, après plusieurs tentatives de retransmission manquées, ce qui entraîne la dégradation des débits disponibles. La congestion peut avoir lieu quand le volume du trafic entrant dépasse les ressources disponibles et aussi quand les collisions entre les paquets deviennent fréquentes. Les paquets rejetés ont des effets très négatifs sur les performances du réseau car dans le cas d'une retransmission du paquet perdu, une énergie supplémentaire est requise et un délai supplémentaire est ajouté.

La congestion, dans les réseaux ad hoc, dégrade les débits disponibles. Elle cause aussi la perte de l'énergie, à cause du grand nombre de retransmissions. Une estimation du niveau de congestion est nécessaire pour caractériser les ressources disponibles dans le réseau. Cette opération dans un milieu radio partagé est plus difficile que dans un milieu filaire à cause de la variation de la capacité en fonction du temps et de l'ordre non déterministe de la transmission des noeuds.
L'estimation du niveau de congestion revient à estimer les ressources disponibles ou bien quantifier le taux d'utilisation de ces ressources.

2.4.2.2 Mesure de la charge du noeud

Les transmissions manquées conduisent à une accumulation au niveau des files d'attente. Étant donné que la taille des files d'attente est finie, les transmissions manquées causent l'augmentation du nombre de paquets perdus et par conséquent à une surcharge du canal. Kang [30] montre que l'utilisation simultanée de trois métriques : la charge du canal, le taux d'abandon des paquets et le taux de remplissage des buffers, donne des résultats corrects, en dépit de l'hétérogénéité des équipements et des protocoles MAC implémentés.

2.4.2.2.1 Mesure de la charge du canal La technique la plus utilisée est d'échantillonner le canal de manière périodique et utiliser une moyenne pondérée pour avoir des mesures

lisses. On note C_i^{avg} la charge moyenne du canal après l'échantillonnage numéro i. On note $C_{i+1}^{sampled}$ la valeur de l'$(i+1)^{\text{ème}}$ échantillon. Soit α le poids de la valeur de l'échantillon le plus récent (souvent appelé facteur d'oubli). On a donc la valeur de la charge moyenne du canal à l'instant $i+1$:

$$C_{i+1}^{avg} = (1-\alpha) \times C_i^{avg} + \alpha \times C_{i+1}^{sampled}.$$

La fréquence d'échantillonnage peut être fixe ; ceci est efficace dans le cas où les noeuds n'ont pas de contraintes énergétiques.

2.4.2.2.2 Mesure du taux d'abandon des paquets Pour mesurer le taux d'abandon des paquets, une période fixe appelée *époque* est définie. Lors de chaque époque, les statistiques sur le nombre de paquets arrivés et de paquets abandonnés sont maintenues. Afin d'économiser les temps de calcul, la mesure du taux d'abondan est réalisée uniquement à l'arrivée d'un nouveau paquet dans la file d'attente ou à l'abandon d'un paquet.

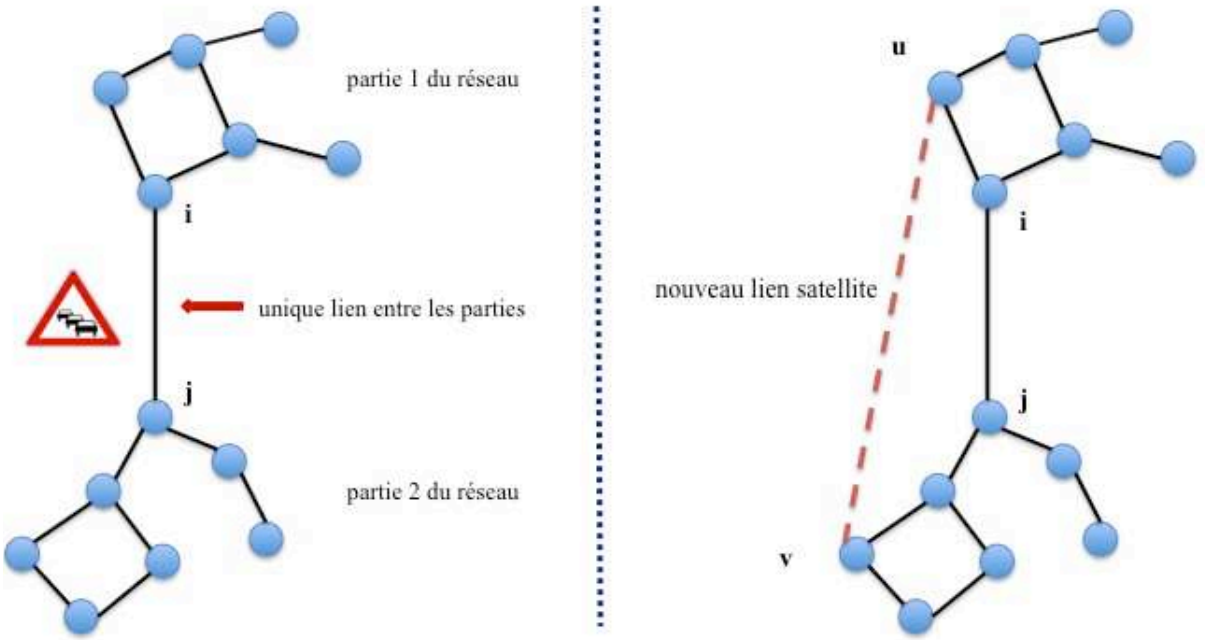
2.4.2.2.3 Mesure du taux de remplissage des buffers De la même manière, la mesure du taux de remplissage des buffers est réalisée uniquement quand un paquet est inséré ou retiré de la file d'attente. Cette méthode permet d'adapter la fréquence d'échantillonnage en fonction du volume de trafic entrant.

2.4.2.3 Estimation de la bande passante résiduelle

La différence entre la bande passante maximale du lien et sa bande passante actuelle représente la demande supplémentaire en termes de trafic qui peut être encore satisfaite par le noeud sous les conditions actuelles. Cette différence définit « la bande passante résiduelle ». Le concept de bande passante résiduelle considère la surcharge due à l'accès partagé au médium au niveau de la couche de liaison du modèle OSI (*Open Systems Interconnection*).

L'estimation des ressources disponibles a été développée dans la littérature pour des problématiques de gestion de la qualité de service (QoS) et de routage. On peut distinguer deux catégories de solutions proposées : les techniques intrusives et les techniques passives. Les techniques intrusives se basent sur l'envoi de paquets de contrôle pour explorer les caractéristiques du lien radio [31]. L'insertion de ces paquets sondes introduit un *overhead* supplémentaire et consomme de la bande passante. Avec l'approche passive aucun paquet n'est introduit dans le réseau.

Atalay [32] propose un modèle analytique qui permet d'estimer la bande passante résiduelle d'un lien radio et qui prend en considération la dynamique du canal radio et le changement du volume de trafic. Le modèle proposé présente une analyse de la capacité des liens radio sous différentes conditions du réseau : les débits des liens, la taille des paquets, les évanouissement du canal radio et les noeuds cachés.
Une solution envisageable à ce problème est l'ajout d'un lien satellite qui va alléger la charge du lien (i,j). Le trafic entre les deux parties du réseau va être partagé entre ce lien et le lien satellite. Schématiquement, on peut représenter le nouveau lien à créer par satellite comme une nouvelle arête entre 2 sommets u et v (Fig. 2.4(b)).



(a) (b)

FIG. 2.4 – La création d'un goulot d'étranglement (a), création d'un lien satellite pour la répartition de charge (b).

2.4.3 Partitionnement du réseau

2.4.3.1 Définition

Dans les MANETs, les noeuds sont mobiles. Le graphe représentant le réseau à instant $t + \Delta t$ peut être différent du graphe représentant le réseau à instant t. Si nous prenons

par exemple, le réseau représenté par la figure 2.5, les noeuds i et j sont mobiles et en se déplaçant, ils s'éloignent l'un de l'autre. Or le lien (i,j) est le seul lien qui relie les deux parties du réseau. Avec la perte de ce lien, aucune communication n'est possible entre un noeud de la partie 1 du réseau et la partie 2 du réseau. On parle alors de partitionnement du réseau.

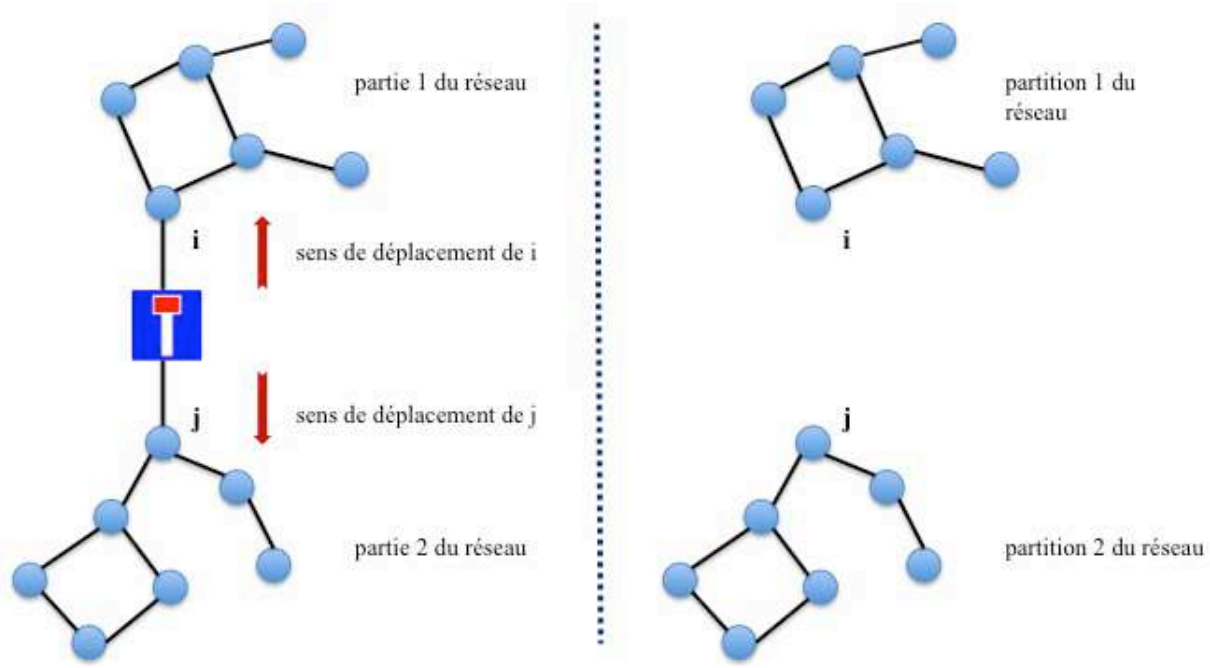


FIG. 2.5 – Le partitionnement du réseau à cause de la mobilité des noeuds.

Le partitionnement peut se reproduire durant le régime de fonctionnement du réseau, comme c'est le cas dans les MANETs. On peut aussi envisager un scénario où le réseau est partitionné dès la phase de déploiement : le service de communications est déployé sur plusieurs sites éloignés.

Certes les réseaux ad hoc sont auto-organisés grâce à la collaboration des noeuds. Mais dans le cas de partitionnement du réseau, les techniques classiques tels que les trajets disjoints ne sont plus valables car aucun chemin n'existe entre les deux partitions. Les noeuds d'une partition ne peuvent pas joindre les noeuds d'une autre partition et les communications sont interrompues. Or, dans les situations d'urgence et de secours, une telle interruption est intolérable car elle peut mettre en péril la vie des équipes de secours. Une solution envisageable est de relier les partitions grâce à un lien satellite dont la dis-

ponibilité n'est pas perturbée par les catastrophes naturelles et les incidents industriels, comme le montre la figure 2.6.

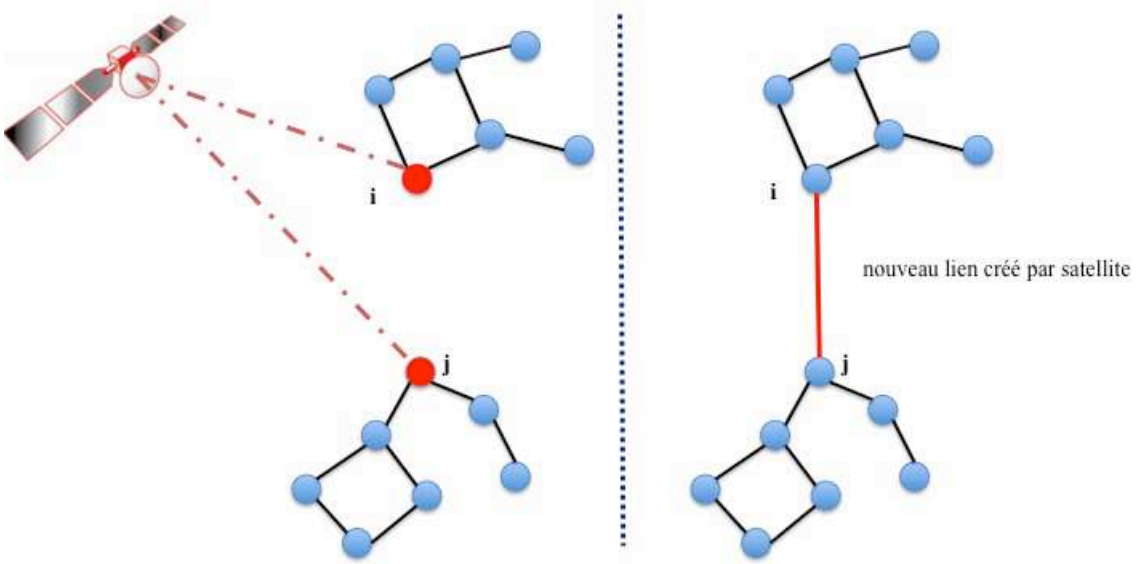


FIG. 2.6 – La réparation de la connexité du réseau grâce à un lien satellite.

Dans le cadre de cette thèse, on va plus s'intéresser à l'étude de la réparation par satellite de la connexité d'un réseau ad hoc. Par conséquent, nous allons traiter plus en détails ce problème et les solutions envisagées pour le résoudre.

2.4.3.2 Détection du partitionnement du réseau

H. Ritter et al. [33] ont développé deux approches, une distribuée et l'autre centralisée. Les noeuds périphériques jouent un rôle primordial dans la détection du partitionnement du réseau. L'heuristique est basée sur le principe d'échange d'informations entre ces noeuds. Si un noeud périphérique u ne peut plus joindre un autre noeud périphérique v alors

le réseau est partitionné. Le mécanisme de détection du partitionnement du réseau est complété par un mécanisme de surveillance locale pour éviter toute fausse détection qui peut être engendrée par l'échec d'un noeud périphérique.

2.4.3.2.1 Concept du noeud actif On définit les noeuds actifs comme les noeuds périphériques situés à la bordure du réseau. Ces noeuds échangent régulièrement des signaux sondes et chaque noeud actif u surveille un certain nombre d'autres noeuds actifs A_u. Si le *timer* associé à un noeud appartenant à A_u expire avant que le noeud u ne reçoive un signal pilote, le noeud u suspecte le partitionnement du réseau.

Afin d'éviter toute fausse détection, chaque noeud actif u sélectionne parmi ses voisins un noeud particulier appelé *camarade*. Si le noeud camarade n'entend plus son noeud actif, il le sollicite par l'envoi d'une requête de route (RREQ). Si le noeud u se trouve à plusieurs sauts de son camarade, il sélectionne un nouveau camarade. Si le noeud u ne répond pas, son camarade suspecte son échec et envoie une notification aux autres noeuds actifs A_u. Par conséquent, la probabilité d'une fausse alerte diminue.

2.4.3.2.2 Approche centralisée Dans cette approche, un seul noeud actif appelé source, envoie périodiquement des messages *beacon* à destination des autres noeuds actifs qui, à leur tour envoient des messages de notification au noeud source pour l'informer de leur existence. Par conséquent, tout le réseau entre le noeud source et les autres noeuds de frontière (actifs) est contrôlé.

Si un noeud actif ne reçoit pas un message *beacon* de la part du noeud source, il peut déduire que le réseau est partitionné. Pour valider cette déduction, on fait appel au mécanisme de validation locale, décrit brièvement dans le paragraphe précédent. Uniquement le noeud source sélectionne un camarade. Le noeud camarade envoie périodiquement un message *ping* au noeud source qui doit répondre par un accusé de réception. Le noeud camarade possède une copie de la liste des noeuds actifs. Ainsi, il est capable de leur envoyer une notification d'échec du noeud source.

L'approche centralisée présente un inconvénient majeur. En cas de partitionnement du réseau, la partition qui contient le noeud source ne détecte pas le partitionnement.

2.4.3.2.3 Approche distribuée L'approche distribuée est dans son mode de fonctionnement, similaire à l'approche centralisée. La seule différence est que chaque noeud actif possède un camarade et envoie des messages *beacon*.

Chaque noeud sauvegarde les identifiants d'un sous-ensemble de noeuds actifs dans le réseau appelés partenaires, ainsi que la distance en nombre de sauts vers ces noeuds. Chaque noeud, en mettant à jour sa table de partenaires, garde toujours les adresses des noeuds actifs les plus éloignés, afin de couvrir une partie plus large du réseau. L'échange de messages *beacon* a lieu uniquement entre le noeud actif et ses partenaires.

En cas d'échec d'un noeud actif u, ses partenaires sont informés par son camarade. Les noeuds partenaires arrêtent alors d'envoyer des messages *beacon* au noeud u. Si le noeud perd un grand nombre de partenaires à cause de leur échec où à cause du partitionnement du réseau, alors il initie une procédure de recherche de noeuds actifs.

La figure 2.7 montre la structure du réseau avec les noeuds participant au mécanisme de détection du partitionnement du réseau dans le cas de l'approche centralisée. L'approche distribuée est plus robuste et plus stable que l'approche centralisée. Mais ceci a un coût : l'*overhead* introduit par l'échange de messages.

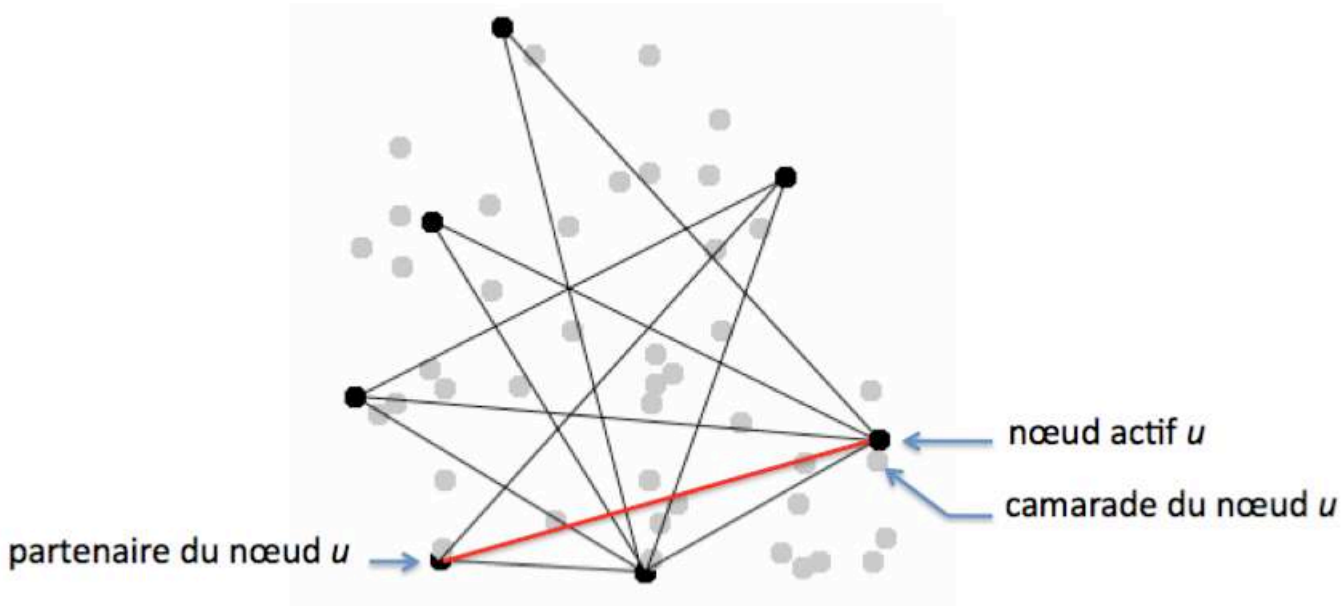


FIG. 2.7 – Les noeuds participant au mécanisme de détection du partitionnement du réseau - approche distribuée.

2.4.3.3 Prédiction du partitionnement

Le mécanisme introduit dans la partie précédente permet de détecter le partitionnement du réseau. Mais on peut aussi envisager un mécanisme qui permet de prédire le partitionnement.
Derhab [34] propose une solution basée sur l'algorithme de routage TORA (*Temporally Ordered Routing Algorithm*) pour prédire le partitionnement du réseau. Son approche a pour application la prédiction de la perte de communication, à cause du partitionnement, entre un noeud et un serveur d'applications. Le but est de faire une réplication du service et désigner dans chaque partition un nouveau serveur.

2.4.3.3.1 Durée de vie résiduelle d'un lien Dans cette approche, on a besoin de distinguer les liens robustes des liens fragiles. Chaque noeud envoie de manière périodique à ses voisins des messages *beacon*. On note $S_{ij}^k(t)$ la puissance du $k^{\text{ème}}$ *beacon* envoyé par le noeud j et reçu à l'instant t par le noeud i. Chaque noeud garde la trace des l dernières valeurs de $S_{ij}(t)$. On note $V_{ij}^k(t)$ le taux du changement de la puissance du signal *beacon*. La durée de l'intervalle de temps entre la réception de deux messages *beacon* successives est noté $(t - t_{prev})$. On a donc :

$$V_{ij}^k(t) = \frac{S_{ij}^k(t) - S_{ij}^{k-1}(t_{prev})}{t - t_{prev}}.$$

La durée de vie résiduelle du lien (i, j) est notée $\xi_{ij}(t)$. Le noeud i estime qu'il va perdre le lien (i, j) après une période temps égale $\xi_{ij}(t)$. La puissance du signal seuil en dessous de laquelle le lien radio (i, j) n'existe plus est notée S_R. On a donc :

$$\xi_{ij}(t) = \frac{S_{ij}^k(t) - S_R}{\frac{1}{l-1}\sum_{k=m-l+1}^{m} |V_{ij}^k(t)|},$$

où ξ_{th} définit le seuil en dessous duquel le lien (i, j) qui est considéré fragile. La figure 2.8 montre le cycle de vie d'un lien.

2.4.3.3.2 Perte de liens downstream Chaque noeud a besoin d'établir vers son serveur d'applications un chemin auquel est associé un poids noté H_i. Pour chaque noeud j appartenant au voisinage de i, on compare H_i et H_j. Si $H_i < H_j$, alors le lien (i, j) est marqué *upstream* et *downstream* sinon.

Quand un noeud ordinaire i perd son dernier lien *downstream* robuste en direction du serveur actif, il initie l'algorithme de prédiction du partitionnement du réseau. En cas d'échec de la connexion avec le serveur actif, un serveur passif qui sauvegarde une copie

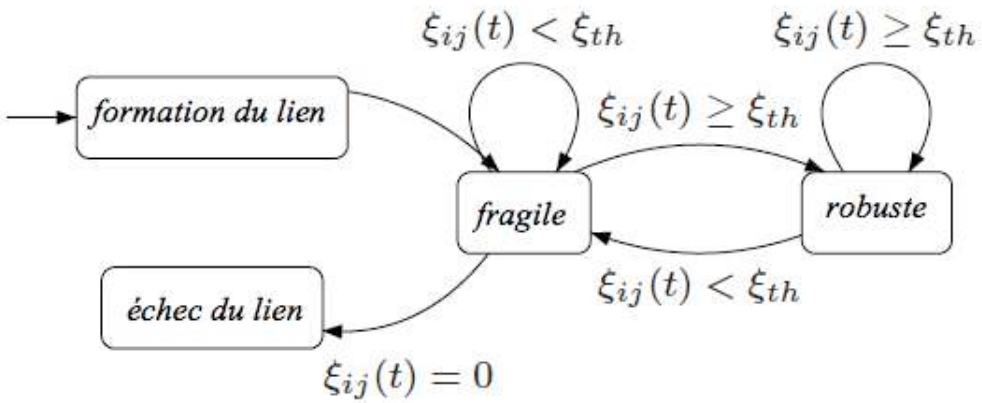


FIG. 2.8 – Le cycle de vie d'un lien [34].

des informations nécessaires pour démarrer un service, prend le relais. Si le noeud a au moins un chemin robuste vers un serveur passif, il lui envoie une requête d'affiliation et ce serveur passif deviendra actif. Si aucun chemin robuste n'existe entre le noeud i et un serveur passif, alors le noeud i se déclare comme serveur passif.

L'approche basée sur la perte de liens *downstream* vers un serveur d'applications peut être généralisée pour s'appliquer à la perte de liens critiques. En effet, la perte de liens critiques du réseau engendre le partitionnement et la création de groupes isolés.

Zhang [35] propose un autre mécanisme de prédiction de partitionnement du réseau. Leur approche est basée sur le modèle de mobilité RRGM (*Reference Region Group Mobility*), décrit dans le paragraphe et l'algorithme de *clustering* DGMA (*Distributed Group Mobility Adaptive Clustering Algorithm*), décrit dans le paragraphe . Leur procédure de prédiction utilise aussi la notion de durée de vie résiduelle d'un lien. Mais leur lien est un lien fictif. En effet, les noeuds sont groupés par petits groupe et suivent le mouvement d'un point fictif appelé point de référence du groupe. Donc le lien considéré dans l'algorithme de Zhang relie les points de références de deux groupes distincts. Si le lien entre ces deux points fictifs disparaît, on suppose que les deux groupes ne peuvent plus communiquer donc le réseau est partitionné. Cette approche est intéressante mais le problème c'est qu'elle ne peut pas être généralisée à n'importe quel modèle de mobilité. En effet, elle exploite les particularités du modèle de mobilité pour développer la méthode de prédiction.

2.5 Conclusion

Pour conclure, plusieurs projets tels que WISECOM et CHORIST ont été initiés pour améliorer l'efficacité des équipes de secours et offrir des moyens de télécommunications dans les zones sinistrées. Nous proposons dans le cadre de cette thèse d'étudier des réseaux ad hoc en mode 'stand alone', c'est à dire le réseau étudié est déployé dans une zone dépourvue de tout moyen de communication et n'est pas relié à l'infrastructure encore fonctionnelle.

Dans ce type d'architecture, l'indisponibilité des ressources peut être causée essentiellement par la congestion et le partitionnement. Le point commun dans les deux situations est qu'un sous-ensemble de noeuds pourrait accéder au satellite. La différence réside dans la prise de décision d'activer la liaison satellite et le choix des points d'accès au satellite.

Avec ce chapitre introductif, nous avons présenté les caractéristiques des réseaux ad hoc mesh et mobiles et les problématiques liés à leurs déploiement pour les communications d'urgence, notamment le problème du partitionnement. L'utilisation du satellite pour la restauration de la connexité du réseau soulève le problème du choix des noeuds qui vont avoir accès au segment spatial. Le suivant chapitre discute les principales approches permettant de structurer le réseau ad hoc mobile, leur adéquation à la problématique du choix des noeuds et les aspects techniques liés à leur application dans un scénario de lutte contre les feux de forêt.

Chapitre 3

Structuration des réseaux ad hoc mobiles

Sommaire

Dans ce chapitre, nous discutons les aspects techniques relatifs à l'intégration du segment satellite à un réseau ad hoc mobile. Cette intégration passe tout d'abord par la détection de la défaillance du système avant d'activer le lien satellite. Pour activer ce lien, uniquement

un sous-ensemble de noeuds va avoir accès au satellite. Nous nous intéressons alors aux approches développées qui permettent d'attribuer des fonctionnalités supplémentaires à des noeuds particuliers. De la structuration du réseau sous forme d'un ensemble dominant connecté à la sélection d'un ensemble dominant indépendant, nous analysons les propriétés des algorithmes proposés et discutons leur adéquation à notre scénario choisi. Mais avant de décider quelle est la meilleure approche à appliquer, nous commençons tout d'abord par analyser le problème afin de comprendre ses enjeux.

3.1 Analyse du problème

L'intégration du segment spatial à un réseau ad hoc mobile repose sur trois étapes. Premièrement, les noeuds doivent détecter/prédire le partitionnement du réseau et prendre la décision d'activer les liens satellite. Une fois cette décision est prise, il faut choisir les noeuds qui vont fournir de la connexité dans leurs partitions. Finalement, grâce à la dynamique de la topologie et l'apparition de nouveaux liens, la connexité du réseau peut être rétablie. La connexion via le satellite doit être arrêtée.

3.1.1 Détection de la défaillance du réseau

La topologie du réseau est dynamique avec de nouveaux noeuds qui apparaissent et viennent se connecter au réseau et d'autres qui disparaissent. La mobilité des noeuds entraîne aussi la rupture de liens existants et la création de nouveaux liens. Les réseaux ad hoc sont par définition auto-configurables et auto-réparables. Cette capacité se traduit par l'adaptation des algorithmes de routage à une certaine dynamique du réseau. En cas d'invalidité d'une route, ils sont capables de détecter ce dysfonctionnement et établir d'autres chemins qui relient la source à la destination. Mais si toutes les routes sont coupées entre la source et la destination et aucun chemin n'est disponible, la propriété d'auto-réparation des algorithmes de routage n'a plus d'utilité. Ils annoncent un *Route Error* et abandonnent la requête au bout d'un certain temps.

Le changement de la topologie peut entraîner la formation de plusieurs partitions et aucun lien radio n'est plus disponible pour assurer l'interconnexion entre les différentes parties du réseau. Les noeuds doivent alors prédire/détecter ce changement de topologie et prendre la décision d'activer une connexion satellite pour maintenir la connexité. Un aperçu de quelques méthodes de prédiction et de détection du partitionnement est donné dans la section 2.4.3.

3.1.2 Choix des noeuds communiquant avec le satellite

On suppose que tous les noeuds ont la capacité d'utiliser le segment spatial. Mais afin d'optimiser l'utilisation des ressources du réseau terrestre et du satellite, seul un sous-ensemble de noeuds assurera l'interconnexion entre les différentes partitions.
On prend comme exemple le réseau illustré dans la figure 3.1. On suppose que les noeuds i et j sont choisis pour communiquer avec le satellite dans les partitions A et B respectivement. Chaque source de la partition A souhaitant communiquer avec une destination de la partition B, envoie son trafic vers le noeud i. Les données transitent ensuite par le satellite pour arriver jusqu'au noeud j, qui à son tour les transmet à leur destination. Tout le trafic entre les deux partitions passe par les noeuds i et j. Ils peuvent, en fonction du volume de trafic, représenter des goulots d'étranglement. Le choix des noeuds communiquant avec le satellite n'est donc pas trivial car il détermine les performances du réseau (durée de vie, délais de transmission et capacité). Dans le contexte d'un réseau ad hoc mobile, la procédure du choix des noeuds doit donc satisfaire les propriétés suivantes :

1. Minimisation du nombre de noeuds communiquant avec le satellite : le but est d'optimiser l'utilisation des ressources du réseau terrestre et du satellite.
2. Focalisation sur les propriétés intrinsèques des noeuds : les noeuds choisis doivent supporter la charge de trafic supplémentaire. La procédure doit donc tenir en compte l'hétérogénéité des propriétés des noeuds (mobilité, énergie, etc.).

3. Réduction de la signalisation : le volume de trafic engendré par les messages de contrôle et de signalisation affecte considérablement les performances du réseau. Si la vision globale du réseau permet de mieux optimiser les critères de choix, elle nécessite l'échange d'informations et l'interaction entre tous les noeuds du réseau. La stratégie locale réduit le coût protocolaire et supporte mieux le passage à l'échelle.

4. Émergence d'un comportement global à partir des informations locales : tous les noeuds contribuent à la procédure du choix et chaque noeud prend une décision tout en ayant une vision partielle du réseau. Des règles d'interactions et de décisions locales (entre les noeuds voisins) définissent le comportement global du réseau.

5. Adaptation à la dynamique du réseau : les noeuds doivent s'adapter aux changements topologiques et à l'évolution des propriétés des noeuds. Chaque noeud surveille son voisinage et réagit aux nouvelles informations (arrivée/défaillance d'un voisin).

6. Cohérence : les situations d'incohérence peuvent être engendrées par la dynamique du réseau. Elles peuvent aussi être dues au caractère local des informations disponibles. Les règles d'interaction et de décision doivent faire émerger un comportement global cohérent, où les conflits potentiels sont détectés et résolus.

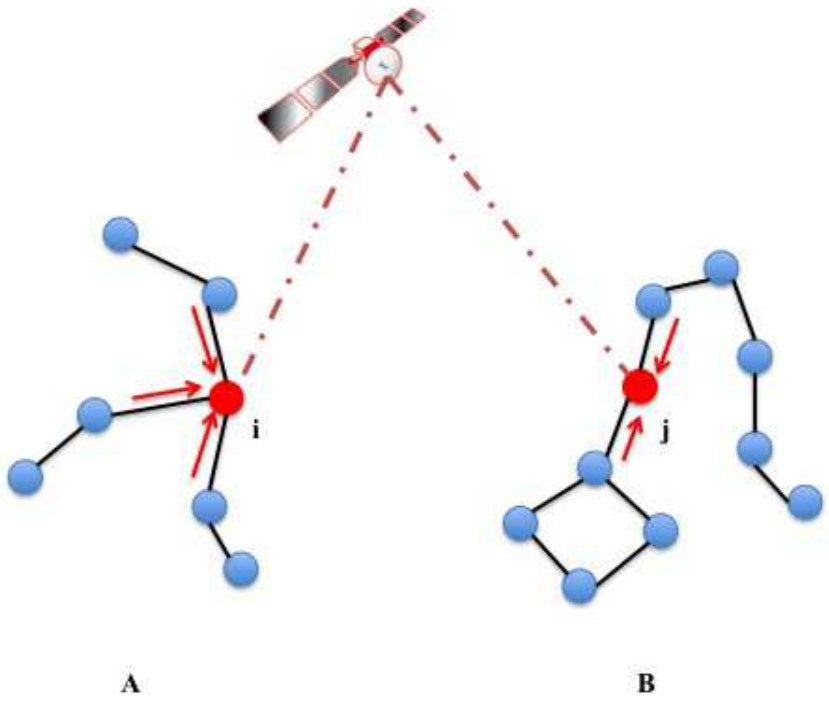


FIG. 3.1 – Exemple de choix de noeuds qui ont accès au segment satellite.

Le but est de trouver une méthode qui satisfait au mieux les propriétés citées ci-dessus. Nous avons passé en revue les méthodes existantes qui permettent de structurer les réseaux ad hoc et attribuer des fonctionnalités supplémentaires à des noeuds particuliers, afin de trouver celle qui correspond au mieux à notre problématique.

Le *Gateway Placement Problem* est une problématique des réseaux *mesh* qui consiste à choisir parmi les noeuds du réseau, des passerelles assurant l'interconnexion avec le monde IP extérieur. Les solutions proposées permettent de garantir une certaine qualité de service en optimisant la charge des noeuds et les délais de transmission. Mais dans les réseaux *mesh*, les noeuds sont quasi-statiques et le déploiement est généralement précédé par une phase de planification. En effet, leur principale application est l'extension de la

couverture réseau aux zones blanches où le déploiement d'une infrastructure terrestre est très coûteux. La résolution du *Gateway Placement Problem* intervient lors de cette phase de planification et suppose la connaissance de la topologie de tout le réseau [36][37][38]. Dans un MANET, les noeuds sont très dynamiques, donc les algorithmes proposés pour le *Gateway Placement Problem* ne conviennent pas à notre problème. Nous allons étudier dans les deux sections suivantes l'organisation des réseaux MANETs. Cette organisation repose sur deux types de structures virtuelles : les dorsales et les *clusters*. Les noeuds choisis pour former la dorsale D du réseau forment un ensemble dominant connecté : toute communication entre deux noeuds de D passe uniquement par des noeuds relais appartenant à D. Cette propriété de connexité n'est pas exigée quand on forme des ensembles dominants indépendants ou *clusters*.

3.1.3 Arrêt de la connexion

C'est un problème qui ressemble au premier problème évoqué précédemment : la détection de la défaillance du réseau. Les noeuds, en se déplaçant, peuvent rétablir la connexité du réseau. Par conséquent, une procédure automatique d'arrêt de la connexion avec le satellite doit se déclencher pour libérer la bande passante précédemment allouée. La détection du rétablissement de la connexité du réseau ne sera pas abordée dans le cadre de cette thèse.

3.2 Ensemble dominant connecté

Afin de construire les tables de routage et les mettre à jour, les protocoles de routage utilisent un mécanisme de découverte de route. Dans la plus grande partie des protocoles tels que AODV [9] et DSR [14], ce mécanisme est initié par un processus d'inondation aveugle où chaque noeud relaye tous les messages de contrôle qu'il reçoit. L'inondation permet aux messages de contrôle d'atteindre l'ensemble des noeuds du réseau mais elle consomme de la bande passante et risque de dégrader considérablement les performances du réseau. Ce phénomène est connu sous le nom du problème des tempêtes de *broadcast*. La formation d'une dorsale permet de structurer le réseau et collecter le trafic de contrôle de manière plus efficace. On présente dans cette section, la technique basée sur la recherche de l'ensemble dominant connecté.

La recherche d'un ensemble dominant connecté (CDS pour *Connected Dominating Set*) pour un graphe $G = (V, E)$ consiste à trouver un sous-ensemble D de V tel que tous les sommets qui ne sont pas dans D soient voisins à k ($k \geq 1$) sauts d'un sommet de D. Les noeuds formant l'ensemble dominant D sont connectés entre eux et servent de relais pour la diffusion de messages de *broadcast*. Par conséquent, le contrôle du nombre de rediffusions réduit considérablement le nombre total de paquets transmis, la redondance

de transmission, la consommation d'énergie et la collision [24]. La recherche de l'ensemble dominant connecté minimal (MCDS pour *Minimal Connected Dominating Set*) est NP-difficile [39]. Plusieurs heuristiques ont été proposées pour l'approximation du MCDS. Le processus est composé de deux phases : la construction du CDS et la maintenance de la structure du réseau. La première phase est accomplie en choisissant l'ensemble des noeuds qui vont jouer le rôle des dominants. Une dorsale est alors formée autour des noeuds dominants. La deuxième phase est nécessaire pour maintenir l'organisation du réseau, en présence de la mobilité des noeuds.

3.2.1 Construction du CDS

On distingue deux méthodes pour la construction du MCDS. La première méthode appelée *pruning* consiste à trouver un CDS valide, qui respecte la condition sur le nombre de sauts (k) entre les dominés et les dominants et garantit la connexité de la dorsale. Ensuite, on élimine les redondances pour essayer d'obtenir un MCDS [40]. Dans l'algorithme de Wu [41] où $k = 1$, chaque noeud échange la liste de ses voisins et s'il a deux voisins non connectés, alors il se considère comme dominant. Cette phase permet de construire une dorsale connectée, mais elle est encore loin de présenter un MCDS : sa taille doit être réduite. L'élimination des noeuds est basée sur des relations de connexité dans le voisinage et peut être traduite par deux règles. La première règle exclut tout noeud ayant son voisinage déjà couvert par un autre noeud dominant, possédant un identifiant supérieur que le sien. La deuxième règle élimine tout noeud ayant son voisinage couvert par deux autres noeuds dominants, s'il le plus petit identifiant.

Yang [42] propose une méthode itérative d'élimination en considérant le cas où $k \geq 1$, en se basant sur le travail de Wu [41] présenté dans le paragraphe précédent. A chaque noeud x, est associé un numéro noté $x.num$. Ce numéro prend des valeurs entières entre 0 et ∞. Il est différent de l'identifiant du noeud, noté $x.id$. $N(x)$ désigne l'ensemble y des voisins de x et $P(x)$ l'ensemble des voisins de x vérifiant la propriété suivante : ($y.num$,$y.id$) > ($x.num$,$x.id$). En disposant uniquement d'informations locales, il est difficile de vérifier la cohérence de la structure globale du réseau. Il existe néanmoins une condition suffisante qui garantit la connexité de la dorsale : si pour tout noeud dominé x, $P(x)$ est un sous-graphe connecté de G, alors la dorsale est connectée. Le processus est composé de k tours et à chaque tour j ($1 \leq j \leq k$), l'ensemble $D_k[j]$ formant le CDS est obtenu par élimination à partir de l'ensemble $D_k[j-1]$ construit au tour précédent. Au tour j, tout noeud x vérifiant les deux conditions suivantes est éliminé de $D_k[j-1]$:

1. Tous les noeuds appartenant à l'ensemble $N(x) \cap D_k[j-1]$ sont directement connectés.
2. Il existe un ensemble I tel que $I \subset \{z | z \in N(x) \cap D_k[j-1] \wedge z.id > x.id\}$, les éléments

de I forment un graphe connecté et $(N(x) - I) \cap D_k[j-1] \subseteq \bigcup_{z \in I} N(z) \cap D_k[j-1]$.

Une deuxième méthode consiste à construire un ensemble indépendant maximal (MIS pour *Maximal Independant Set*) et puis sélectionner des connecteurs pour rendre l'ensemble indépendant connecté. Dans un MIS, il n'y a aucune paire de noeuds adjacents. [43] exige la construction d'un arbre couvrant avant de commencer le processus de la formation du MIS. L'inconvénient de cette méthode est l'existence d'une structure globale, donc toute modification de la topologie entraîne la recherche d'un nouvel arbre couvrant et un nouveau MIS. [44] propose une méthode basée sur des informations locales uniquement. Le processus utilise un système de marquage par couleurs : au début tous les noeuds sont blancs. Deux couleurs permettent de distinguer les noeuds appartenant au MIS (noirs) des autres noeuds (gris). Un état transitoire peut exister lors de l'exécution de l'algorithme. Au début, un noeud v se déclare comme initiateur et annonce son état à ses voisins par un message BLACK. Ce noeud est choisi aléatoirement ou suivant des critères de sélection. En interceptant ce message, les noeuds voisins deviennent gris et annoncent à leur tour leur état à leurs voisins par un message GREY. Un noeud blanc recevant un message GREY envoie un message INQUIRY et entre dans une phase transitoire d'attente. Le message INQUIRY sert à demander des informations sur l'état et le poids de ses voisins. Le poids est une somme pondérée du degré et du niveau de batterie. Le degré est le nombre de voisins blancs et en phase transitoire. Si durant la phase de transition, le noeud reçoit un message BLACK, il émet un message GREY et devient gris. Quand le temporisateur expire, si le noeud a le plus grand poids parmi ses voisins, alors il devient noir ; sinon il devient blanc. Le processus est répété jusqu'à ce que tous les noeuds soient marqué gris ou noir. La sélection des connecteurs suit un mécanisme similaire de marquage et d'échange de messages sur l'état et le poids des voisins.

3.2.2 Maintenance du CDS

La dorsale construite dans l'étape précédente peut perdre sa connexité, à cause de la mobilité des noeuds ou la disparition d'un noeud dominant. La maintenance dépend fortement de l'algorithme de construction du CDS. Les algorithmes basés sur le calcul des arbres couvrants minimaux (MST pour *Minimum Spanning Tree*) sont obligés de redéfinir la structure du réseau et reformer un nouveau CDS, en cas de changement topologique. La structure du MST est intéressante dans le cas de la diffusion *multicast*, mais la maintenance d'une telle structure engendre un surcoût important en signalisation. Elle a été essentiellement utilisée dans les réseaux de capteurs, où les noeuds sont quasi-statiques [45][27][26].

Teymoori [46] a développé une méthode de maintenance en partant d'un CDS construit

grâce au système de marquage par couleurs présenté dans le paragraphe précédent [44]. Cette méthode prend en considération la mobilité et le niveau de batterie des noeuds. Trois situations peuvent engendrer le changement d'état : le noeud est un dominant et le niveau de ses batteries devient faible, le noeud est en déplacement et le niveau de signal sur bruit d'un ancien lien descend en dessous d'un certain seuil (β) (disparition de lien) et le noeud est en déplacement et le niveau de signal sur bruit d'un nouveau lien est supérieur à un certain seuil ($\beta+\epsilon$) (apparition d'un lien). Pour chaque changement d'état, le noeud en informe ses voisins. De la même manière que dans la construction du CDS, un système de marquage par couleurs est utilisé : noir pour les dominants et gris pour les dominés. Deux états transitoires peuvent exister : en mouvement et stable. Le noeud en mouvement doit attendre de devenir stable (fixe) pour essayer de rejoindre un nouveau dominant. L'utilisation de ces états transitoires n'est pas adaptée à des environnements fortement dynamiques. Mais la méthode a l'avantage de passer facilement à l'échelle, où les changements locaux n'engendrent pas le changement de la structure globale du réseau.

Yang [42] qui a proposé une méthode pour la construction d'un CDS dans le cas où $k \geq 1$, propose aussi une méthode pour la maintenance de la dorsale. Sa méthode requiert la définition et la diffusion de plusieurs informations dans le voisinage :

* $x.parent$ est le parent de x. Si x est dominé, son parent est initialement le noeud ayant le plus grand numéro dans l'ensemble $P(x)$.
* $x.down$ est le nombre maximal de sauts entre x et ses descendants.
* $x.up$ est le nombre de sauts entre x et son dominateur.

Un noeud x est un descendant du noeud y s'il existe un ensemble de noeuds $x_1, x_2, \cdots, x_n$ avec $x_1 = x$, $x_n = y$ et $x_i.parent = x_{i+1}$ pour $1 \leq i \leq n-1$. Un noeud y est un dominateur de x, si y est un dominant et x est un descendant de y. On note C la condition suivante : $P(x)$ n'est pas un sous-graphe connecté de G ou $x.down + z.up \geq k$ pour tout $z \in P(x)$.

Périodiquement :
- Chaque noeud x échange avec tous ses voisins sa position et les valeurs de $x.id$, $x.num$, $x.parent$, $x.down$ et $x.up$ et si x est dominant alors, il échange la valeur de sa priorité ($x.pri$) avec ses voisins dominants. En fonction des informations reçues, le noeud met à jour les valeurs de $x.down$ et $x.up$.
- Ensuite, si le noeud est dominant et vérifie la condition C, alors il fixe la valeur de sa priorité de manière à avoir la priorité la plus inférieure parmi ses voisins dominants, car la probabilité d'être éliminé est proportionnelle à la priorité. Si le noeud est dominant mais il ne vérifie pas la condition C, alors il quitte la dorsale et fixe la valeur de son numéro de manière à avoir le plus large numéro parmi ses voisins dominés. Cette mesure évite de perturber les ensembles P de ces derniers.

- Finalement, si le noeud est dominé et vérifie la condition C, alors il rejoint la dorsale, réinitialise la valeur de son numéro à ∞ et fixe la valeur de sa priorité de manière à avoir la priorité la plus inférieure parmi ses voisins dominants. Dans la dernière étape, le noeud met à jour et $x.parent$ et $x.up$.

3.2.3 Adéquation du concept basé sur le CDS

Parmi les six propriétés recherchées pour la procédure de choix des noeuds communiquant avec le réseau, la majorité des algorithmes proposés pour la structuration du réseau sous forme d'une dorsale vérifient ces quatre propriétés : la réduction de la signalisation (uniquement des informations locales), l'émergence d'un comportement global à partir des informations locales (construction de la dorsale), l'adaptation à la dynamique du réseau (maintenance) et la cohérence (garantie de la connexité). En revanche, une des propriétés n'est pas toujours respectée : la focalisation sur les propriétés intrinsèques des noeuds. La plupart des algorithmes reposent sur les propriétés topologiques de connexité dans le voisinage, car ils se sont focalisés en premier lieu sur la contrainte de connexité de la dorsale.

Il reste la dernière propriété à vérifier : la minimisation du nombres de noeuds choisis. En effet, la construction d'un ensemble dominant impose une contrainte forte : garantir la connexité de la dorsale. Avec cette contrainte, le nombre de noeuds sélectionnés est assez important. A titre d'exemple, pour un réseau de 100 noeuds, [46] génère une dorsale de taille 30. Dans notre contexte, il n'est pas nécessaire d'avoir des passerelles connectées entre elles, car on s'intéresse plus au trafic à destination des autres partitions qu'au trafic entre les noeuds d'une même partition. Nous étudions dans la section suivante, le cas où l'ensemble dominant est indépendant ; c'est-à-dire le cas où la condition sur la connexité de la dorsale est relaxée.

3.3 Ensemble dominant indépendant

La technique de construction de l'ensemble dominant indépendant est communément appelée *clustering* dans la littérature. Cette technique a été utilisée dans les réseaux ad hoc depuis leur apparition pour démontrer leur capacité d'adaptation aux changements topologiques [47] [48] [49]. Initialement, le *clustering* est une technique qui a été proposée pour résoudre le problèmes de mise à l'échelle des réseaux ad hoc : le routage, la réutilisation spatiale des ressources, le partage des applications et l'utilisation de la bande passante [50].

Le réseau est divisé en plusieurs groupes virtuels appelés *clusters* et à l'intérieur de chaque *cluster*, un noeud particulier appelé *clusterhead* coordonne les activités des autres membres (Figure 3.2). Un *cluster* est associé à une zone de service, une zone géographique ou une fonction donnée. Le *clustering* peut être alors comparé à la notion de sous-réseaux dans les réseaux IP et les *clusterheads* à des stations de base dans les réseaux cellulaires. Dans le réseau, on distingue trois types de noeuds : les *clusterheads* qui sont les chefs de *clusters*, les membres qui sont les noeuds affiliés à un *clusterhead* et les orphelins qui sont les noeuds qui n'ont pas encore choisi leur *clusterhead*. De la même manière que la technique de l'ensemble dominant connecté, le *clustering* est composé de deux étapes : la formation des *clusters* et la maintenance de la structure du réseau. Mais nous avons des approches plus focalisées sur les propriétés intrinsèques des noeuds que sur les propriétés de connexité du voisinage [1].

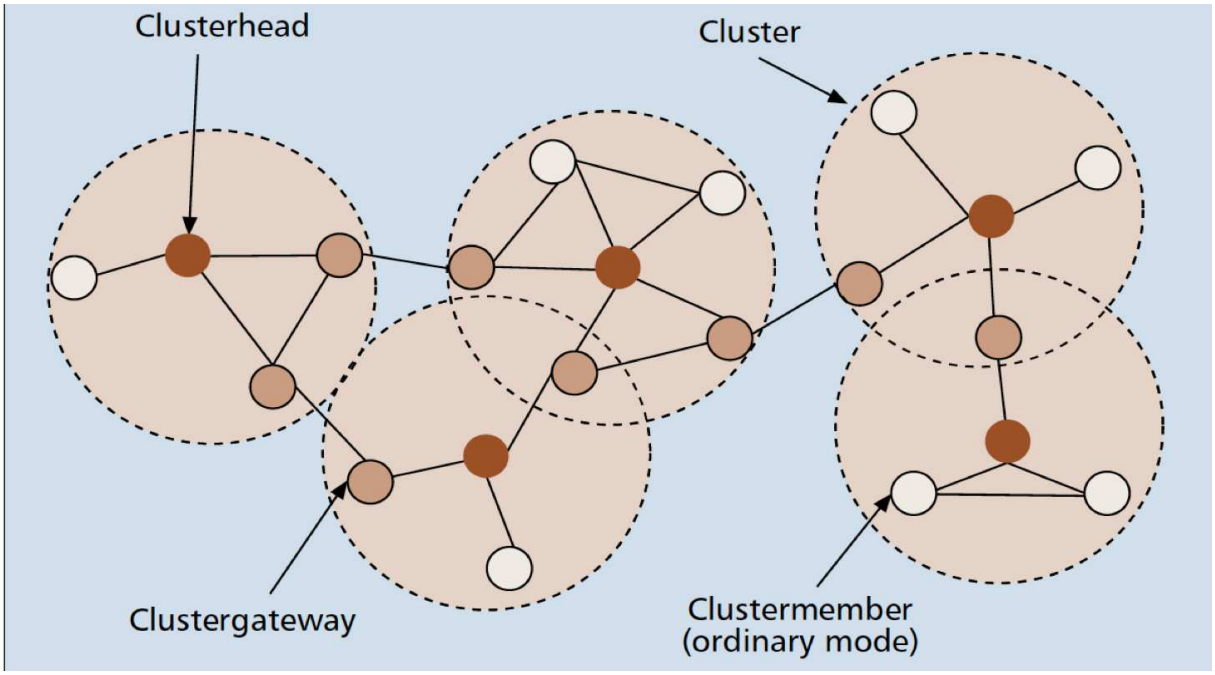


FIG. 3.2 – Organisation du réseau avec des *clusters* (source [51]).

Un noeud qui va avoir accès au satellite doit présenter certaines caractéristiques qui le permettent d'assurer son rôle comme point d'accès au niveau de chaque partition. Par exemple, nous devons éviter de sélectionner un noeud à faible niveau d'énergie où à très forte mobilité. Pour l'étude des algorithmes de *clustering*, nous adoptons une découpe

[1] Il existe toutefois, quelques algorithmes qui n'utilisent aucun critère de choix et la désignation du *clusterhead* est complètement aléatoire.

légèrement différente de la section précédente. Nous étudions dans un premier temps les critères utilisés pour choisir qui peut prétendre au statut de *clusterhead*, afin de montrer leur diversité par rapport au concept précédent (ensemble dominant connecté). Ensuite, nous décrirons les mécanismes de formation et de maintenance des *clusters* pour le cas où un membre est à un seul saut de son *clusterhead* et le cas où il est à plusieurs sauts.

3.3.1 Critères de choix du clusterhead

Les premiers travaux sur le *clustering* se sont intéressés plus à des problématiques liées à la structuration des *clusters* qu'aux critères de choix du *clusterhead* [47] [48] [49]. Lin et Gerla[52] posent le problème de la taille optimale des *clusters*. Leur objectif est de trouver un compromis entre la réutilisation spatiale des canaux de transmission (petits *clusters*) et la minimisation des délais de transmissions (gros *clusters*). Dans leur algorithme, la décision est basée sur les identifiants des noeuds : le noeud qui a le plus petit identifiant est choisi comme *clusterhead*. LCC (*Least Cluster Change*)[53], 3hBAC (*3-hop Between Adjacent Clusterhead*)[54] et [52] sont tous basés sur l'identifiant du noeud comme critère de choix. L'idée de prendre en considération les propriétés intrinsèques du noeud a été proposée par Basagni [55] en 1999 en introduisant la notion du poids.

Les caractéristiques d'un noeud peuvent être classées en trois groupes : le premier groupe considère les propriétés topologiques du noeud, le deuxième la stabilité des liens radio et le troisième la mobilité des noeuds.

3.3.1.1 Propriétés topologiques

3.3.1.1.1 Degré du noeud Le degré du noeud est le nombre de voisins à un seul saut. Il est le critère le plus utilisé dans les algorithmes de *clustering*. Il fut la métrique de base de plusieurs travaux de recherche WCA [2] [56], WACA [3] [57], k-CONID [4][58] et CEMCA [5] [59]. Il y a deux méthodes pour modéliser le voisinage entre les noeuds dans le réseau. La première méthode considère uniquement les distances euclidiennes. Si la distance entre deux noeuds u et v est inférieure à une distance seuil r, alors la liaison radio existe et les deux noeuds sont voisins. La distance r est choisie en fonction de la technologie utilisée et du milieu (rural ou urbain, à l'intérieur ou à l'extérieur des bâtiments). Par exemple un équipement « Cisco Aironet 802.11a/b/g Wireless CardBus Adapter » assure un débit égal à 18 Mbit/s avec une portée de 54 m à l'intérieur des bâtiments [60].

[2] WCA : Weighted Clustering Algorithm
[3] WACA : Weighted Application Aware Clustering Algorithm
[4] k-CONID : k-hop CONnectivity-IDentity based clustering algorithm
[5] CEMCA : Connectivity, Energy and Mobility Driven Clustering Algorithm

La deuxième méthode est fondée sur le rapport signal sur bruit (SNR pour *Signal-to-Noise Ratio*). Cette méthode est plus réaliste car elle prend en considération les atténuations causées par l'environnement (interférences, multitrajets, etc.) et elle est mesurée directement au niveau du récepteur. Au dessus de la valeur d'une puissance seuil appelée la sensibilité du récepteur (*receiver sensitivity*), le récepteur u est capable de démoduler le signal envoyé par v et les noeuds u et v sont donc considérés comme voisins. La sensibilité du récepteur est déterminée en fonction de la modulation, du taux d'erreur binaire (*BER* pour *Bit Error Rate*), de l'application et de la nature des données transmises.

Maximiser ce critère présente plusieurs avantages. Plus le noeud a de voisins, plus il a la capacité à collecter et diffuser l'information. Le noeud qui a un degré plus grand est capable d'atteindre en un seul saut plus de noeuds. Un autre facteur important est la robustesse contre le partitionnement. Le réseau est dynamique et la connectivité des noeuds peut changer au cours du temps. Plus le degré du noeud est faible, plus la probabilité de devenir un noeud isolé [6] est plus grande. Or, le noeud élu pour communiquer avec le satellite doit être connecté à tout instant aux autres noeuds de la partition. Maximiser ce critère présente au même temps une importante limitation. Plus le noeud a de voisins, plus il est soumis aux interférences. Les perturbations d'un système de radiocommunication mobile proviennent du système lui-même car plusieurs communications sont actives simultanément et partagent le même canal. L'accès au médium se heurte à deux problèmes classiques en communication hertzienne, connus sous les noms de problème du noeud caché et de problème du noeud exposé.

3.3.1.1.2 Distance moyenne entre le noeud et ses voisins La distance entre le noeud et ses voisins peut être déterminée de deux manières : l'utilisation d'un système de positionnement ou l'utilisation de la puissance du signal reçu. En effet, l'atténuation du signal est proportionnelle au carré de la distance, en espace libre. DSCAM [50] se base sur la formule de Friis pour calculer la distance entre les noeuds u et v à un instant t :

$$d_{(u,v)}(t) = \frac{\alpha}{\sqrt{P_r}},$$

où P_r est la puissance du signal reçu à l'instant t et α un facteur multiplicatif. Le calcul de la distance moyenne donne une relation entre le degré et la distance :

$$DM(u) = \frac{1}{\delta(u)} \sum_{v \in N(u)} d_{(u,v)}(t),$$

avec $\delta(u)$ est le degré du noeud u.

[6] On appelle un noeud isolé, un noeud qui n'a pas de voisins.

3.3.1.1.3 Centralité Avec des critères de sélection tel que le degré et la distance, on ne peut pas identifier les noeuds ayant des positions non favorables de point de vue topologique. Les noeuds qui sont localisés par exemple au bord de la partition sont supposés quitter la partition plus tôt que les noeuds qui sont au centre. M.R. Brust et al. [61] ont développé des algorithmes qui permettent d'identifier les noeuds-pont et les noeuds de frontière. Un noeud-pont est un noeud qui offre le seul chemin local pour communiquer entre différents groupes et sa perte engendre le partitionnement du réseau. Placer le *clusterhead* aussi central que possible dans la partition permet de réduire le chemin entre le *clusterhead* et les autres noeuds. Étant donné qu'il est difficile de localiser le centre global de la topologie à partir uniquement des informations locales, il est néanmoins possible d'éliminer les noeuds situés sur la bordure de la partition.

3.3.1.2 Stabilité des liens radio

Des critères purement topologiques ne prennent pas en considération les propriétés du médium sans fil ; d'où l'idée d'évaluer la qualité et la stabilité des liens radio. L'algorithme de *clustering* WACHM (*Weight Based Adaptive Clustering for Large Scale Heterogeneous MANET*) [62] quantifie la stabilité des liens en introduisant le concept de durée de vie. Le but est d'estimer la moyenne de temps $T_{(u,v)}$, durant lequel le noeud v reste dans la zone de couverture radio du noeud u. Cette estimation est basée sur la probabilité de dépendance $P_{d(u,v)}$ qui représente la probabilité que v reste dans la zone de couverture de u : $P_{d(u,v)} = 1 - (\frac{d(u,v)}{r})^2$, où r est la portée radio. La variation de la probabilité de dépendance entre deux instants t et $t + \Delta$, est égale à : $\Delta P_{d(u,v)}(t, \Delta t) = P_{d(u,v)}(\Delta t) - P_{d(u,v)}(t)$. Chaque noeud enregistre cette mesure pour m intervalles de temps successifs et calcule la variation moyenne de la probabilité de dépendance $VP_{d(u,v)}(m)$ pour déterminer si le noeud est en train de se rapprocher ou de s'éloigner de son voisin. Il calcule aussi la moyenne absolue de la variation de la probabilité de dépendance $MVP_{d(u,v)}$ afin de caractériser la stabilité du lien radio. Ces deux mesures sont exprimées de la manière suivante :

$$\begin{aligned} VP_{d(u,v)}(m) &= \frac{1}{m} \sum_{k=t-m+1}^{t} \Delta P_{d(u,v)}(k); \\ MVP_{d(u,v)} &= \frac{1}{m} \sum_{k=t-m+1}^{t} |\Delta P_{d(u,v)}(k)|. \end{aligned}$$

Finalement, la durée de vie du lien radio est exprimée de la manière suivante :

$$T_{(u,v)}(m) = \begin{cases} \frac{1-P_{d(u,v)}(t)}{MVP_{d(u,v)}(m)} & VP_{d(u,v)} \geq 0; \\ \\ \frac{1+P_{d(u,v)}(t)}{MVP_{d(u,v)}(k)} & VP_{d(u,v)} < 0. \end{cases}$$

Remarque 1. *Un deuxième algorithme appelé KCMBC (K-hop Compound Metric Based Clustering) [63] intègre aussi la notion de durée de vie pour caractériser la stabilité des liens radio. Une description détaillée de cet algorithme sera donnée dans la section 3.4.*

3.3.1.3 Mobilité des noeuds

Les noeuds du réseau sont mobiles d'où le besoin de caractériser et d'identifier les noeuds les plus stables. La stabilité est liée à l'évolution au cours du temps des propriétés du noeud. L'approche basée sur la caractérisation de la stabilité des liens radio, présentée dans le paragraphe précédent, prend en considération la dynamique du réseau. Mais les approches que nous allons présenter dans ce paragraphe se focalisent particulièrement sur la caractérisation de la mobilité relative. Pour illustrer cette notion de mobilité relative, prenons l'exemple de deux noeuds qui possèdent la même vitesse et qui suivent une même ligne droite (dans le même sens). Même si la vitesse est grande, la mobilité relative est nulle car la distance entre les deux noeuds ne varie pas au cours du temps.

3.3.1.3.1 MOBIC : Mobility Based Metric for Clustering Les travaux de Basu et al. [64] furent les premiers travaux de recherche qui essaient d'introduire la notion de quantification de la mobilité dans le choix des *clusterheads* [7]. L'idée de leur algorithme appelé MOBIC est qu'en mesurant le rapport de puissance ($RxPr$) de deux paquets HELLO successifs, on peut avoir une approximation de la mobilité relative entre deux noeuds. Le but est de choisir comme *clusterhead* le noeud le plus stable par rapport à ses voisins. Donc la mobilité relative du noeud y par rapport au noeud x est définie par :

$$M_y^{rel}(x) = 10 \log_{10} \frac{RxPr_{x \to y}^{new}}{RxPr_{x \to y}^{old}}.$$

Si $M_y^{rel}(x)$ est négative, alors les noeuds sont en train de s'éloigner l'un de l'autre. Dans le cas contraire, les noeuds sont en train de se rapprocher. La mobilité locale du noeud est égale à la variance par rapport à zéro de toutes les valeurs de mobilité relative et est définie par :

$$M_y = var_0(M_y rel(x_j))_{j=1}^{\delta(y)} = E[(M_y^{rel})^2].$$

Une petite valeur de M_y signifie que le noeud à une mobilité très réduite relativement à ses voisins, donc plus stable.

Remarque 2. *Pour le calcul de MOBIC, on suppose que deux paquets successifs subissent la même atténuation due à l'environnement.*

[7] A titre comparatif, MOBIC a été développé en 2001 et WACHM en 2007.

3.3.1.3.2 DSCAM : Distributed Scenario-based Clustering Algorithm for MANET Dans une approche basée sur MOBIC, le noeud ne garde en mémoire que la dernière valeur de la puissance reçue et la comparaison se fait uniquement entre deux instants successifs. Anitha [50] propose un algorithme appelé DSCAM où il considère l'évolution de la distance entre les noeuds voisins sur plusieurs intervalles de temps. Cette approche donne une vision sur la dynamique du voisinage et caractérise la stabilité locale du noeud. Chaque noeud u enregistre m mesures : à chaque instant $t = t_1, t_2, \cdots, t_m$, il calcule la mobilité relative (MR) par rapport à chacun de ses voisins v :

$$MR_{(u,v)}(t) = d_{(u,v)}(t + \Delta t) - d_{(u,v)}(t).$$

A ce niveau, la mesure MR est similaire à celle de MOBIC. L'intérêt de DSCAM réside dans l'introduction de la moyenne de cette mesure sur m intervalles de temps successifs :

$$\overline{MR_{(u,v)}} = \frac{1}{m}(MR_{(u,v)}(t_1) + MR_{(u,v)}(t_2) + \cdots + MR_{(u,v)}(t_m)).$$

Dans une dernière étape, la stabilité locale (ST) est exprimée en moyennant la déviation standard de la mobilité relative (MR) :

$$ST = \sum_{v \in N(u)} \sqrt{\frac{1}{m}\sum_{i=1}^{m}(MR_{(u,v)}(t_i) - \overline{MR_{(u,v)}})^2}.$$

3.3.1.3.3 DGMA : Distributed Group Mobility Adaptive Clustering Algorithm DSCAM, présenté dans le paragraphe précédent, propose une méthode pour quantifier la dépendance spatiale des schémas de déplacement. Mais cette quantification se limite à une vision microscopique de la dynamique du voisinage. Dans les réseaux MANETs, on peut assister à un phénomène de déplacement de groupe. Les noeuds appartenant à un même groupe sont mobiles du point de vue micro-mobilité, mais tout le groupe peut être considéré comme stationnaire du point de vue macro-mobilité. DGMA [65] est un algorithme de *clustering* qui propose une méthode pour détecter la similarité des déplacements et extraire les caractéristiques de mobilité de groupe. Le but est de trouver le *leader* du groupe dont le schéma de mobilité est suivi par les membres de son groupe. On note :

* t : l'instant présent.
* T : l'instant de la dernière mise à jour de la vitesse et la direction.
* Δx_T : le changement de l'abscisse entre l'instant t et T ; $\Delta x_T = (x_t - x_T)$.
* Δy_T : le changement de l'ordonnée ; $\Delta y_T = (y_t - y_T)$.

* D_{seuil} : une distance seuil.

La variation de la distance euclidienne est calculée de la manière suivante : $D = \sqrt{\Delta x_T^2 + \Delta y_T^2}$. Si $D < D_{seuil}$, alors on suppose que le noeud n'a pas bougé. Dans le cas contraire, la vitesse et la direction du noeud u sont mises à jour :

$$\text{* la vitesse} \quad : S_u = \frac{D}{t-T}$$

$$\text{* la direction} \quad : \theta_u = \begin{cases} \phi \times sgn(\Delta y_T) & \text{si } \Delta x_T > 0 \ ; \\ \pi/2 \times sgn(\Delta y_T) & \text{si } \Delta x_T = 0 \ ; \\ (\pi - \phi) \times sgn(\Delta y_T) & \text{sinon} \ ; \end{cases}$$

où $tan\phi = \left|\frac{\Delta y_T}{\Delta x_T}\right|$, $\phi \in [-\pi, \pi]$ et $sgn(x)$ est le signe du nombre réel x. La vitesse et la direction sont ensuite utilisées pour le calcul la dépendance spatiale totale (DST). Au début, chaque noeud échange avec ses voisins les valeurs de S_u et de θ_u. Ensuite, il calcule pour chaque voisin v, la direction relative (DR) et le rapport de vitesse (RV) :

$$DR(u,v,t) = cos(\theta_u(t) - \theta_v(t)) \quad ; \quad RV(u,v,t) = \frac{min(S_u(t), S_v(t))}{max(S_u(t), S_v(t))}.$$

La dépendance spatiale (DS) de u par rapport à v est égale à $DS(u,v,t) = DR(u,v,t) \times RV(u,v,t)$. Si u a une petite valeur de DS relativement à v, alors leurs mouvements sont indépendants. Finalement, la dépendance spatiale globale (DST) est égale à :

$$DST(u,t) = \sum_{v \in N(u)} DS(u,v,t).$$

Plus la valeur de la dépendance spatiale globale est grande, plus le noeud a de voisins ayant le même schéma de mobilité que le sien.

Dans la littérature, on trouve un large panel de critères de sélection des *clusterheads*, couvrant plusieurs aspects tels que les propriétés topologiques, la mobilité et la stabilité. La notion du noeud le plus performant dépend essentiellement du contexte, des applications et des services fournis dans le réseau. La sélection du *clusterhead* peut être mono ou multi critères. En effet, elle peut combiner plusieurs paramètres au même temps tels que le degré, la mobilité, le niveau des batteries [66]. La méthode la plus courante, *Simple Additive Weight* affecte à chaque paramètre un poids suivant l'importance accordée à un paramètre par rapport à un autre. Toutefois, les paramètres ont des ordres de grandeurs et des unités très hétérogènes. En outre, certains paramètres sont à minimiser et d'autres à maximiser. Une méthode de décision appelée *Grey Decision Method* [59] réduit à la même

échelle les différents paramètres et distingue les paramètres à maximiser des paramètres à minimiser en définissant la notion de bénéfice et de perte.

Nous décrivons dans les deux sections suivantes les procédures de formation et de maintenance des *clusters*. Les algorithmes basés sur un ou plusieurs critères requièrent une phase d'initialisation pour le calcul du poids avec un échange d'informations avec le voisinage. Donc cette phase ne sera pas à chaque fois reprise. Pour l'étude des procédures de formation et de maintenance, on considère le cas où le membre est à un seul saut de son *clusterhead* et le cas où il est à plusieurs sauts.

3.3.2 Clustering à un seul saut

Les premiers travaux sur le *clustering* se sont intéressés au développement d'heuristiques qui divisent le réseau en plusieurs groupes de manière distribuée. Donc ils se sont focalisés tout d'abord sur le cas où les membres et les *clusterheads* sont voisins ; ce cas est appelé le *clustering* à un seul saut. Le travail de Lin et Gerla [52] est l'une des plus importantes contributions dans ce domaine ; il est à la base de nombreux travaux jusqu'à aujourd'hui. Leur objectif est de trouver un ensemble de *clusters* couvrant tout le réseau.

La deuxième grande contribution dans ce domaine est la travail de Basagni [55] qui a introduit la notion du poids dans le choix des *clusterheads*. En effet, dans l'approche de Lin et Gerla, le choix est basé sur les identifiants des noeuds. Elle ne prend pas donc en considération le fait que le *clusterhead* est capable ou pas de supporter les fonctionnalités supplémentaires qui y lui sont attribuées du fait qu'il est le chef du *cluster*. Ces deux contributions étaient deux étapes importantes pour le développement de nouveaux algorithmes plus performants. La troisième approche que nous allons étudier est une mise en pratique de la notion de poids proposée par Basagni.

3.3.2.1 Approche de Lin et Gerla : lowest ID

Chaque noeud u dispose d'un identifiant unique $id(u)$ et connaît les identifiants de ses voisins à un seul saut, tous stockés dans un ensemble $\Gamma(u)$. Si le noeud possède le plus faible identifiant dans $\Gamma(u)$, alors il devient *clusterhead* et l'identifiant du *clusterhead* id_{ch} est initialisé à $id(u)$. Il diffuse ensuite un message CLUSTER (id, id_{ch}) pour informer ses voisins de sa décision et supprime id_{ch} de l'ensemble Γ. Si le noeud ne connaît pas encore son *clusterhead*, id_{ch} est initialisé à *unknown*. Chaque noeud u exécute une série d'instructions jusqu'à ce que Γ devienne l'ensemble vide :

1. A la réception d'un nouveau message CLUSTER d'un voisin v : si v est un *clusterhead* d'identifiant $id_{ch}(v)$ et si $id_{ch}(u)$ est inconnu ou supérieur à $id_{ch}(u)$, alors u

rejoint le *cluster* de v et enlève l'identifiant de v de Γ.

2. Si son identifiant est le plus petit dans Γ et son *clusterhead* est inconnu, alors il se déclare comme *clusterhead*, diffuse l'information à ses voisins et enlève son identifiant de son ensemble Γ.
3. Si Γ est l'ensemble vide, le processus de formation des *clusters* est terminé pour le noeud u.

Grâce à ce mécanisme, le réseau est entièrement divisé en plusieurs *clusters*. Mais cette structure peut être altérée à cause de la mobilité des noeuds. L'algorithme propose un mécanisme de maintenance qui essaye de limiter le nombre de transitions (réaffiliations) autant que possible. Quand un noeud u découvre qu'un membre de son *cluster* v ne fait plus partie de son *cluster*, il vérifie si le noeud au plus haut degré est un de ses voisins. Si c'est le cas, u enlève v de sa table de *cluster* ; sinon il change lui-même de *cluster*. Ce mécanisme suppose que le noeud connaît son voisinage à deux sauts.

3.3.2.2 Approche de Basagni : DCA et DMAC

L'idée de Basagni est que chaque noeud doit être affilié au meilleur voisin. L'avantage de l'utilisation d'un poids est qu'en quantifiant les paramètres du noeud, on peut choisir comme *clusterheads* les noeuds les plus performants. DCA (*Distributed Clustering for Ad hoc networks*) [55] est une généralisation de l'algorithme de Lin et Gerla, présenté dans le paragraphe précédent. DMAC (*Distributed and Mobility-Adaptative Clustering*) [55] est une extension du DCA avec la mise en place de mécanismes de maintenance.

Chaque noeud dispose d'informations uniquement sur ses voisins à un seul saut : leurs identifiants (id_u) et leurs poids (w_u). Dans DCA, chaque noeud décide de son propre rôle uniquement après que tous ses voisins de poids supérieurs décident de leur rôle dans le *cluster*. Donc initialement, uniquement les noeuds ayant les poids les plus élevés commencent par diffuser un message CH, pour mettre au courant leurs voisins qu'ils seront *clusterheads*. A la réception de ce message venant d'un noeud u, le noeud v vérifie s'il a reçu de tous ses voisins z tels que $w_z > w_v$ un message JOIN *(z, x)*, indiquant que z va rejoindre un *clusterhead* x quelconque. Dans ce cas, v rejoint le *clusterhead* u, sinon il attend un message de z. Si un noeud ne reçoit aucun message CH de la part de ces voisins de poids supérieurs, il se déclare comme *clusterhead* et suppose que ces noeuds ont rejoint d'autres *clusters* comme membres. DCA permet à des noeuds d'être des *clusterheads* même s'il a des voisins possédant des poids supérieurs mais appartenant à d'autres *clusters*.

L'algorithme de Lin et Gerla a commencé par définir le cas où un noeud quitte le *cluster* ou quitte défini-tivement le réseau. DMAC définit trois autres cas : un noeud rejoint le réseau, un lien apparaît et un lien disparaît. Ces quatre cas forment ensemble les bases pour tous les algorithmes de *clustering* venant après et implémentant des mécanismes de maintenance. Dans DMAC, quand un noeud u perd le lien avec v, s'il est lui même *clusterhead*, il supprime v de sa table de *cluster* et si v est le *clusterhead*, il essaye de joindre un autre *cluster*. Si un noeud u devient voisin avec un *clusterhead* v et le poids de v est supérieur à celui du *clusterhead* actuel de u, alors u rejoint v.

3.3.2.3 Approche de Dhurandher et Singh : WBACA

WCA (*Weight Clustering Algorithm*) [56] est l'un des premiers algorithmes qui ont mis en pratique la notion de poids proposée par Basagni [55], en calculant le poids en fonction du degré, de la vitesse, de la somme des distances et du temps durant lequel le noeud est élu comme *clusterhead*. Mais son inconvénient majeur est le calcul d'un minima global. Tous les noeuds ont donc besoin de connaître l'état de tous les autres noeuds du réseau.

WBACA (*Weight Based Adaptive Clustering Algorithm*) [67] est un algorithme de *clustering* proposé par Dhurandher et Singh où le choix des *clusterheads* repose aussi sur plusieurs critères : le degré, la vitesse, la puissance et le débit de transmission. Contrairement à WCA, il repose sur la connaissance du voisinage à deux sauts uniquement. Chaque noeud envoie à ses voisins une liste comportant les identifiants et les poids de tous ses voisins. De cette manière, chaque noeud prend connaissance de son voisinage à deux sauts. Si le noeud possède le plus petit poids dans son voisinage, il se déclare comme *clusterhead*, sinon le noeud devient membre et envoie une requête JOIN-REQUEST au *clusterhead* candidat. Ce dernier peut accepter ou décliner cette requête et dans les deux cas, il répond par un acquittement (JOIN_ACK). Si la réponse est négative, il réitère l'opération avec tous ses voisins jusqu'à joindre un *cluster* ou épuiser un nombre limite de tentatives.

Pour la maintenance des *clusters*, WBACA considère trois cas :

1. Quand le noeud quitte son *cluster* et n'arrive pas à joindre un nouveau, il se déclare comme *clusterhead* ;
2. En cas de rupture de lien, le noeud met à jour sa table de voisinage et en informe ses voisins. Si le noeud perd le lien avec son *clusterhead*, il essaye de joindre un nouveau en envoyant un JOIN_REQUEST.
3. Si deux *clusterheads* deviennent voisins, celui qui a le plus grand poids rejoint le *cluster* de l'autre.

Autres que WBACA, plusieurs algorithmes tels que MOBIC [64], CEMCA[59] et DGMA [65] mettent en oeuvre divers critères de choix. Un aperçu de ces critères est donné dans la section 3.3.1.

Contrairement à la structuration du réseau sous forme d'un ensemble dominant connecté, le *clustering* à un seul saut se focalise sur les propriétés intrinsèques du réseau. Mais son inconvénient majeur c'est qu'il crée un grand nombre de *clusters* de tailles très réduites. En outre, la structure des *clusters* change dès qu'un noeud se retrouve en dehors de la portée de transmission de son *clusterhead*. A cause de la mobilité des noeuds, la topologie du réseau change fréquemment modifiant ainsi la structure des *clusters*. Pour plus de stabilité, le concept de *clustering* est généralisé pour créer des *clusters* où les noeuds sont au maximum à k sauts de leur *clusterhead* ($k \geq 1$). Le regroupement des noeuds permet plus d'évolutivité pour les réseaux MANETs. Le paramètre k est un paramètre d'entrée, à fixer par l'administrateur en fonction de la structure du réseau.

Pour le *clustering* à un seul saut, nous avons adopté une approche plus historique pour montrer le développement progressif dans la performance des algorithmes de *clustering*. Pour l'étude du *clustering* à multi-sauts, nous adoptons une approche différente. Nous allons progressivement du moins intéressant vers le plus intéressant et ceci par rapport à la problématique de la thèse et les propriétés souhaitées, énoncées dans le paragraphe 3.1.2.

3.3.3 Clustering à multi-sauts

La grande différence entre les algorithmes de *clustering* à un seul saut et à multi-sauts est la collecte et la diffusion d'informations entre les noeuds. La difficulté est d'organiser le réseau de manière distribuée et en se basant le plus possible sur des informations locales uniquement.
Pour chaque approche, nous donnerons un aperçu sur les mécanismes implémentés et nous discuterons, à fur et à mesure, son adaptation à notre problème de sélection des noeuds qui vont avoir accès au satellite.

3.3.3.1 (k,r)-dominating set

Nous allons commencer par l'algorithme DKR (*Distributed (K,R) dominating set*) proposé par Spohn et Garcia-Luna-Aceves [68] pour faire le lien avec l'approche qui consiste à structurer le réseau sous forme d'un ensemble dominant connecté, présentée dans la

section 3.2.

Dans DKR, la sélection des *clusterheads* est basée sur la recherche de l'ensemble dominant (k,r), où k est le nombre minimal de *clusterheads* par membre [8] et r est le nombre maximal de sauts entre un membre et son *clusterhead*. Cette approche est différente de la construction de l'ensemble dominant connecté car la condition sur la connexité de la dorsale n'est pas requise.

3.3.3.1.1 Formation des *clusters* Dans la première phase d'élection, chaque noeud sélectionne les k noeuds qui ont les plus petits identifiants dans son voisinage à r sauts. Ces noeuds sélectionnés sont des candidats pour devenir des *clusterheads*. C'est à l'issue de la deuxième phase que les *clusterheads* seront définitivement désignés. La première phase comporte r tours où à chaque tour j ($j \leq r$), chaque noeud sélectionne et informe ses voisins de la liste $D_j(u)$ des m ($m \leq k$) voisins à j sauts ayant les plus petits identifiants. A la fin de cette phase, un ensemble de noeuds dominants D est construit : le noeud u devient dominant si $u \in D_r(u)$ ou si $|D_r(u)| < k$. Durant la deuxième phase, chaque noeuds dominé u informe ses voisins à un seul saut de sa liste $D_r(u)$ par un message LOCAL_ADVERTISEMENT. Si un noeud dominé v reçoit ce message et il figure dans cette liste, il devient *clusterhead* et l'annonce à tous ses voisins à r sauts. Si dans la liste $D_r(u)$ figurent des éléments qui ne sont pas dans sa propre liste ($D_r(v)$), alors il les informe pour devenir des *clusterheads*. Un noeud de l'ensemble dominant initial ne devient pas *clusterhead*, s'il reçoit l'annonce de k *clusterheads*.

DSCAM [50] modifie la deuxième phase de DKR pour introduire des critères de choix autres que l'identifiant (degré, niveau des batteries et mobilité) et fixer une taille maximale pour les *clusters*. Chaque noeud dominant calcule son poids et en informe ses voisins à r sauts. Chaque noeud dominé sélectionne le dominant le plus qualifié comme *clusterhead* et lui envoie un message NODE_JOIN_REQUEST. En fonction de la taille de son *cluster*, le *clusterhead* accepte ou non la requête. Le noeud dominé est informé par la décision par un acquittement (NJ_ACK). Si un noeud dominant ne reçoit aucun message NODE_JOIN_REQUEST et a k *clusterheads* dans son voisinage à r sauts, alors il sélectionne un *clusterhead* et quitte l'ensemble des dominants.

3.3.3.1.2 Maintenance des *clusters* L'algorithme DKR n'intègre pas la maintenance des *clusters*, tandis que DSCAM s'adapte à la dynamique du réseau et à l'évolution du niveau énergétique des noeuds. Chaque *clusterhead* envoie périodiquement un message

[8] Dans le cas général, k désigne le nombre de sauts entre un membre et son *clusterhead* et dans ce contexte, k désigne exceptionnellement le nombre de *clusterheads*

CLUSTERHEAD_ADVERTISEMENT pour informer ses voisins à r sauts de sa présence et de son poids. Chaque membre du *cluster* met alors à jour sa table de *clusterheads* et sélectionne un nouveau *clusterhead* si besoin. Ensuite, il envoie un NODE_JOIN et attend l'acquittement de sa requête avec un NJ_ACK. Si aucun acquittement n'est reçu, alors il envoie sa requête au suivant sur sa liste. Dans le cas où le niveau du *clusterhead* est en dessous d'un certain seuil, il en informe ses voisins à r sauts et sélectionne un nouveau *clusterhead*. Pour un nouvel arrivé, s'il existe k *clusterheads* dans le voisinage, il sélectionne le *clusterhead* le plus qualifié et lui envoie une requête NODE_JOIN, sinon il se déclare comme *clusterhead*.

L'avantage majeur d'une telle structure est la résilience vis-à-vis de la dynamique du réseau et les changements topologiques. En contre partie, la condition sur le nombre minimal de *clusterheads* par noeud induit de la redondance et augmente considérablement le nombre des *clusterheads* dans le réseau. En outre, la maintenance de cette structure engendre un surcoût important en signalisation. Le paragraphe suivant expose une méthode dont l'objectif est de trouver un nombre minimal de *clusterheads* couvrant tout le réseau.

3.3.3.2 Sélection aléatoire

CIRCLE [69] est un algorithme de *clustering* où la sélection des *clusterheads* est totalement aléatoire. CSC (*Connectivity-based Stretchable Clustering scheme*) [70] part du même principe que CIRCLE mais essaye de réduire la part de l'aléatoire dans la formation des *clusters*. Le principe est de générer des cercles virtuels dont les centres sont les *clusterheads* et le rayon est égal à k. Le but est de réduire le nombre de *clusterheads* en s'assurant que le nombre de sauts entre deux *clusterheads* est au moins égal à $k+1$ et que les *clusters* ne superposent pas. La distance optimale entre deux *clusterheads* est appelée distance de délégation et notée g ($g > k$) [9].

3.3.3.2.1 Formation des *clusters* Dans CIRCLE, un initiateur se déclare comme *clusterhead* et débute la procédure de *clustering*. Il commence par diffuser un message FORMATION dans son voisinage à g sauts. Les noeuds situés à k sauts de l'initiateur deviennent des membres et les noeuds situés entre $k+1$ et g sauts, appelés délégués répondent à ce message en indiquant leur priorité et leur disponibilité. En fonction des réponses choisies, l'initiateur détermine le nouveau *clusterhead* qui sera informé par un message de DELEGATION. De manière itérative, chaque *clusterhead* répète les mêmes

[9] Dans CIRCLE, g est égal à $k\sqrt{3}$ et dans CSC, $2k+1$.

opérations que l'initiateur : il envoie un message à ses voisins à g sauts et attend la réponse de ses délégués. Les noeuds qui appartiennent déjà à un *cluster* ne sont pas considérés pour une possible délégation. Le processus se termine pour chaque *clusterhead* quand il n'a plus de délégués candidats. La priorité de chaque délégué est déterminée en fonction des relations de voisinage à g sauts entre lui et les *clusterheads* du réseau. Cette méthode évite la formation de zones blanches où les noeuds ne sont ni *clusterheads* ni membres d'un *cluster*.

CSC a l'avantage de déterminer comment l'initiateur est choisi. Mais on n'a pas un seul initiateur mais plusieurs. Au début, chaque noeud connaît son degré ($\delta(u)$) et les identifiants de ses voisins. Le noeud u qui a le plus petit identifiant parmi ses voisins déclenche un temporisateur $t_u = \alpha_c \max(0, 1 - \frac{\delta(u)}{\theta_c}) + bcd)$, où bcd est un délais aléatoire, α_c et θ_c des constantes. A l'expiration du temporisateur, le noeud devient *clusterhead* et procède de la même manière que dans CIRCLE, en envoyant un message FORMATION. Le délégué u attend td_u secondes avant de devenir un *clusterhead* : $td_u = \alpha_c \max(0, 1 - \frac{\delta(u)}{\theta_c}) + \alpha_d \max(0, 1 - \frac{\beta_u}{2k+1}) + bcd$, où β_u est le nombre de sauts entre le délégué et le *clusterhead* précédent et α_d une constante. Si un noeud reçoit un message FORMATION de la part d'un *clusterhead* avant l'expiration de son temporisateur, il arrête ce dernier et rejoint le *clusterhead* source du message.

3.3.3.2.2 Maintenance des *clusters* CIRCLE ne propose pas de mécanismes de maintenance ; on s'intéresse alors à CSC. Dans la phase de formation des *clusters*, les membres sont à au plus k sauts de leurs *clusterheads*. CSC propose de relaxer cette condition et permettre aux membres de se trouver à au plus k' sauts de leurs *clusterheads* avec $k' > k$. Une seule condition s'impose : il ne faut pas que les *clusters* se chevauchent. La réaffiliation est basée sur deux mesures : le degré de connectivité intra et inter *clusters*. Chaque noeud mesure le nombre de voisins appartenant au même *cluster* et le nombre de voisins appartenant à des *clusters* différents. Si un noeud s'aperçoit que la taille de son *cluster* est petite, le nombre de sauts est supérieur à k' ou le degré de connectivité intra-*cluster* est inférieure à un certain seuil, il décide de quitter son *cluster* et rejoindre un autre.

CIRCLE et CSC réduisent considérablement le nombre des *clusterheads* dans le réseau. Mais les *clusterheads* ne sont pas forcément ceux qui sont les plus performants : la sélection comprend un grand part d'aléatoire. A noter que cette approche présente tout de même plusieurs avantages notamment pour les réseaux denses, où le critère primaire est le nombre de *clusterheads* déclarés (jusqu'à 8 voisins par noeud avec une distribution uni-

forme sur le terrain).

Comme pour le *clustering* à un seul saut où les travaux de Lin/Gerla et Basagni forment les deux références clés, les propositions de Chen [58] et Amis[71] avec leurs algorithmes k-lowestID et Max-Min, qu'on présentera dans les deux paragraphes suivants, forment les deux références clés pour le *clustering* à multi-sauts.

3.3.3.3 k-lowestID

K-lowestID [58] est un algorithme proposé par Chen en 2002 et est une extension de l'algorithme de Lin et Gerla [52] pour un *clustering* à multi-sauts.

3.3.3.3.1 Formation des *clusters* k-lowestID suit presque les mêmes règles que l'algorithme originel, sauf qu'il suppose que chaque noeud connaît tous ses voisins à k sauts : chaque noeud diffuse son identifiant pour en informer son voisinage. Dans ce paragraphe, quand on parle de voisins ou de voisinage sans préciser à combien de sauts, on désigne implicitement les voisins ou le voisinage à k sauts. Le noeud possédant le plus faible identifiant devient *clusterhead* et diffuse sa décision dans son voisinage par un message CLUSTER. A la réception de plusieurs messages de ce type, un noeud choisit comme *clusterhead* celui qui a le plus petit identifiant et en informe son voisinage. Si un noeud ne reçoit pas de message CLUSTER de la part de ses voisins possédant de plus petits identifiants que lui, alors il se déclare comme *clusterhead* et envoie un message CLUSTER à ses voisins.

3.3.3.3.2 Maintenance des *clusters* k-lowestID considère quatre cas modélisant les changements topologiques dans le réseau :

1. Un noeud vient de joindre le réseau : il prend sa décision en fonction du nombre de sauts qui le sépare des *clusterheads* existants. S'il est à moins de k sauts d'un *clusterhead*, alors il rejoint ce *cluster*. Sinon, il s'auto déclare comme *clusterhead* et initie la formation d'un nouveau *cluster*.
2. Un noeud quitte le réseau : aucune modification n'est faite si c'est un membre. Sinon, le noeud qui a le meilleur poids dans le *cluster* prendra la relève et se déclarera comme *clusterhead*. Si un noeud de l'ancien *cluster* se trouve à plus de k sauts du nouveau *clusterhead*, il cherchera dans son voisinage à k-sauts un autre *cluster* pour le rejoindre.
3. Un lien disparaît entre les noeuds u et v :
   – Si u et v appartiennent à deux *clusters* différents, rien ne se passe.

- Si u et v appartiennent au même *cluster*, tous les noeuds du *cluster* sont informés et le *clusterhead* vérifie si tous ses membres restent dans son voisinage. Les noeuds se trouvant à plus de k sauts du *clusterhead* forment un nouveau *cluster*.

4. Un lien entre les noeuds u et v est formé : si aucun des deux noeuds n'est *clusterhead*, la structure du réseau reste inchangée.

Après de multiples changements, la structure du *cluster* peut devenir de mauvaise qualité. La qualité peut être mesurée en fonction de la taille du *cluster* et le taux de membres situés à la bordure du *cluster*. Par exemple, le *cluster* où tous les membres sont à un seul saut du *clusterhead* est considéré de mauvaise qualité. Dans ce cas, le *cluster* fusionne avec d'autres *clusters*.

Les auteurs de k-lowestID proposent dans la même référence un deuxième algorithme appelé k-CONID. Ce dernier introduit la connectivité (nombre de voisins à k sauts) comme critère de choix. L'identifiant n'est utilisé qu'en cas d'égalité des degrés. K-CONID présente à priori de bonnes propriétés pour être appliqué dans notre contexte. Mais le problème est le risque d'avoir des tempêtes de *broadcast*, étant donné que tous les messages sont diffusés dans le voisinage à k sauts. [63] a démontré qu'une autre approche appelée Max-Min réduit le surcoût en signalisation par rapport à une approche basée sur la connaissance de tout le voisinage à k sauts.

3.3.3.4 Max-Min

Max-Min [71] est une heuristique qui permet, à moindre coût, de diffuser l'information dans le voisinage à k sauts. Les algorithmes de *clustering* à seul saut et k-lowestID ont une complexité égale à $O(n)$ où n est la taille du réseau, tandis que l'algorithme Max-Min a une complexité égale à $O(k)$ ($k \ll n$).

Les ressources requises au niveau de chaque noeud sont minimales et consistent en quatre règles à appliquer et deux structures de données dont la taille est proportionnelle à k. En outre, la responsabilité de formation des *clusters* est distribuée de manière égale entre les noeuds. La métrique utilisée est l'identifiant du noeud, en l'occurrence son adresse MAC qui est par définition unique. On verra dans la section suivante comment Max-Min a été amélioré pour prendre en considération la qualité des liens radio dans le choix des *clusterheads*. Max-Min comporte quatre étapes : FloodMax, FloodMin, le choix du *clusterhead* et l'affiliation des membres.

3.3.3.4.1 FloodMax Chaque noeud diffuse à tous ses voisins directs l'identifiant du noeud qui a actuellement le meilleur poids : WINNER_ID. Dans FloodMax, le noeud qui a l'identifiant le plus élevé est considéré comme gagnant. Après avoir reçu les messages de tous ses voisins directs, le noeud met à jour son WINNER_ID en comparant son précédent WINNER_ID avec ceux de ses voisins. Ce processus est répété k fois.
FloodMax permet de propager le plus grand identifiant dans le voisinage à k sauts, sans inonder le réseau par des messages de contrôle. Dans certains cas, un *clusterhead* pour être disjoint de son *cluster* car il peut être dominé par un autre *clusterhead*. Par conséquent, un noeud doit se rendre compte qu'il a le plus grand identifiant dans son voisinage à k sauts et aussi qu'il a le plus grand identifiant dans le voisinage à k sauts d'un autre noeud.

3.3.3.4.2 FloodMin Une deuxième série d'envoi de k messages, appelée FloodMin, permet d'équilibrer la distribution des *clusters*. Cette étape est similaire à FloodMax, sauf que cette fois le noeud enregistre le plus petit identifiant au lieu du plus grand identifiant. La première série de messages part des identifiants sortis gagnants lors du $k$ème tour de FloodMax. FoodMin permet aux *clusterheads* qui ont des identifiants relativement petits de reprendre les noeuds qui sont dans leur voisinage à k sauts et se rendre compte qu'ils ont le plus grand identifiant dans le voisinage à k sauts d'un autre noeud.

3.3.3.4.3 Choix du *clusterhead* Chaque noeud applique trois règles dans l'ordre. Il se déclare *clusterhead*, si son identifiant est celui du gagnant du processus FloodMin. Si le noeud u trouve que le même identifiant du noeud v apparaît comme gagnant au moins une fois dans un tour parmi les k tours de FloodMax et qu'il apparaît au même temps comme gagnant au moins une fois dans un tour parmi les k tours de FloodMin, alors u choisit v comme *clusterhead*. Dans le cas où il y a plusieurs candidats, le noeud qui a le plus petit identifiant est choisi comme *clusterhead*. Finalement, le noeud qui sort gagnant du processus FloodMax se déclare comme *clusterhead*.

A la fin de cette étape, les noeuds diffusent les identifiants de leur *clusterhead* à leurs voisins. Si u a des voisins appartenant à des *clusters* autres que le sien, alors il est désigné comme passerelle entre les *clusters* voisins.

3.3.3.4.4 Affiliation des membres Le but de cette étape est d'informer le *clusterhead* de la liste des membres de son *clusterhead*. Une fois les *clusterheads* sont choisis, chaque passerelle initie la procédure d'affiliation. La passerelle envoie un message à ses voisins qui contient deux champs : le premier champ contient son identifiant et le deuxième champ est initialisé à zéro. Le deuxième champ sert à déterminer le nombre de sauts entre le noeud et le *clusterhead*. A la réception de ce paquet, chaque noeud intermédiaire ajoute son identifiant, incrémente de 1 le deuxième champ et relaye le paquet. Le paquet, en

arrivant au *clusterhead*, contient la liste de tous les membres.

Dans sa première version, Max-Min n'implémente pas de méca-nismes de maintenance. Un deuxième inconvénient est que la sélection des *clusterheads* est basée sur les identifiants des noeuds. Supeng et al. [63] ont proposé un algorithme qui remédie à ces deux problèmes : le choix des *clusterheads* repose sur l'évaluation de la qualité des liens radio et l'adaptation à la dynamique du réseau est assurée grâce à un processus de maintenance.

3.3.4 Synthèse et sélection d'un algorithme de clustering

Nous avons étudié deux concepts différents qui permettent de structurer les réseaux ad hoc mobiles. Le premier concept s'appuie sur la recherche d'un ensemble dominant connecté pour former la dorsale du réseau. La condition sur la connexité de la dorsale introduit de la complexité, de la surcharge en signalisation et de la redondance. La relaxation de cette hypothèse nous conduit au deuxième concept qui repose sur la recherche d'un ensemble dominant indépendant. Ce dernier concept est communément connu sous le nom de *clustering*.

Cette technique a été utilisée dans les réseaux ad hoc depuis leur apparition pour démontrer leur capacité d'adaptation aux changements topologiques et de passage à l'échelle. Le réseau est divisé en plusieurs groupes virtuels appelés *clusters* et dans chaque *cluster*, un noeud particulier appelé *clusterhead* coordonne les activités à l'intérieur de son *cluster*. De la même manière que la formation de l'ensemble dominant connecté, le *clustering* est composé de deux étapes : la formation des *clusters* et la maintenance de la structure du réseau. Toutefois, avec le *clustering* on a des approches plus focalisées sur les propriétés intrinsèques des noeuds que sur les propriétés de connexité du voisinage.

Le *clustering* à un seul saut se focalise sur les propriétés intrinsèques du réseau. Mais son inconvénient majeur c'est qu'il crée un grand nombre de *clusters* de taille très réduite. En outre, la structure des *clusters* change dès qu'un noeud se retrouve en dehors de la portée de transmission de son *clusterhead*. A cause de la mobilité des noeuds, la topologie du réseau change fréquemment modifiant ainsi la structure des *clusters*. Pour plus de stabilité, le concept de *clustering* est généralisé pour créer des *clusters* où les noeuds sont au maximum à k sauts de leur *clusterhead* ($k \geq 1$). Le regroupement des noeuds permet plus d'évolutivité pour les réseaux MANETs.

Dans le cas du *clustering* multi-sauts, quatre approches ont été proposées. La première

approche construit un ensemble dominant (k, r), où chaque membre doit avoir au moins k *clusterheads* dans son voisinage à r sauts. La structure créée est stable vis-à-vis de la dynamique du réseau mais la redondance augmente considérablement le nombre des *clusterheads*. La deuxième approche propose de remédier à ce problème en s'appuyant sur le principe des cercles couvrants. Elle permet de diminuer le nombre de *clusterheads* sélectionnés mais le souci est que le choix des *clusterheads* est aléatoire. Les deux dernières approches sont les plus intéressantes.

K-lowestID et Max-Min dans leurs versions originelles, considèrent les identifiants des noeuds comme critère de choix. Mais, certains travaux se sont basés sur ces deux références pour proposer d'autres critères : le degré ou la qualité des liens. La comparaison concerne alors l'étape de formation des *clusters* et plus particulièrement comment les noeuds obtiennent des informations sur leur voisinage à k sauts, afin de prendre la décision de se déclarer comme *clusterhead* ou de s'affilier à un *cluster*. [63] présente une étude comparative entre ces deux approches et montre Max-Min présente des meilleures performances en termes de surcoût en signalisation et la durée de vie des *clusterheads*.

Pour résumer, nous rappelons tout d'abord l'ensemble de critères que la procédure du choix des noeuds doit satisfaire et dont un descriptif est donné dans le paragraphe 3.1.2. Nous dressons ensuite un tableau comparatif entre les différentes approches exposées dans ce chapitre. Les critères à satisfaire sont :

- Critère 1 (C # 1) : minimisation du nombre de noeuds communiquant avec le satellite.
- Critère 2 (C # 2) : focalisation sur les propriétés intrinsèques des noeuds.
- Critère 3 (C # 3) : réduction de la signalisation.
- Critère 4 (C # 4) : émergence d'un comportement global à partir des informations locales.
- Critère 5 (C # 5) : adaptation à la dynamique du réseau.
- Critère 6 (C # 6) : cohérence.

Pour conclure, nous adoptons pour la suite une approche basée sur le concept de Max-Min, mais dans une version plus développée avec KCMBC [63] qui a introduit la maintenance et les propriétés intrinsèques des noeuds. Mais avant de passer à la description de KCMBC, nous avons une remarque concernant particulièrement la maintenance. Les auteurs de k-lowestID introduisent une technique pour maintenir la structure des *clusters*, tout en indiquant qu'il faut développer des techniques plus spécialisées. Cette remarque nous conduira par la suite à l'étude des solutions proposées. Cette étude sera présentée dans le chapitre 5. La section suivante traitera en détail les améliorations introduites à Max-Min à travers l'algorithme KCMBC.

Classe	Algorithme	Critères					
		C # 1	C # 2	C # 3	C # 4	C # 5	C # 6
ensemble dominant connecté	Wu [41]	-	-	+	+	-	+
	Yang [42]	-	-	+	+	+	+
	Yuanyuan [44]	-	+	+	+	-	+
	Teymoori [46]	-	+	+	+	+	+
ensemble dominant indépendant (*clustering)*	DKR [68]	-	-	-	+	-	+
	DSCAM [50]	-	+	-	+	+	+
	CIRCLE [69]	+	-	+	+	-	+
	CSC [69]	+	-	+	+	+	+
	k-lowestID [58]	+	-	-	+	+	+
	k-CONID [58]	+	+	-	+	+	+
	Max-Min[71]	+	-	+	+	-	+
	KCMBC [63]	+	+	+	+	+	+

Tab. 3.1 – Comparaison des approches de structuration d'un réseau ad hoc.

3.4 KCMBC

KCMBC améliore l'algorithme Max-Min en modifiant tout d'abord la méthode d'affiliation des membres à leur *clusterhead*, puis en introduisant de nouveaux critères de choix et finalement de la maintenance.

3.4.1 Affiliation des membres

La méthode d'affiliation de Max-Min souffre d'un inconvénient : les noeuds qui sont situés à la bordure du réseau ne peuvent pas annoncer leur affiliation à leur *clusterhead*. La solution consiste à modifier la manière dont on identifie les noeuds périphériques du *cluster*.

Durant chaque tour de FloodMin et de FloodMax, chaque noeud enregistre la distance (en nombre de sauts) qui le sépare du WINNER et l'identifiant du SENDER. Le SENDER est le noeud de la part de qui la meilleure valeur de poids a été reçue. Ces informations sont ensuite diffusées aux voisins. Ainsi les noeuds qui possèdent la plus grande distance vers le *clusterhead* initient la procédure d'affiliation. Chaque noeud situé à la bordure du *cluster* envoient des messages contenant l'identifiant du noeud, du *clusterhead* et la liste des noeuds voisins. Un noeud intermédiaire attend jusqu'à la réception des messages d'af-

filiation de la part de tous ses voisins qui ont une distance plus grande vers le *clusterhead*. Il ajoute ses propres informations (identifiant et distance) et rediffuse le message.

3.4.2 Critères de choix

Le choix des *clusterheads* utilise une métrique combinée CP_i qui réunit le degré, la qualité des liens radio et les identifiants des noeuds : $CP_i = (\delta_i, T_i, id_i)$, où δ_i est le degré du noeud i, T_i son temps d'expiration et id_i son identifiant. Afin de définir le meilleur candidat, la comparaison entre deux noeuds i et j est exprimée par la relation suivante :

$$\{CP_i > CP_j \equiv (\delta_i > \delta_j) \cup [(\delta_i = \delta_j) \cap (T_i > T_j)] \cup [(\delta_i = \delta_j) \cap (T_i = T_j) \cap (id_i > id_j]\}.$$

On remarque que l'identifiant n'est vraiment utilisé qu'en cas d'égalité des degrés et des temps d'expiration. Donc la métrique introduite par KCMBC prend en considération les propriétés intrinsèques des noeuds. Le temps d'expiration d'un lien est une estimation de sa durée de vie résiduelle et traduit donc la qualité du lien radio. Si un lien a un temps d'expiration de 100 s, cela signifie qu'on estime que le lien disparaîtra dans 100 s. Au niveau du noeud, le temps d'expiration est une moyenne des temps d'expiration de tous ses liens. Afin de s'assurer que seuls les noeuds les plus performants vont être sélectionnés comme *clusterheads*, l'algorithme impose une condition sur le temps d'expiration. Ce dernier doit être supérieur à un certain seuil T_{ALT} ; sinon le noeud ne pourra pas être élu.

Nous allons maintenant voir comment on effectue le calcul de T_i. On note s_i le vecteur de position de i, v_i le vecteur de vitesse et T_{ij} le temps d'expiration du lien (i, j). Dans un repère cartésien, on a $s_i \equiv (s_{ix}, siy)$ et $v_i \equiv (v_{ix}, viy)$. Afin d'obtenir le temps d'expiration T_{ij}, il faut résoudre une équation différentielle du second degré qui admet deux solutions de signes opposés T_+ et T_- [10]. Étant donné que T_{ij} est un nombre réel positif, on va choisir suivant le signe entre T_+ et T_-. T_{ij} est exprimé en secondes comme suit :

$$T_{ij} = \begin{cases} T_+, & \text{si } T_+ > 0; \\ \\ T_-, & \text{si } T_- > 0; \end{cases}$$

avec

$$T_\pm = [\pm\sqrt{r^2(\Delta_{vx}^2 + \Delta_{vy}^2) - (\Delta_{sx}\Delta_{vy} - \Delta_{sy}\Delta_{vx})^2} - \Delta_{sx}\Delta_{vx} - \Delta_{sy}\Delta_{vy}]/(\Delta_{vx}^2 + \Delta_{vy}^2)$$

$$\text{et} \begin{cases} \Delta_{sx} = s_{ix} - s_{jx}, \Delta_{sy} = s_{iy} - s_{jy}, \\ \Delta_{vx} = v_{ix} - v_{jx}, \Delta_{vy} = v_{iy} - v_{jy}, \\ r = \text{la portée radio.} \end{cases}$$

[10] Cette équation n'admet pas de solution si $|v_i| = 0$ et $|v_j| = 0$.

Finalement, le temps d'expiration du noeud i est calculé à partir de la formule suivante :

$$T_i = \frac{1}{\delta_i} \sum_{j \in N(i)} T_{ij}.$$

3.4.3 Maintenance des clusters

Comme on l'a indiqué dans le paragraphe 3.3.3.4, Max-Min n'introduit pas de mécanisme de maintenance. Les auteurs de KCMBC se sont basés sur la procédure de maintenance proposée dans l'algorithme k-lowestID et décrite dans le paragraphe 3.3.3.3 et l'ont adaptée pour prendre en compte la nouvelle métrique de choix.

De la même manière, KCMBC considère quatre cas modélisant les changements topologiques dans le réseau : activation d'un noeud ou d'un lien et disparition d'un noeud ou d'un lien.

3.4.3.1 Activation d'un noeud ou d'un lien

Chaque noeud envoie périodiquement un message HELLO à ses voisins à un seul saut. Quand un noeud orphelin i reçoit des messages HELLO, il essaye de joindre le *clusterhead* situé à moins de k sauts. Si plusieurs *clusterheads* sont candidats, il rejoint un *cluster* de son voisin possédant le plus grand temps d'expiration. Pour la décision d'affiliation, chaque noeud i doit donc connaître l'identifiant de chacun de ses voisins j situés à un seul saut, l'identifiant du *clusterhead* de j, le temps d'expiration de j et le nombre de sauts entre j et son *clusterhead* (celui de j).
Si le noeud orphelin détecte plus de D_r voisins orphelins, il initie un processus de *reclustering* pour former un nouveau *cluster*.

La création d'un nouveau lien entre deux noeuds i et j peut changer la structure du *cluster*. Si les deux noeuds sont des *clusterheads*, les deux *clusters* fusionnent et le *clusterhead* à plus petit temps d'expiration i rejoint, avec ses membres le nouveau *cluster* j. Si les deux noeuds appartiennent à deux *clusters* différents, aucune réaffiliation n'est initiée s'ils sont à moins de k sauts de leurs *clusterheads*. Pour prendre une telle décision, le noeud doit donc connaître le nombre de sauts qui le séparent de son *clusterhead*.

3.4.3.2 Disparition d'un noeud ou d'un lien

La structure du réseau est dynamique avec le départ des noeuds et la disparition des liens. Ces changements peuvent entraîner la perte du chemin vers le *clusterhead* ou l'allongement du plus court chemin entre le noeud et son *clusterhead* à plus de k sauts. Dans

ce cas, le noeud devient orphelin et suit la procédure d'activation d'un noeud décrite dans le paragraphe 5.2.2.1. Suite à ces évènements, la structure du *cluster* peut devenir de mauvaise qualité : tous les membres sont à un seul saut de leur *clusterhead*. Dans ce cas, le *cluster* fusionne avec les *clusters* voisins. En outre, quand le *clusterhead* prédit la disparition de son dernier voisin, il choisit un nouveau noeud pour prendre le relais. Les noeuds qui ne sont pas dans le voisinage à k sauts du nouveau *clusterhead* essayent de rejoindre d'autres *clusters*.

3.4.4 Conclusion

La définition des règles pour gérer la dynamique du réseau assure la maintenance de sa structuration en *clusters*. Mais comme ont indiqué les auteurs de k-lowestID, définir les règles n'est qu'une étape. Il est indispensable de déployer des techniques plus spécialisées et plus performantes en termes de qualité des *clusters* et surtout en termes de surcoût en signalisation. L'impact de la signalisation durant la phase de maintenance sur les performances du réseau est encore un sujet ouvert, alors que la phase de maintenance dure plus longtemps que la première phase de formation des *clusters*. Donc son effet sur l'état du réseau est plus important. L'évaluation d'une procédure de maintenance dans un environnement mobile relève une deuxième problématique : comment modéliser la mobilité des noeuds sur le terrain ? Avant d'étudier l'effet de la maintenance sur les performances du réseau, on va étudier tout d'abord la modélisation de la mobilité dans les réseaux MANETs.

Chapitre 4

Modélisation de la mobilité dans un MANET

Sommaire

Une catastrophe peut avoir lieu sur des zones complètement dépourvues d'infrastructure de télécommunications. Elle peut aussi mettre hors service l'infrastructure existante. Dans les deux cas, un service de communications d'urgence dans la zone sinistrée est indispensable : il permet aux équipes de secours de coordonner leurs interventions et aux autorités d'informer l'ensemble de la population affectée (recommandations, évolution de la situation, etc.).
Les services de secours utilisent actuellement des systèmes de communications professionnels à ressources partagées (standard TETRA en Europe). Ces systèmes reposent sur un

contrôle centralisé pour la transmission des ordres, la définition des groupes et la gestion des ressources du réseau et des équipes sur le terrain. Face au besoin de capacités accrues pour le transfert de données dans un environnement mobile, de nouvelles architectures basées sur la technique ad hoc ont été proposées. Un aperçu de quelques projets est donné dans le paragraphe 2.4.1.

Un réseau ad hoc présente plusieurs avantages, notamment la flexibilité et la rapidité de mise en place. Les informations sont routées de proche en proche vers la destination, en utilisant les noeuds intermédiaires comme relais. Sans besoin d'une administration centralisée, les noeuds coopèrent pour découvrir et utiliser les ressources disponibles.
Si le mode de fonctionnement ad hoc est intéressant, il pose néanmoins plusieurs défis techniques, notamment dans un MANET où les noeuds sont mobiles et peuvent changer librement de position. Des solutions ont été proposées pour réaliser la flexibilité de ce type de réseau et pour résoudre plusieurs problèmes liés au routage, aux interférences radio, à la consommation d'énergie, à la synchronisation et à la découverte de services. Quelques solutions sont présentées dans le paragraphe 2.3.

Le réseau déployé pour les communications d'urgence doit répondre à certaines exigences spécifiques : la sûreté de fonctionnement, la tolérance aux pannes, l'accès rapide au canal radio, la disponibilité des ressources et interopérabilité [72]. Afin d'utiliser un réseau ad hoc pour les communications d'urgence, les solutions techniques choisies doivent donc répondre à ces exigences spécifiques. Or les performances d'une solution technique dépendent fortement du scénario de déploiement : l'environnement (ex : urbain, forêt), le type de données (voix, vidéo, web) et le schéma de déplacement des utilisateurs. Dans ce chapitre, nous nous intéressons plus particulièrement à la modélisation des déplacements des utilisateurs. Pour l'évaluation des protocoles de routage par exemple, l'importance de la mobilité est présentée dans le RFC 2501 de l'IETF [73].

La simulation des performances d'une solution technique repose sur des modèles qui sont des représentations logiques décrivant le fonctionnement du système. Par exemple, le modèle de Weissberger [74] et la distribution de Rayleigh [75] sont des modèles analytiques qui caractérisent les affaiblissements dans un canal radio. Tout comme le canal radio, les déplacements des utilisateurs peuvent être décrits grâce à un modèle de mobilité.
Un modèle est une image réduite et représentative du réel. Il doit être suffisamment représentatif de la réalité pour faire apparaître les propriétés du scénario et en même temps relativement simple pour éviter la complexité des calculs. Si on se limite à une représentation simplifiée et générique, la caractérisation de la mobilité peut être faite assez convenablement grâce à un modèle analytique. Mais cette simplification peut estomper certaines propriétés du fonctionnement réel du système et la perte d'information

peut compromettre les résultats d'évaluation des choix techniques [76][77]. L'efficacité du modèle et sa pertinence dépendent donc du compromis fait entre la simplicité et la représentativité.

Dans ce chapitre, nous présentons quelques modèles génériques et montrer l'importance de développer des modèles dédiés aux scénarios de déploiement. On s'intéresse particulièrement aux scénarios de communications d'urgence.

4.1 État de l'art des modèles de mobilité

Un modèle de mobilité représente grâce à des équations mathématiques, la manière dont évolue la topologie du réseau. Il ne se limite pas à décrire les déplacement des noeuds dans le réseau. Il couvre aussi d'autre aspects tels que l'arrivée et le départ des noeuds. Un modèle de mobilité peut être soit générique en s'appliquant à un large spectre de scénarios, soit dédié à un scénario bien particulier.

4.1.1 Modèles de mobilité génériques

On distingue deux catégories de modèles génériques : les modèles de mobilité d'entité et les modèles de mobilité de groupe.

4.1.1.1 Modèles de mobilité d'entité

Dans ce type de modèles, le mouvement de chaque noeud est indépendant des autres noeuds du réseau.

4.1.1.1.1 Random Walk Le modèle de la marche aléatoire (*Random Walk*) représente un système possédant une dynamique discrète. Cette dynamique est composée d'une succession de pas aléatoires, totalement décorrélés les uns des autres. Cette dernière propriété, fondamentale, est appelée le caractère markovien du processus. Elle signifie intuitivement qu'à chaque instant, le futur du système dépend de son état présent, mais pas de son passé. Ce modèle a été initialement décrit par Einstein en 1926 pour représenter le mouvements de certaines particules dans la nature [78].

Chaque noeud choisit une direction et une vitesse de manière complètement aléatoire dans les intervalles $[0, 2\pi]$ et $[v_{min}, v_{max}]$ respectivement. Le changement de direction et de vitesse a lieu soit à chaque intervalle de temps t, soit quand une certaine distance d est parcourue. L'inconvénient majeur d'un tel schéma est le changement de position qui peut

être très brutal. Plusieurs autres modèles dérivent de ce modèle : *Random Gauss-Markov* [79] et *Markovian Model* [80].

4.1.1.1.2 Random WayPoint C'est l'un des modèles de mobilité les plus utilisés dans la simulation des réseaux ad hoc mobiles. En faisant l'inventaire de toutes les conférences MobiHoc entre 2000 et 2005, Kurkowski [81], a montré que 64% des contributions utilisant un modèle de mobilité sont basées sur *Random WayPoint*.
Dans ce modèle, le mouvement du noeud est une alternance entre des temps de pause et des temps de déplacement. Durant les temps de pause, le noeud reste stationnaire pendant une certaine période de temps. Ensuite, il se dirige vers une nouvelle destination choisie aléatoirement. La vitesse de déplacement est uniformément choisie dans l'intervalle [0, v_{max}]. Le noeud prend une pause en arrivant à destination, avant de reprendre pour se diriger vers une autre destination et ainsi de suite.

Certaines études consacrées à l'étude du comportement de ce modèle ont montré que la modèle, dans sa forme courante, n'atteint pas un état d'équilibre : la vitesse diminue progressivement pendant que la simulation progresse [82][83]. Cette propriété peut fausser l'évaluation des performances. En outre, les noeuds tendent à se déplacer au centre de la surface de simulation [84]. Pour contourner ce problème, le modèle *Random Direction* [85] force les noeuds à atteindre le bord de la surface de simulation avant de changer de direction et de vitesse.

4.1.1.1.3 Boundless Simulation Area Les modèles de mobilité spécifient généralement une surface de simulation et les noeuds sont contraints de rester à l'intérieur de cette surface. Le problème avec ce type de modèles est la convergence des noeuds vers le centre de la surface de simulation. Haas [86] propose le modèle *Boundless Simulation Area* qui permet de s'affranchir des bordures de la surface de simulation : le noeud qui atteint la bordure réapparaît du côté opposé. Les bordures sont donc reliées deux par deux et une surface rectangulaire prend la forme d'un tore.
Dans ce modèle, il existe une relation de dépendance entre la direction et la vitesse actuelles et les précédentes. On note $\bar{v} = (v, \theta)$ le vecteur de vitesse et (x, y) la position. v_{max} est la vitesse maximale. La relation de dépendance temporelle s'exprime de la façon suivante :

$$\begin{aligned} v(t+\Delta t) &= \min[\max(v(t)+\Delta v, 0), v_{max}]; \\ \theta(t+\Delta t) &= \theta(t)+\Delta\theta; \\ x(t+\Delta t) &= x(t)+v(t)\times\cos\theta(t); \\ y(t+\Delta t) &= y(t)+v(t)\times\sin\theta(t). \end{aligned}$$

Δv est la variation de la vitesse ; elle est uniformément distribuée dans l'intervalle $[-A_{max} \times \Delta t, A_{max} \times \Delta t]$. A_{max} est l'accélération maximale du noeud. $\Delta\theta$ est la variation de direction : elle est uniformément distribuée dans l'intervalle $[-\alpha \times \Delta t, \alpha \times \Delta t]$. α est le changement angulaire maximal.

Dans les modèles de mobilité décrits ci-dessus, la position de chaque noeud est définie indépen-demment des autres. En revanche, il existe certains scénarios, où les noeuds entretiennent entre eux certaines relations de dépendance. On définit alors les modèles de mobilité de groupe.

4.1.1.2 Modèles de mobilité de groupe

Dans plusieurs situations telles que le visites des musées et les champs de bataille, la notion de groupe intervient. Les noeuds sont divisés en plusieurs groupes et les déplacements des noeuds du même groupe sont corrélés. Les modèles de mobilité de groupe permettent de représenter grâce à des lois mathématique ces relations de dépendance spatiale.

4.1.1.2.1 Modèles de Sanchez Sanchez [87] propose trois modèles représentant trois applications différentes.

Column Mobility Model reproduit les opérations de recherches d'objets ou de personnes. Les membres du même groupe s'alignent et se déplacent en suivant une colonne déterminée. On définit une grille de référence initiale et à chaque point est associé un point de référence.

A un instant t, le noeud i modifie la position de son point de référence RP_i^t selon la relation suivante :

$$RP_i^t = RP_i^{t-1} + \alpha_i^t,$$

où α_i^t est un vecteur prédéfini.

Sur la grille, chaque noeud choisit aléatoirement une position P_i^t autour de son propre point de référence RP_i^t.

$$P_i^t = RP_i^{t-1} + w_i^t,$$

Nomadic Community Mobility Model est différent du précédent du fait que les noeuds du même groupe partagent le même point de référence. Le groupe de noeuds se déplace donc d'une façon collective d'un point à un autre et à l'intérieur du groupe, les noeuds bougent en utilisant un modèle de mobilité individuelle qui s'exprime selon la relation suivante :

$$P_i^t = RP_i^{t-1} + w_i^t,$$

sachant que tous les noeuds partagent le même point de référence.

Pursue Mobility Model est une variante du *Nomadic Community Mobility Model*. Il limite les mouvements aléatoires des noeuds pour maintenir une éventuelle poursuite. Les membres du groupe essayent d'intercepter une cible qui suit le modèle de mobilité Random WayPoint. Donc la position d'un noeud i s'exprime comme suit :

$$P_i^t = P_i^{t-1} + v_i^t(P_{target}^t - P_i^{t-1}) + w_i^t,$$

où P_{target}^t est une estimation de la position de la cible à un instant t et w_i^t est une déviation aléatoire.

4.1.1.2.2 Reference Region Le *Reference Region Mobility Model* [88] essaie de reproduire le mode opérationnel des équipes sauvetage et les scénarios d'exploration scientifique. Il comporte deux niveaux de mobilité : il représente à la fois la mobilité du groupe et la mobilité individuelle d'un noeud au sein d'un groupe. Le groupe est vu comme un noeud, en faisant abstraction à son centre de gravité. Il est représenté par un point logique appelé point de référence, possédant son propre schéma de mobilité : position, direction et vitesse. Aux alentours du point de référence, une zone délimite les positions possibles de chaque noeud du groupe. On parle alors de région de référence.
Initialement, une destination est assignée à chaque groupe qui procède par étapes pour atteindre cette destination. Des *checkpoints* sont définis entre la position initiale du groupe et sa destination et pour passer d'un *checkpoint* à un autre, il faut attendre l'arrivée de tous les noeuds du groupe qui peuvent avoir des vitesses de déplacement différentes. Chaque noeud de ce groupe suit donc ce point logique dans son mouvement. En attendant l'arrivée des autres noeuds de son groupe, le noeud suit le modèle de mobilité *Random WayPoint* à l'intérieur de la région de référence.

La position d'un *checkpoint* $i+1$ est définie en fonction de la position du *checkpoint* précédent i. Plusieurs variables définissent cette relation.

* γ et γ' sont deux variables aléatoires comprises entre 0 et 1.
  d'_x et d'_y définissent la trajectoire prise par les *checkpoints* entre la position initiale et la destination.
* x_d et y_d définissent la position de la destination du groupe.

Cette relation entre les positions successives du point de référence s'exprime de la manière suivante :

$$\begin{aligned} x_{i+1} &= (x_d - x_i) \times d'_x \times \gamma + x_i; \\ y_{i+1} &= (y_d - y_i) \times d'_y \times \gamma' + y_i. \end{aligned}$$

Les groupes peuvent fusionner et se diviser. Une fois arrivé à destination, un groupe reste pendant un certaine période de temps τ avant d'aller joindre un autre groupe. Les deux groupes fusionnent et se dirigent ensemble vers la même destination. Pour simuler l'attribution de nouvelles tâches, de nouvelles destinations sont générées périodiquement. S'il existe un groupe au repos, il se dirige vers la nouvelle destination D. Sinon, le groupe le plus proche de cette destination est divisé en deux groupes et chacun est associé à une destination différente D et D'.

Le *Reference Region Mobility Model* (RRGM) dérive d'un autre modèle de mobilité défini par Hong en 1999 : le *Reference Point Mobility Model* (RPGM) [89]. Dans RPGM, le point de référence est défini pour chaque noeud d'un groupe, tandis que dans RRGM, le point de référence est défini pour tout le groupe.

4.1.2 Modèles de mobilité dédiés

Avec les modèles de mobilité de groupe, on commence à avoir des modèles qui font apparaître des propriétés particulières et imaginer des scénarios d'applications spécifiques. Le *Reference Region Mobility Model* tente de reproduire le mode de fonctionnel des équipes de secours (pompiers, médecins, etc.) et le *Pursue Mobility Model* simule les missions de poursuite de cibles (forces de l'ordre, soldats, etc.). Mais ils restent tout de même des modèles génériques.
Avec les modèles dédiés, on part du scénario et on essaye de définir l'ensemble de ses descriptions mathématiques. La première catégorie de modèles dédiés prend en considération les propriétés de l'environnement du scénario. La deuxième catégorie prend en considération la distribution des rôles entre les personnes intervenant dans le scénario.

4.1.2.1 Modèles avec des restrictions géographiques

Les modèles avec des restrictions géographiques traduisent les contraintes de l'environnement par des conditions sur les trajectoires, en introduisant des obstacles ou en traçant les chemins que les noeuds peuvent emprunter.

4.1.2.1.1 Pathway Mobility Model Le moyen le plus simple de définir des restrictions géographiques est de définir à l'avance l'ensemble des trajectoires possibles. Un noeud est contraint alors à suivre l'une de ces trajectoires. On peut assimiler la surface de simulation à une carte de ville avec les routes et les chemins ouverts à la circulation. Dans le *Pathway Mobility Model*, développé par Tian en 2002 [90], on génère un graphe aléatoire où les sommets représentent les bâtiments de la ville et les arêtes les routes.

Initialement, les noeuds sont placés aléatoirement sur les arêtes. Ensuite, chaque noeud choisit une destination et se dirige vers cette destination en suivant le chemin le plus court. Une fois arrivé à destination, le noeud y reste pendant une certaine période de temps T_{pause}. Il choisit après une nouvelle destination et ainsi de suite jusqu'à la fin de la simulation.

FreeWay Mobility Model et *Manhattan Mobility Model* sont aussi des modèles qui imposent aux noeuds des itinéraires déterminés. Ces modèles, avec le *Pathway Mobility Model*, permettent de simuler la mobilité dans un environnement urbain mais ils possèdent néanmoins un certain caractère aléatoire, notamment dans la génération des destinations et du graphe initial.

4.1.2.1.2 Modèles de Johansson Une deuxième manière d'introduire les restrictions géographiques est d'inclure des obstacles dans la zone de simulation. Pour éviter ces obstacles, les noeuds sont contraints à changer de trajectoires. Il est intéressant de noter que l'introduction des obstacles n'a pas un effet sur la mobilité uniquement mais aussi sur d'autres aspects tels que la propagation radio. Les obstacles causent l'affaiblissement de la puissance des signaux et créent des effets de masque, pour les communications *indoor* et *outdoor*.

Johansson [76] a développé trois modèles correspondant à trois scénarios différents.

- Couverture d'un évènement : les noeuds sont très mobiles.
- Conférence : la plupart des noeuds est statique et une petite partie est mobile.
- Opérations de secours : une partie des noeuds est très dynamique et l'autre a une mobilité réduite.

Dans tous les scénarios précédents, des obstacles en formes de rectangles (en 2D) sont aléatoirement placés sur la surface de simulation. Un noeud doit donc bien choisir sa trajectoire pour éviter une collision avec ces obstacles. En outre, si un signal rencontre un obstacle, il est totalement absorbé. Donc si un obstacle est situé entre deux noeuds, ces derniers ne peuvent plus communiquer jusqu'à ce que l'un deux sort de la zone de masquage.

Dans les trois modèles de Johansson, l'obstacle modifie la trajectoire d'un noeud, au cas où ces deux derniers se croisent. Dans [91], les auteurs proposent de combiner l'introduction d'obstacles avec la définition des trajectoires possibles pour les noeuds. Au lieu de se déplacer de manière aléatoire et juste éviter les obstacles, les noeuds suivent des chemins tracés entre les bâtiments. Après une distribution aléatoire des obstacles (bâtiments), le graphe des itinéraires est généré grâce à la décomposition de Voronoï [92]. La ligne tracée

entre deux obstacles se situe sur la médiatrice qui les sépare.

Les modèles avec des restrictions géographiques essayent de faire apparaître certaines propriétés du scénario, notamment celles de l'environnement. Mais on a garde encore une grande part d'aléatoire pour appliquer ces modèles à des scénarios d'urgence. On a donc besoin de représentations qui s'approchent plus de la réalité. Mais il ne faut pas oublier que si on introduit plus de réalisme dans les scénarios, on ajoute plus de complexité aux modèles. On peut observer cette complexité avec les deux modèles suivants dédiés aux scénarios d'urgence.

4.1.2.2 Modèles dédiés aux situations d'urgence

4.1.2.2.1 Urgence médicale Le modèle proposé par Aschenbruck et son équipe [93] reproduit un scénario d'urgence médicale. La surface de simulation est divisée en plusieurs zones, possédant chacune son propre modèle de mobilité. Une zone appartient à une de ces cinq classes tactiques suivantes : le lieu de l'incident (LI), le point de stationnement des ambulances (PSA), le poste de commande (PC), l'endroit des premiers soins (EPS) et la station d'évacuation sanitaire (SES). On note l_r la classe tactique à la quelle appartient la zone $r \in R$. On a donc $l_r \in \{LI, PSA, PC, EPS, SES\}$.

On distingue deux types de noeuds : noeuds statiques et noeuds de transport. Les noeuds statiques N_r^{stat} sont assignés à une seule zone tactique r et ne peuvent pas en sortir. Les noeuds de transport N_r^{trans} peuvent passer d'une zone à une autre en passant par les points d'entrée et de sortie. Les zones de commande et d'évacuation comportent uniquement des noeuds statiques et les zones d'incident des noeuds de transport. On a donc

$$\begin{aligned} l_r &\in \{PC, SES\} : N_r^{trans} = \emptyset, \quad \forall r \in R; \\ l_r &\in \{LI\} : N_r^{stat} = \emptyset, \quad \forall r \in R. \end{aligned}$$

Les noeuds dans le modèle de mobilité *Random WayPoint* ont tendance à converger vers le centre la zone de simulation. Les noeuds statiques qui effectuent leurs missions à l'intérieur de leurs zones, suivent donc ce modèle de mobilité. Un noeud recevant un ordre se dirige vers une destination et y reste pendant un certain intervalle de temps pour effectuer sa mission. Il change de position dès la réception d'un nouvel ordre.

Chaque noeud de transport possède un cycle de déplacement qui dépend de la classe tactique de sa zone :

– Lieu d'incident ($r \in R | l_r = LI$) : le noeud commence par se positionner sur le point de sortie de sa zone a_r. Il choisit ensuite un point p_{rand} à l'intérieur de la zone et se déplace vers ce point. Il retourne après au point a_r, marque un temps de pause, choisit

aléatoirement une zone de premiers soins EPS_{rand} et se dirige vers le point d'entrée de cette dernière $e_{EPS_{rand}}$. Finalement, il marque un temps de pause au point $e_{EPS_{rand}}$ avant de retourner vers le point a_r. Ce cycle modélise un noeud transportant un blessé du lieu de l'incident vers un autre (plus sûr) pour recevoir les premiers soins.
- Endroit des premiers soins ($r \in R | l_r = EPS$) : le noeud suit un cycle similaire au précédent. La différence est que les blessés sont transportés vers une zone d'évacuation sanitaire.
- Point de stationnement des ambulances ($r \in R | l_r = PSA$) : le noeud commence par se positionner sur le point d'entrée de la zone e_r. Il choisit ensuite un point p_{rand} à l'intérieur de sa zone et se déplace vers ce point. Après un temps de pause, il sort de la zone de parking à travers son point de sortie a_r et se dirige vers le point de sortie $e_{SES_{rand}}$ d'une zone d'évacuation sanitaire choisie aléatoirement. Après un temps de pause au point $e_{SES_{rand}}$, il quitte le scénario de simulation. Ce cycle représente une ambulance qui attend au début une mission à l'intérieur du parking puis récupère un blessé pour le transférer vers l'hôpital.

Le modèle introduit aussi des obstacles sur la surface de simulation, que les noeuds doivent éviter. Un noeud de transport, pour passer d'une zone à une autre (en respectant son cycle de déplacement) est interdit de passer par une zone intermédiaire. Pour calculer les trajectoires des noeuds, le modèle génère un graphe de visibilité : les sommets des obstacles, les points d'entrée et de sortie et les sommets des différentes zones forment les sommets de ce graphe. Une arête est construite entre deux sommets si elle ne croise pas l'intérieur d'un obstacle ou une zone.

Le modèle précédent est focalisé sur la modélisation des opérations de secours médicaux et introduit un certain niveau de complexité. On va voir dans l'exemple suivant comment la complexité croît proportionnellement avec le niveau de réalisme du modèle.

4.1.2.2.2 Premiers secours Dans [94], la subdivision ne concerne pas les espaces uniquement mais les rôles aussi. Chaque noeud a donc plusieurs attributs qui permettent de le caractériser :
- Structure : est l'organisme auquel le noeud est affilié (médecins, pompiers, etc.).
- Rôle : indique le domaine d'expertise : l'ensemble des évènements et incidents sur lesquels le noeud peut intervenir.
- Zone de couverture : contient la description de la zone dans laquelle le noeud peut intervenir sur une mission (surface, point d'entrée, points de sortie).
- Vitesse : est choisie aléatoirement dans un intervalle donné.

Les groupes dans ce modèle se forment de manière dynamique en s'associant pour réaliser une même mission. Dès que la mission est achevée, le groupe se dissout. Une mission est traduite par la notion d'évènements. Les évènements peuvent avoir lieu dans des endroits différents et sont déclenchés de manière aléatoire. On distingue deux types d'évènements :

- Les évènements d'attention exigent l'intervention des secouristes dans les plus brefs délais : une victime coincée, premiers soins pour les blessés. Dans ce cas, le noeud se déplace à l'intérieur de la zone de l'évènement : à côté de la victime ou du blessé. Un évènement d'attention disparaît dès qu'un nombre suffisant de secouristes est intervenu.
- Les évènements d'avertissement sont des évènements dangereux : explosion, déversement de produits chimiques. Dans ce cas, les noeuds ne peuvent pas entrer dans la zone concernée à moins qu'ils soient correctement équipés et protégés.

Comme pour les noeuds, les évènements sont caractérisés grâce aux attributs suivants :

- Position : l'endroit où l'évènement a lieu.
- Zone de portée : l'évènement est visible par tous les noeuds qui sont dans cette zone.
- Liste demandée : chaque évènement nécessite une liste de domaines d'expertise. Cette liste contient l'ensemble de rôles que l'évènement demande pour résoudre le problème associé.
- Liste interdite : chaque évènement interdit aussi à certaines personnes d'intervenir pour des raisons de sécurité.
- Zone interdite : elle délimite la zone qu'un noeud de la liste interdite doit éviter.
- Zone d'assistance : elle délimite la zone sur laquelle les unités de secours peuvent intervenir.
- Durée de vie : un évènement a une durée de vie donnée. Un évènement d'attention peut disparaître même si le travail n'est pas complètement achevé par manque de moyens. Il sera considéré comme un échec pour les équipes de secours.
- Effort nécessaire : chaque évènement d'attention nécessite des moyens pour l'achever.

Dans le modèle global du scénario, un sous-modèle d'interaction décrit la manière dont un noeud est affecté à un évènement donné. Un évènement attire les noeuds qui ont des rôles figurant dans sa liste demandée et avertit ceux qui ont des rôles figurant dans sa liste interdite.

Un noeud, en cherchant un évènement d'attention, se déplace de manière aléatoire dans sa propre zone de couverture. Un noeud prend conscience de l'évènement dès qu'il entre dans la zone de portée de ce dernier. Autour de l'évènement, le noeud voit un ensemble de zones d'assistance et de zones interdites. Cette vision n'est pas partagée par tous les noeuds : les unités de secours appartenant à plusieurs agences différentes et disposant de moyens différents, n'ont pas la même vision des zones interdites et des zones d'assistance.

Si le noeud *voit* une zone d'assistance nécessitant des efforts supplémentaires, il s'engage dans cette zone. Il choisit une destination aléatoire à l'intérieur de la zone d'assistance et applique l'algorithme de Dijkstra pour trouver le plus court chemin vers cette destination. Il doit éviter de passer par les zones interdites. Une fois la mission achevée, il entre en mode recherche d'évènements, et ainsi de suite.

Le modèle décrit un ensemble d'interactions entre les unités de secours et les missions sur la scène de l'incident. La notion de groupe n'existe que de manière temporaire par la relation qui relie les unités à une mission donnée. Malgré la définition de la notion d'organismes auxquels appartiennent les unités, il manque la définition concept de coordination entre les unités. Or, quand on parle de la mise en place de communications d'urgence, on parle de coordination des efforts et de coopération des équipes.

4.1.3 Synthèse sur les modèles de mobilité

Nous avons essayé à travers cette présentation de mettre en évidence l'importance du modèle de mobilité dans la simulation des scénarios. Les modèles d'entité définissent la mobilité de chaque noeud sans prendre en considération celle des autres. Dans ces modèles, la notion de coordination est complètement absente. Les modèles de groupe essayent de surmonter ce problème en définissant deux niveaux de mobilité : l'un qui définit la mobilité individuelle du noeud et l'autre qui exprime la relation des mobilités individuelles au sein du même groupe. Malgré la définition de la notion de groupe, ces modèles restent trop génériques pour les appliquer dans un scénario d'urgence.

Certains modèlent essayent d'intégrer l'environnement du scénario en introduisant des obstacles ou en définissant à l'avance l'ensemble des trajectoires possibles. Avec ces modèles, on commence à faire apparaître certaines propriétés du scénario. Mais les scénarios d'urgence avec la diversité des missions et l'hétérogénéité des équipes ne peuvent pas être réduits à des contraintes géographiques.

Développer un modèle qui soit représentatif exige la connaissance de la réalité : comment les équipes sont organisées sur le terrain et comment elles réagissent face à un danger donné ? Quels sont les dangers auxquels sont exposées les équipes et les victimes ? Qui est responsable de la gestion des équipes ? Quels sont les particularités du terrain (urbain, montagneux, etc.) ? La réponse à ces questions définit le modèle de manière logique dans une première étape avant de développer ses descriptions mathématiques dans une deuxième étape. Il est important de noter que le mode de fonctionnement des équipes de secours ne dépend pas uniquement du terrain ou de la nature de la mission en elle-même mais aussi de la réglementation en vigueur. En France, le référentiel FdF (Feux de Forêt)

formalise l'organisation des pompiers dans la lutte contre les FdFs et le dispositif ORSEC décrit l'organisation des secours en cas de catastrophe.

4.2 Environnement de simulation

4.2.1 Outil de simulation

Pour la caractérisation numérique du système, nous avons utilisé un simulateur à événements discrets, *SimCore*. Ce simulateur a été développé dans notre laboratoire et comporte plusieurs modules répartis en deux catégories : *core* et *custom*. Les modules *core* fournissent les outils de base pour la modélisation d'un système de communication générique et les modules *custom* définissent les fonctions spécifiques à notre système étudié. Le simulateur étant écrit en C++, les modules *core* correspondent à des classes qui seront dérivées lors de la création des modules *custom*.

Les modules *core* gèrent notamment l'ordonnancement des évènements. Ils définissent aussi la structure du système sous forme de noeuds et fournissent les fonctions nécessaires pour l'étude du graphe du réseau, la génération du trafic et la caractérisation du canal de transmission. Les modules *custom* implémentent l'algorithme de *clustering* KCMBC, les modèles de mobilité *Random Walk*, RRGM et *FireMobility* et deux procédures de maintenance qui seront décrites dans le chapitre 5. Le changement de position d'un noeud dans le modèle de mobilité et l'envoi de messages pour la formation et la maintenance des *clusters* sont traduits par des évènements discrets dont l'ordonnancement est géré par les modules *core*.

4.2.2 Vérification du programme

La vérification d'un programme ne se réduit pas à la détection des erreurs de programmation de point de vue purement informatique ; il est indispensable de vérifier les résultats rendus. Pour illustrer le travail que nous avons effectué sur papier et en informatique pour la vérification des modules *custom*, nous avons choisi comme exemple le module *clustering*. Ce module implémente l'algorithme KCMBC. La figure 4.1 montre un réseau de 39 noeuds répartis aléatoirement sur la surface de simulation. Les temps d'expiration des noeuds sont attribués de manière aléatoire également. Le tableau 4.1 représente la valeur du degré de chaque noeud du réseau, ainsi que son temps d'expiration (exprimé en s). La valeur de k est fixée à 3 et les noeuds dont le temps d'expiration est inférieur à 3.5 ne peuvent être élus comme *clusterheads*.

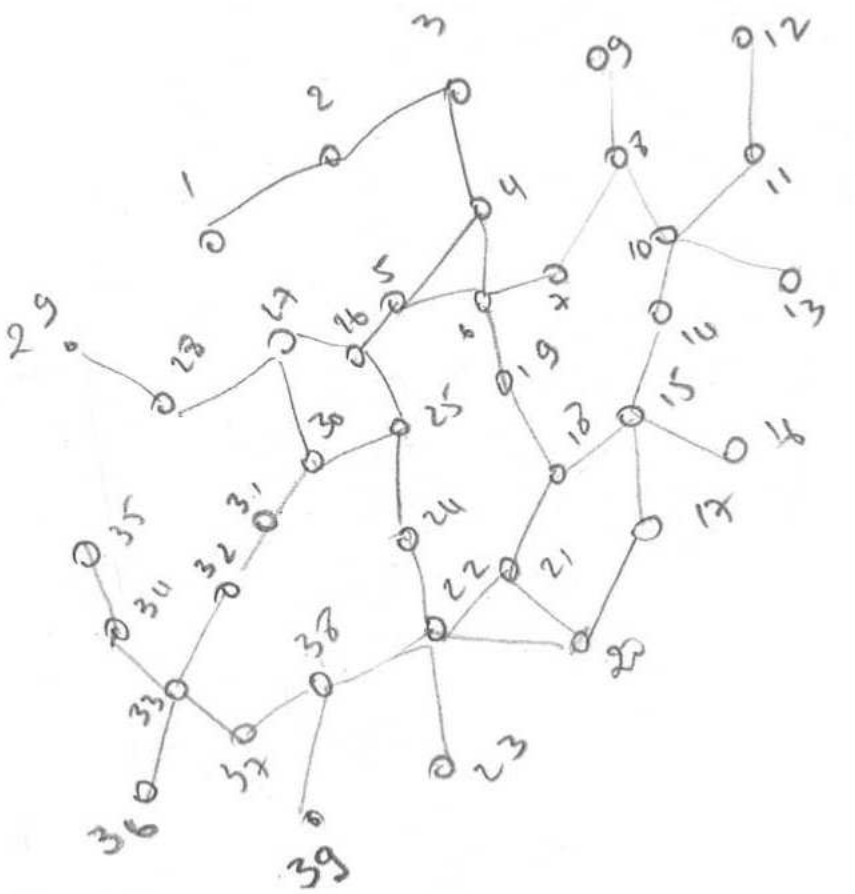


FIG. 4.1 – Vérification de l'algorithme KCMBC : topolohie du réseau.

Noeud	Degré	Temps d'expiration
1	1	6
2	2	8
3	2	4
4	3	3
5	3	10
6	4	5
7	2	4.5
8	3	1.5
9	1	0.5
10	4	8
11	2	9
12	1	2
13	1	1
14	2	9.5
15	4	8.5
16	1	5
17	1	5
18	3	7
19	2	8
20	3	2
21	3	3
22	4	9
23	1	1
24	2	0.5
25	3	8.5
26	3	5
27	3	4.5
28	2	4
29	1	3
30	3	1
31	2	6
32	2	8
33	4	10
34	2	9
35	1	4
36	1	3
37	2	1
38	3	2.5
39	1	1.5

TAB. 4.1 – Propriétés des noeuds : degré et temps d'expiration.

Les résultats obtenus sur papier de chaque tour de FloodMax et FloodMin sont présentés dans le tableau 4.2. Etant donné que la valeur de k est fixée à 3, nous avons 3 tours de FloodMax et 3 tours de FloodMin. *FMax 1* dénote le gagnant du 1^{er} tour de FloodMax.

Finalement, nous avons comparé les résultats obtenus sur papier (dernière colonne du tableau 4.2) à ceux obtenus par simulation (4.3). Nous avons notamment comparé la structure des *clusters* formés et vérifié qu'elle est la même dans les deux cas.

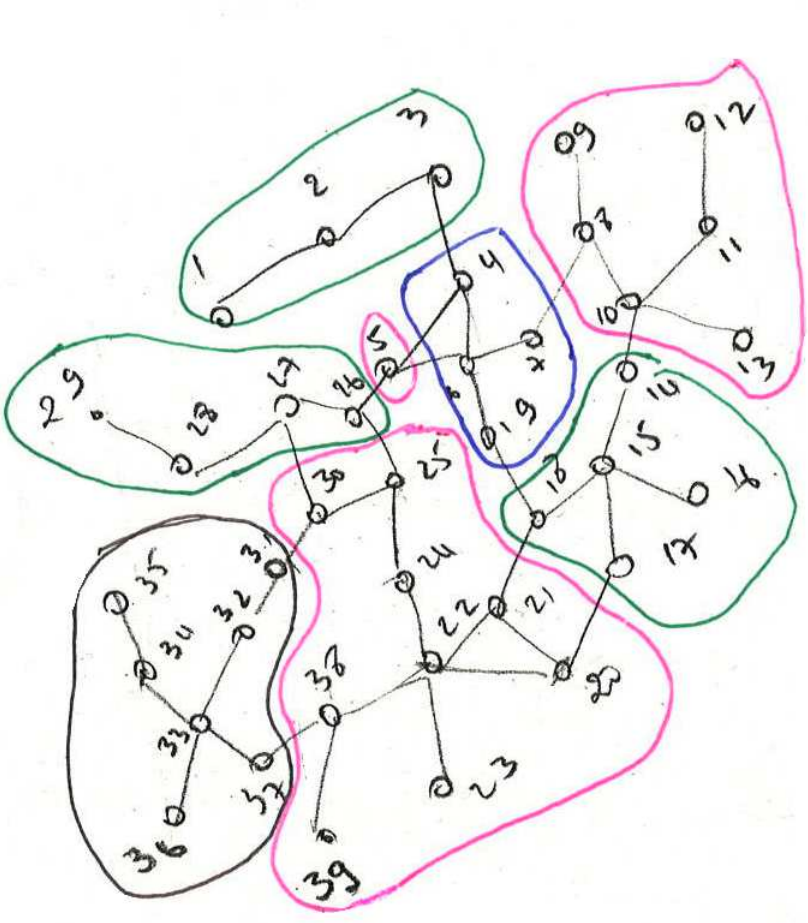


FIG. 4.2 – Vérification de l'algorithme KCMBC : résultat sous forme de *clusters*.

Noeud	FMax 1	FMax 2	FMax 3	FMin 1	FMin 2	FMin 3	Clusterhead
1	2	2	2	2	2	2	2
2	2	2	6	2	2	2	2
3	2	6	6	6	2	2	2
4	6	6	6	6	6	2	6
5	6	6	6	6	6	5	5
6	6	6	15	6	6	6	6
7	6	10	10	10	6	6	6
8	10	10	15	10	10	6	10
9	0	10	10	10	10	10	10
10	10	15	15	15	10	10	10
11	10	10	15	10	10	10	10
12	11	10	10	10	10	10	10
13	10	10	15	15	15	10	10
14	15	15	15	15	15	10	15
15	15	15	22	15	15	15	15
16	15	15	15	15	15	15	15
17	15	22	22	22	15	15	15
18	15	22	22	22	15	6	15
19	6	19	22	15	6	6	6
20	22	22	22	22	22	15	22
21	22	22	22	22	22	15	22
22	22	22	22	22	22	22	22
23	22	22	22	22	22	22	22
24	22	22	22	22	22	6	22
25	25	22	22	22	6	5	22
26	5	6	22	6	5	26	26
27	26	5	6	5	26	26	26
28	27	26	5	26	26	26	26
29	28	27	26	26	26	26	26
30	25	25	22	6	5	26	22
31	32	33	33	33	6	5	33
32	33	33	33	33	33	6	33
33	33	33	22	33	33	33	33
34	33	33	33	33	33	33	33
35	34	33	33	33	33	33	33
36	33	33	33	33	33	33	33
37	33	22	22	22	33	33	33
38	22	22	22	22	22	33	22
39	0	22	22	22	22	22	22

TAB. 4.2 – Vérification de l'algorithme KCMBC : le gagnant dans chaque tour de FloodMin et FloodMax et le clusterhead choisi.

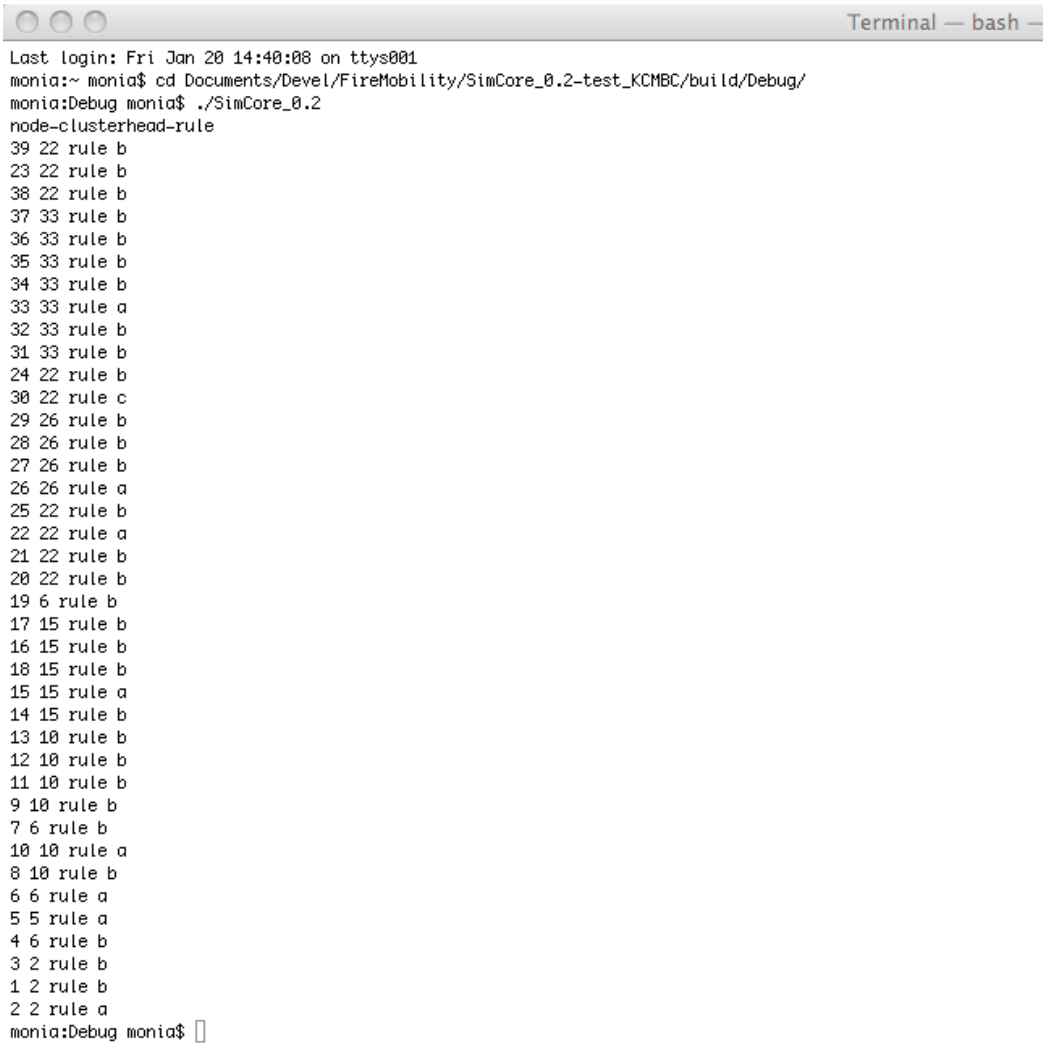


FIG. 4.3 – Vérification de l'algorithme KCMBC : résultat des simulations.

4.2.3 Validation des résultats

Maintenant, nous avons un système de communication correctement reproduit sur le simulateur. L'étape suivante consiste à bien paramétrer ce simulateur, notamment fixer la durée de simulation. Mais comment fixer cette durée ? La réponse à cette question dépend essentiellement des données à représenter. Étant donné que nous nous intéressons au problème de partitionnement, nous avons raisonné par rapport à la durée totale de partitionnement.

Nous avons choisi une durée de simulation égale à 10000 *s* et chaque simulation est répétée 10 fois. Afin d'évaluer la précision de l'estimation de la durée totale de partitionnement, nous avons examiné l'intervalle de confiance à 95%. La figure 4.4 présente la durée totale de partitionnement avec le modèle de mobilité FireMobility. La marge d'erreur est très petite, ce qui démontre que la mesure effectuée est relativement stable et que la simulation dure suffisamment longtemps pour obtenir des résultats représentatifs.

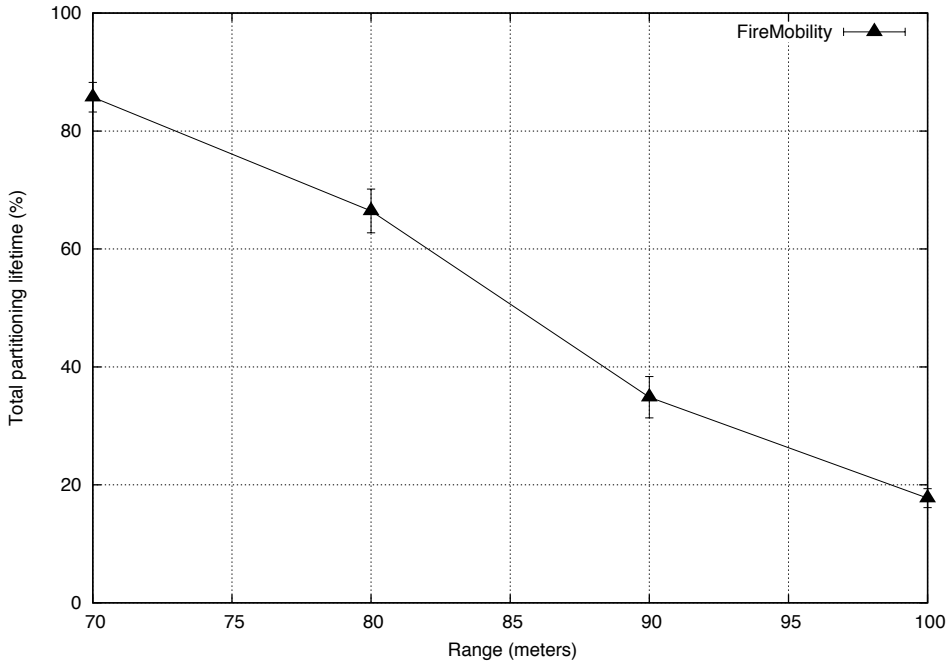


FIG. 4.4 – Validation des résultats : intervalles de confiance à 95%.

4.3 Comparaison des modèles de mobilité

Pour mettre en évidence l'impact du modèle de mobilité, Nous avons choisi de comparer trois modèles appartenant à trois catégories différentes. Dans la catégorie des modèles d'entité, le *Random Walk* est sélectionné. Il représente l'aléatoire total sans aucune notion de scénario. Le *Reference Region Group Model* (RRGM) est un modèle de groupe qui intègre la notion de dépendance spatiale entre les équipes de secours. Finalement, étant donné que nous nous intéressons à un scénario de lutte contre les feux de forêt, *FireMobility* est un modèle de mobilité dédié à ce scénario.

4.3.1 Description des modèles

Les trois modèles : *Random Walk*, RRGM et *FireMobility* sont très différents. Pour que la comparaison soit cohérente, les modèles doivent donc partager certaines propriétés communes. Nous commençons par décrire *FireMobility* pour expliquer comment les deux autres modèles seront paramétrés.

4.3.1.1 FireMobility

Le modèle de mobilité *FireMobility* reproduit le mouvement des équipes de secours lors des opérations de lutte contre les feux de forêt. Il a été développé dans notre laboratoire, en se basant sur des retours d'expérience de professionnels [95]. Le guide de la direction de la défense et de la sécurité civiles fournit aussi des informations sur le mode de fonctionnement des sapeurs-pompiers en France [96].
Les équipes de secours sont organisées de manière hiérarchique en quatre niveaux : colonne, groupe, camion citerne et binôme de sapeurs pompiers. Une colonne est formée d'un véhicule léger tout terrain (CARLV) et quatre groupes d'intervention feux de forêt. Chaque groupe est formé d'un véhicule léger tout terrain (ARLV), quatre camions citerne et quatre binômes de sapeurs pompiers. Une colonne est donc composée de $1+ 4 \times (1+4+4) = 37$ noeuds. La figure 4.5 montre l'organisation hiérarchique du réseau.

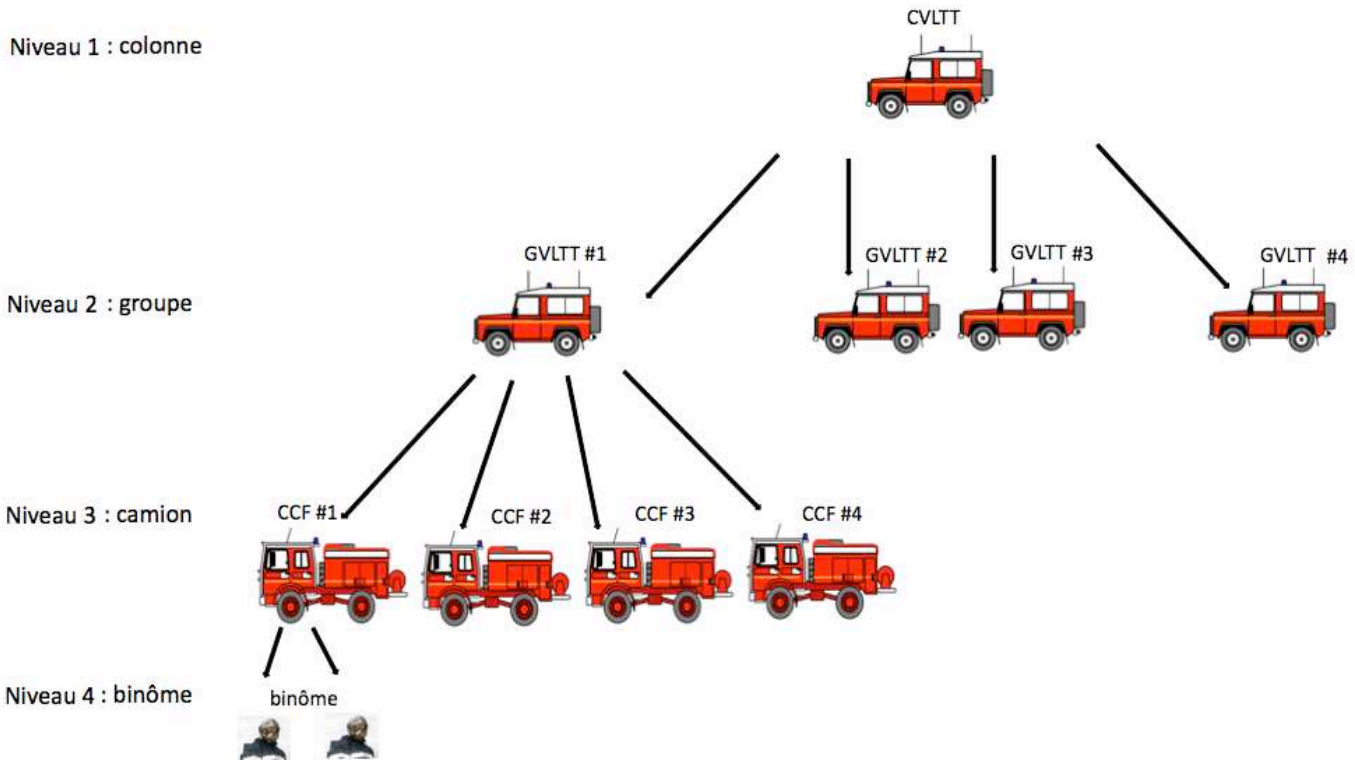


FIG. 4.5 – L'organisation hiérarchique du réseau.

La distribution des équipes sur le terrain suit cette hiérarchie : chaque niveau est tenu à respecter certaines distances par rapport au niveau supérieur. Il y a aussi un élément fondamental qui a un grand impact sur le positionnement des équipes. Ces dernières sont tenues à respecter les consignes de sécurité et garder certaines distances par rapport au feu. La figure 4.6 montre le schéma de déploiement d'un groupe et d'un véhicule de colonne.

Le déplacement du feu sur la surface de simulation implique parfois le redéploiement des équipes. La structuration du réseau en plusieurs niveaux et le mouvement du feu déterminent le schéma de mobilité des noeuds. Si un véhicule de groupe se trouve très près/loin du feu, il s'éloigne/s'approche en choisissant une nouvelle position. Les quatre camions citernes et les quatre binômes de sapeurs pompiers suivent le véhicule de leur groupe et choisissent à leur tour de nouvelles positions. la figure 4.7 donne un aperçu sur la manière dont la mobilité du feu et la hiérarchie du réseau impactent la mobilité des noeuds. Le véhicule du groupe situé en haut de la figure de gauche se trouve très loin du feu. Tout le groupe change de position et s'approche du feu en se mettant plus à gauche du feu (figure de droite). Pour la modélisation du déplacement du feu, la vitesse de ce dernier a été choisie en se basant sur une échelle d'intensité publiée dans [97]. Le tableau 4.3 illustre les paramètres de *FireMobility*.

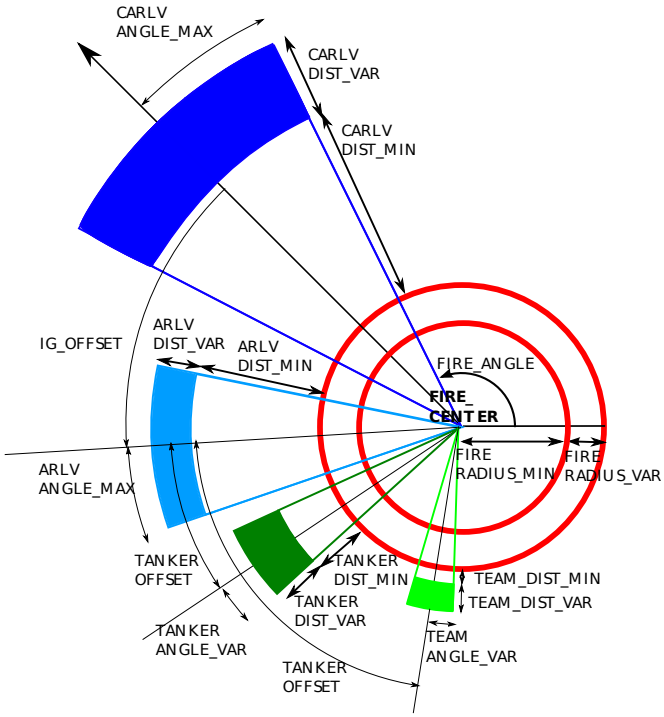


FIG. 4.6 – Schéma de déploiement d'un groupe et un véhicule de colonne autour du feu. IG : groupe d'intervention ; ARLV : véhicule léger tout terrain d'un groupe ; CARLV : véhicule léger tout terrain d'une colonne [95].

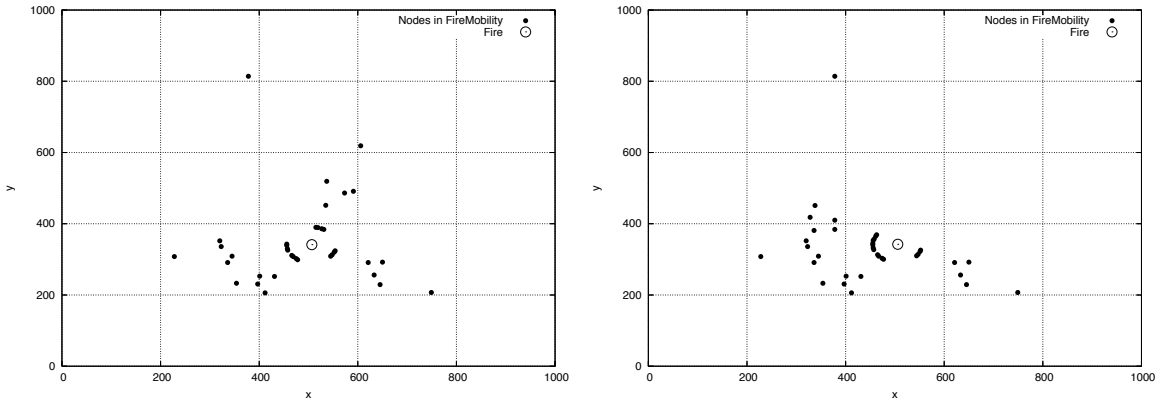


FIG. 4.7 – L'impact de la mobilité du feu et de la hiérarchie du réseau sur la mobilité des noeuds. Le feu est représenté par un cercle.

Paramètre	**Valeur**
Surface de simulation (m)	1000 x 1000
Rayon du feu (m)	50
Nombre de noeuds	37
Nombre de groupes	4
Vitesse du feu (m/s)	0.3
Vitesse moyenne des noeuds (m/s)	4.4
Distance entre le véhicule de colonne et le feu (m)	$[400, 600]$
Distance entre le véhicule de groupe et le feu (m)	$[150, 300]$
Distance entre le camion citerne de groupe et le feu (m)	$[100, 200]$
Distance entre le binôme de sapeurs pompiers le feu (m)	$[50, 60]$

TAB. 4.3 – Paramètres du modèle *FireMobility*.

4.3.1.2 RRGM

Le modèle RRGM, dans sa forme courante, a été décrit dans le paragraphe 4.1.1.2.2. Il définit un ensemble de groupes qui procèdent par étapes pour rejoindre une destination. Il intègre la notion de dépendance spatiale, étant donné que la mobilité individuelle est liée à la mobilité du groupe. En revanche, il n'inclut pas la notion hiérarchie et de distances entre les différentes équipes d'intervention.

RRGM a été adapté pour s'approcher d'un scénario de lutte contre les feux de forêt. Le feu est représenté par un disque qui se déplace de manière aléatoire sur la surface de simulation. Les groupes doivent rester autour du feu pour effectuer leur mission. On définit donc une zone d'intervention centrée sur le feu, avec un diamètre plus large [1]. Au départ, les noeuds forment au plus cinq groupes [2]. Ensuite, chaque chef du groupe choisit de manière aléatoire une destination à l'intérieur de la zone d'intervention. Pour la description de la mobilité du groupe vers une destination, le modèle implémenté reste fidèle aux formules définies dans le modèle RRGM originel.

A cause de la mobilité du feu, la destination d'un groupe peut se trouver en dehors de la zone d'intervention. Dans ce cas, le chef de groupe choisit une nouvelle destination sur laquelle tout le groupe met le cap. La figure 4.8 montre la répartition des groupes autour feu et le tableau 4.4 illustre les paramètres de simulation de RRGM.

[1] Le rayon de la zone d'intervention est égal à la distance maximale entre un groupe d'intervention et le feu.

[2] On définit 5 groupes pour reproduire le modèle avec 1 véhicule de colonne + 4 groupes d'intervention.

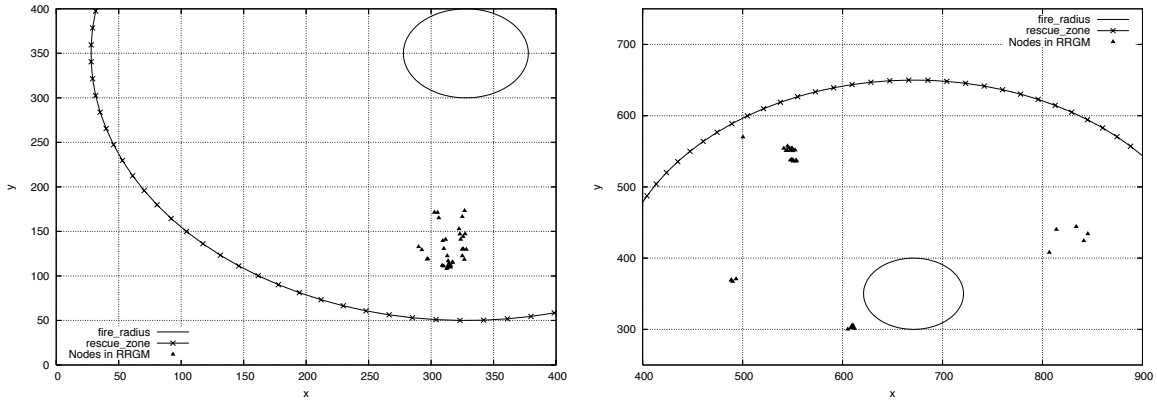


FIG. 4.8 – La répartition des groupes autour feu à un instant t=0 s (gauche) et t=9000 s (droite). Le cercle intérieur représente le feu et le cercle extérieur la zone d'intervention.

Paramètre	**Valeur**
Surface de simulation (m)	1000 x 1000
Rayon du feu (m)	50
Rayon de la zone d'intervention (m)	300
Nombre de noeuds	37
Vitesse du feu	0.3 m/s
Vitesse moyenne des noeuds	4.4 m/s

TAB. 4.4 – Paramètres de RRGM.

4.3.1.3 Random Walk

Le modèle *Random Walk* est un modèle d'entité : il n'intègre ni la notion de groupe, ni le concept de hiérarchie. Les noeuds sont uniformément distribués sur la surface de simulation et ensuite chacun choisit une direction et une vitesse de manière purement aléatoire. Une description détaillée du modèle est donnée dans le paragraphe 4.1.1.1.1. La figure 4.9 montre la répartition des noeuds sur la surface de simulation.

Le modèle *Random Walk* utilise trois paramètres : le nombre de noeuds, la vitesse et bien évidemment la taille de la surface de simulation. Les valeurs des deux premiers paramètres peuvent être fixées facilement en fonction de ceux de *FireMobility*. Fixer la valeur du dernier paramètre est toutefois moins évident. Le problème de *Random Walk* est qu'aucune relation n'existe entre les mobilités individuelles des noeuds. Chaque noeud choisit sa direction et sa vitesse indépendamment des autres noeuds. Donc il est difficile

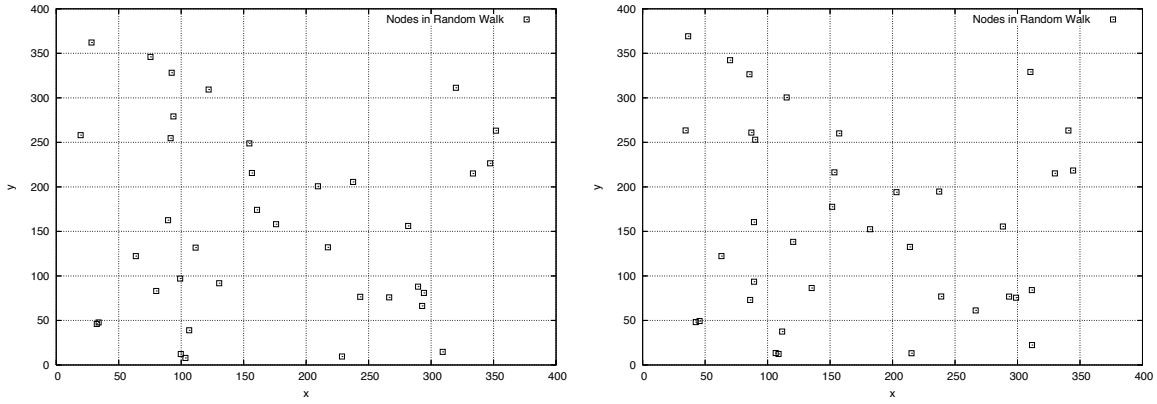


FIG. 4.9 – La répartition des noeuds sur la surface de simulation dans *Random Walk* à deux instants différents.

de procéder de la même manière qu'avec RRGM, où on a essayé de rapprocher ce dernier d'un scénario de lutte contre les feux de forêt.
Au lieu de se baser sur le mode de fonctionnement de *FireMobility* (FM), le rapprochement avec *Random Walk* (RW) peut s'appuyer sur les propriétés de partitionnement du réseau et plus particulièrement la durée totale de partitionnement (DTP). En utilisant la méthode des moindres carrés, on détermine la taille de surface de simulation qui réalise une déviation minimale. La formule suivante permet de calculer la déviation :

$$dev_j = \frac{\sqrt{\sum_{i \in \{70,80,90,100\}} (DTP_{(FM,i)} - DTP_{(RW,i,j)})^2}}{4}, \quad \forall j \in \{360 \times 360, 370 \times 370, 380 \times 380\},$$

où $i \in \{70, 80, 90, 100\}$ représente la portée radio et $j \in \{360 \times 360, 370 \times 370, 380 \times 380\}$ la taille de surface de simulation. Le tableau 4.5 montre que la déviation correspondant à une taille égale à 370×370 m minimise la déviation. La figure 4.10 présente l'évolution de la durée totale de partitionnement en fonction de la portée radio et le tableau 4.6 illustre les paramètres de simulation de *Random Walk*.

Surface de simulation en m	360 × 360	370 × 370	380 × 380
Déviation	3.01317	1.63634	3.45986

TAB. 4.5 – La déviation de la durée totale de partitionnement avec *Random Walk* par rapport à *FireMobility*.

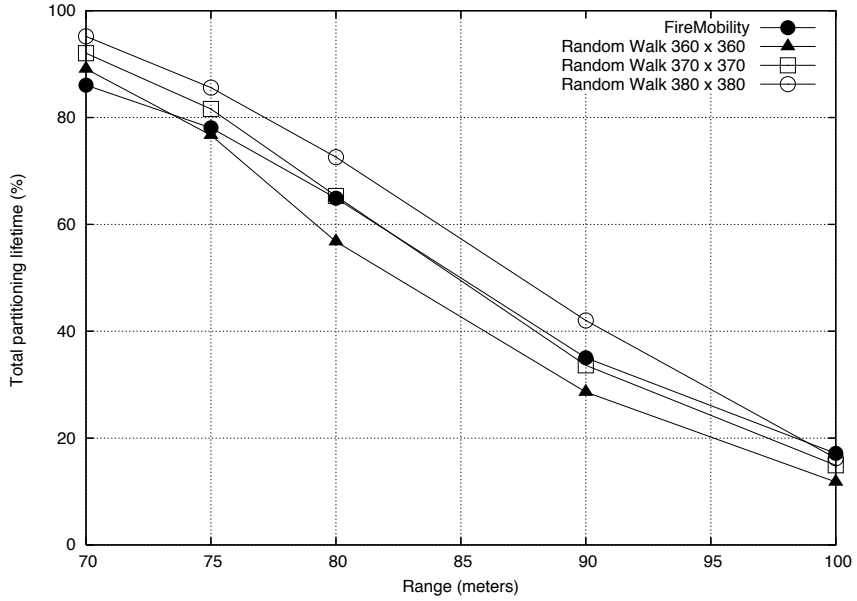


FIG. 4.10 – Durée totale de partitionnement en fonction de la portée radio : *FireMobility* et *Random Walk*.

Paramètre	**Valeur**
Surface de simulation (m)	370 x 370 m
Nombre de noeuds	37
Vitesse moyenne des noeuds	4.4 m/s

TAB. 4.6 – Paramètres de *Random Walk*.

4.3.1.4 Adaptation de Random Walk et RRGM

Les deux modèles de mobilité *Random Walk* et RRGM sont choisis pour être comparé à *FireMobility*. Ils appartiennent à des catégories différentes à celles de *FireMobility* (entité/groupe vs dédié). Pour donner du sens à cette comparaison, chacun des deux modèles

partage des paramètres communs avec *FireMobility*.

***FireMobility* / RRGM**	***FireMobility* / *Random Walk***
Nombre de noeuds	Nombre de noeuds
Vitesse moyenne des noeuds	Vitesse moyenne des noeuds
Surface de simulation	Durée totale de partitionnement
Zone de couverture des groupes	
Vitesse et rayon du feu	

TAB. 4.7 – Paramètres communs : *FireMobility* / RRGM et *FireMobility* / *Random Walk*.

4.3.2 Résultats de simulation : comparaison des modèles

Le but est de comparer deux à deux *FireMobility* à RRGM et *FireMobility* à *Random Walk* et de montrer l'intérêt d'un modèle dédié. La comparaison repose fondamentalement sur l'analyse des propriétés topologiques du réseau. La portée radio varie entre 70 m et 100 m, ce qui correspond aux valeurs typiques de la portée radio pour des communications *outdoor* avec la technologie *Wi-Fi*.

La comparaison entre de différents modèles dépend du contexte général et des applications du réseau. Nous rappelons que le problème de la modélisation de la mobilité a été posé quand nous avons abordé le sujet du choix des noeuds qui vont avoir accès au satellite (chapitre 3). Ce dernier, lui même a été évoqué quand nous avons exposé les problèmes rencontrés dans les réseaux ad hoc, plus particulièrement le problème du partitionnement du réseau.

L'étude de la technique du *clustering* et l'évaluation de l'algorithme KCMBC [3] n'ont un sens que lorsque le réseau est partitionné. C'est pour cette raison que nous nous focalisons essentiellement sur les propriétés du réseau en cas de perte de la connexité. Nous nous intéressons dans un premier lieu à la caractérisation temporelle du partitionnement. Nous étudions ensuite la caractérisation spatiale du réseau.

[3] KCMBC est l'algorithme de *clustering* que nous avons choisi pour résoudre le problème du choix des noeuds qui vont avoir accès au satellite.

4.3.2.1 Caractérisation temporelle

Nous commençons par comparer la durée totale du partitionnement (DTP), définie comme le pourcentage du temps durant lequel le réseau est partitionné par rapport à la durée de la simulation. La figure 4.11 montre que la DTP dans *Random Walk* est équivalente à celle de *FireMobility*, tandis que dans RRGM elle affiche une allure complètement différente. Pour les trois modèles, la DTP décroît progressivement en fonction de la portée radio. Mais la pente dans RRGM est très faible, ce qui signifie qu'avec ce paramétrage du modèle, la portée radio à un faible impact.

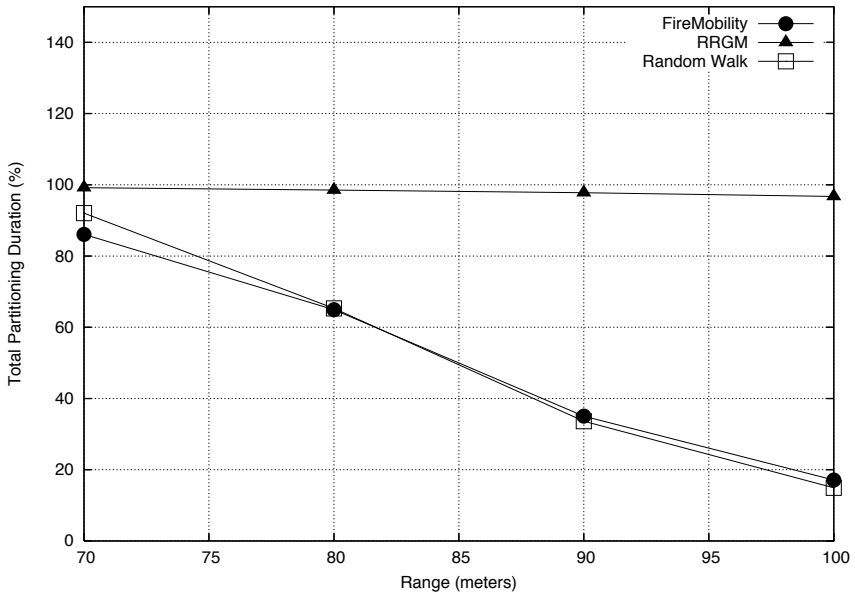


FIG. 4.11 – Durée totale de partitionnement en fonction de la portée radio pour *Random Walk*, RRGM et *FireMobility* (exprimée en pourcentage de la durée de la simulation).

Pour comprendre cette différence, nous examinons la manière dont les noeuds sont répartis sur la surface de simulation. Pour RRGM, nous allons regarder tout d'abord l'organisation des noeuds au sein de chaque groupe et ensuite la distribution des groupes dans la zone d'intervention autour du feu. Un groupe procède par étapes pour atteindre sa destination. Des *checkpoints* intermédiaires sont définis entre la position initiale du groupe et la destination finale. La figure 4.12 montre la progression d'un groupe de sa position initiale (à gauche en bas de la figure) vers sa destination (à droite en haut) en transitant par les *checkpoints* intermédiaires. Avant de passer d'un *checkpoint* à l'autre, on doit attendre l'arrivée de tous les noeuds du groupe. Une fois arrivés, les noeuds ne sont

autorisés à se déplacer qu'à l'intérieur d'une région de référence définie autour de chaque *checkpoint*. Avec ce mode de fonctionnement, les noeuds sont très rapprochés et restent très regroupés ; c'est ce qu'on peut observer dans la figure 4.8 où les groupes peuvent être assimilés à des points singuliers. Le regroupement des noeuds n'explique pas à lui seul les différences observées entre RRGM et *FireMobility*. Si les mouvements des noeuds sont corrélés au sein du même groupe, les déplacements des groupes sont complètement décorrélés. Dans le choix de sa destination et la définition de ses *checkpoints*, chaque groupe opère de manière indépendante des autres groupes du réseau. Dans la figure 4.8 représentant la distribution des noeuds dans la zone d'intervention, on observe que les groupes occupent des zones bien éloignées l'une de l'autre. Étant donné que les noeuds du groupe sont rapprochés et les groupes sont éloignés [4], le réseau est partitionné dans plus de 96% du temps quelque soit la portée radio (comprise entre 70 m et 100 m). La corrélation du mouvement des noeuds et l'indépendance des déplacements des groupes expliquent conjointement le faible impact de la portée radio sur la DTP dans RRGM.

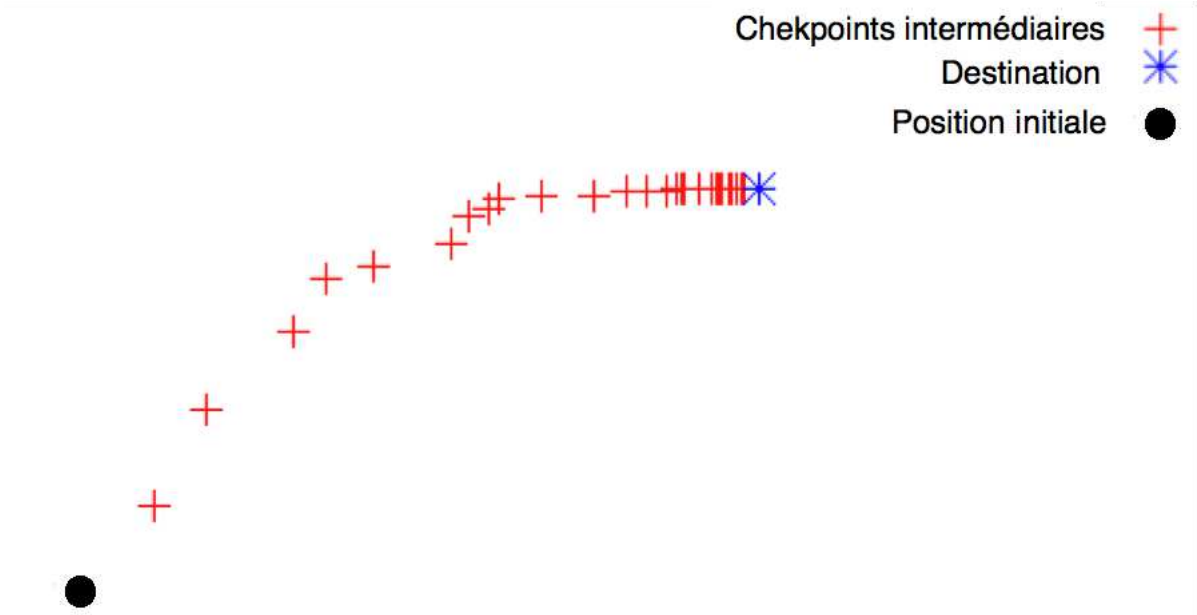


FIG. 4.12 – Progression par étapes vers la destination dans RRGM.

[4]Les groupes peuvent se rapprocher mais dans le réseau, il en existe toujours au moins deux qui sont éloignés.

Nous revenons à notre problème initial du choix des noeuds qui vont avoir accès au satellite. Dans notre définition du scénario de lutte contre les feux de forêt, le satellite a été proposé comme solution pour la restauration d'une défaillance du système due au partitionnement. Dans RRGM, la défaillance est quasi-permanente : le réseau est partitionné dans 96% du temps dans le meilleur des cas. Cette propriété a un effet important sur l'estimation du coût de la solution et notamment sur la bande passante allouée au niveau du satellite.
Nous allons à présent éclaircir comment un modèle d'entité, *Random Walk* peut avoir des similitudes avec un modèle de groupe *FireMobility*. La figure 4.9 montre que dans *Random Walk*, les noeuds sont uniformément distribués sur la surface de simulation et la notion de groupe n'existe pas. C'est donc la proportion entre le nombre de noeuds (petit) et la taille de surface de simulation (relativement large) qui entraîne une durée totale de partitionnement assez importante (90% de la durée de simulation pour une portée radio égale à 70 m).

Comptabiliser la totalité de la durée de partitionnement montre des ressemblances entre *Random Walk* et *FireMobility* quelque soit la valeur de la portée radio. On remarque qu'il y a aussi des similitudes entre les trois modèles pour une portée radio égale à 70 m. Mais l'examen de la distribution de la durée des évènements de partitionnement révèle de nouvelles différences entre les modèles. Nous appelons un évènement de partitionnement le fait de perdre la connexité du réseau. La figure 4.13 montre que pour *Random Walk*, un évènement de partitionnement sur quatre dure une seconde et le pourcentage des évènements qui durent plus de 10 s est insignifiant. Pour *FireMobility*, les évènements peuvent durer jusqu'à 1600 s et pour RRGM jusqu'à 10000 s, c'est-à-dire la durée de la simulation. Le phénomène de longues durées de partitionnement dans RRGM a été expliqué dans le paragraphe précédent. Nous nous intéressons donc à l'analyse du comportement des noeuds dans *Random Walk*.

La particularité de *Random Walk* est le caractère markovien des déplacements : à un instant $t + \Delta t$, chaque noeud choisit sa direction et sa vitesse de manière complètement aléatoire, indépendamment des choix faits dans le passé (à l'instant t). Donc l'état du réseau à un instant t est indépendant de ses états à des instants passés et futurs. Si le réseau est partitionné à un instant donné, on n'a aucune information sur son état précédent ou à l'instant suivant. C'est donc le caractère markovien de ce schéma de mobilité qui explique le pic observé pour une durée égale à une seconde. Les déplacements des noeuds sont décorrélés : chaque noeud choisit sa direction et sa vitesse indépendamment des choix faits par les autres noeuds. En outre, les changements de positions d'un instant à un autre sont un peu brusques. Avec une telle dynamique, même si un évènement de partitionnement a lieu, la connexité est très vite rétablie. La durée de ce rétablissement, comme pour le partitionnement peut ne durer que quelques secondes. Le caractère marko-

vien, la décorrélation des mouvements et la brutalité des changements expliquent l'allure de la courbe observée pour *Random Walk*.

On retourne au problème du choix des noeuds qui vont avoir accès au satellite. La forte dynamique du réseau dans *Random Walk* pose des problèmes de dimensionnement, notamment pour la convergence du système et l'allocation de la bande passante du satellite.

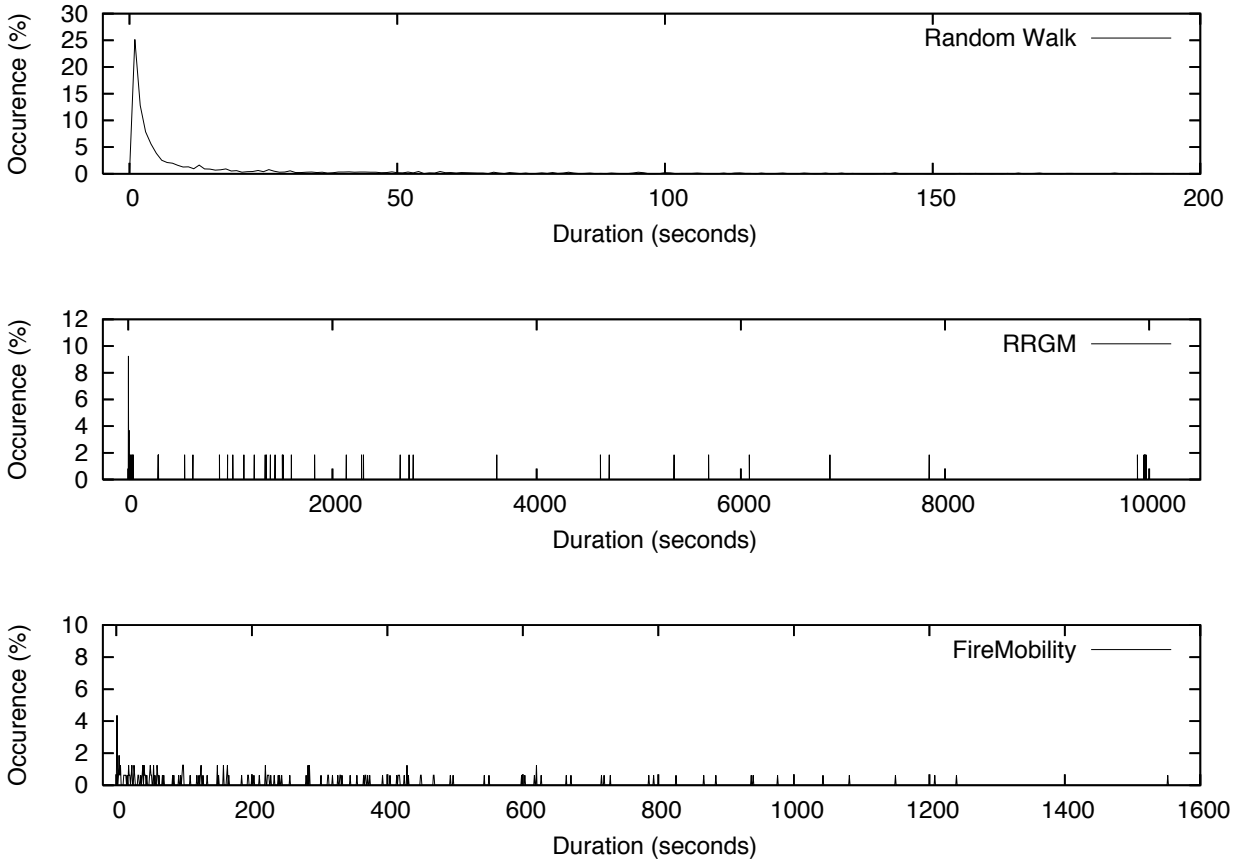


FIG. 4.13 – Distribution de la durée des évènements de partitionnement pour *Random Walk*, RRGM et *FireMobility*. La portée radio est fixée à 70 m.

4.3.2.2 Caractérisation spatiale

Après avoir établi la caractérisation temporelle du partitionnement, nous allons maintenant considérer la caractérisation spatiale du réseau. Nous allons particulièrement étudier

la durée de vie des liens et le degré des noeuds (le nombre de voisins). Ces deux critères, bien qu'ils soient classiques dans l'étude des propriétés spatiales du réseau, sont fortement liés au problème du choix des noeuds qui vont avoir accès au satellite et à la technique du *clustering*. En effet, comme nous l'avons vu dans le chapitre 3, KCMBC définit une métrique de choix composée où le degré du noeud et le temps d'expiration des liens interviennent. Le temps d'expiration n'est qu'une estimation de la durée de vie du lien.

L'étude de la durée de vie des liens combine la caractérisation temporelle du partitionnement et la caractérisation spatiale du réseau : un lien fait partie des éléments topologiques du réseau et nous proposons ici d'étudier la durée pendant laquelle un lien existe (durées cumulées pour toute la simulation en considérant les phases actives et inactives de ce lien). La figure 4.14 montre la distribution de la durée de vie des liens pour les trois modèles. Les liens dans *Random Walk* sont les moins stables et ceci est expliqué par la forte dynamique du réseau et la brutalité des changements de positions entre deux instants successifs. En comparant les deux modèles de mobilité de groupe, on remarque que les liens dans RRGM sont plus stables par rapport à *FireMobility* et 6% des liens peuvent même exister durant toute la simulation. Dans RRGM, les noeuds comme nous l'avons expliqué dans le paragraphe précédent, restent toujours regroupés au sein du même groupe, ce qui allonge la durée de vie des liens intra-groupe. Certes *FireMobility* définit la notion de groupe mais les noeuds sont dispatchés autour du feu de manière qu'à chaque noeud est associée une zone d'intervention. Les zones d'intervention du même niveau hiérarchique ne se superposent pas. Par exemple un groupe d'intervention i du niveau 2 (figure 4.5) agit sur une zone qui lui est attribuée et qui est différente de celle assignée à un autre groupe j. Les distances entre les noeuds appartenant au même groupe dans *FireMobility* sont relativement supérieures à celles qui existent entre les noeuds d'un même groupe dans RRGM. La figure 4.7 illustre la répartition des noeuds autour du feu dans *FireMobility*. En outre, la mobilité des noeuds dans *FireMobility* est très liée à celle du feu. La redistribution des noeuds est effectuée à chaque fois que les consignes sur les distances minimales et maximales par rapport au feu ne sont pas respectées. Ce mode de fonctionnement rend le réseau relativement plus dynamique par rapport à RRGM.

Le dernier point, le degré du noeud, se rapporte essentiellement à la manière dont les noeuds sont répartis sur la surface de simulation. On voit clairement sur la figure 4.15 que *Random Walk* applique une répartition uniforme des noeuds sur la surface de simulation ce qui reflété dans la distribution des degrés. La moyenne notée $\bar{X}$ et l'écart type du degré noté σ dans *Random Walk* sont les plus faibles par rapport aux deux autres modèles. Les écarts types dans RRGM et *FireMobility* ne sont pas très différents mais la grande divergence réside dans la moyenne des degrés. La notion de regroupement et de rapprochement des noeuds dans RRGM se traduit par un degré moyen plus large en le comparant à *Fi-*

reMobility. Même si les deux modèles définissent la notion de groupe, *FireMobility* ajoute la notion de hiérarchie. Chaque noeud est assigné à un niveau hiérarchique donné et obéit des règles qui dépendent de ce niveau. C'est pour cette raison que la distribution de *FireMobility* n'affiche que certaines valeurs particulières pour le degré.

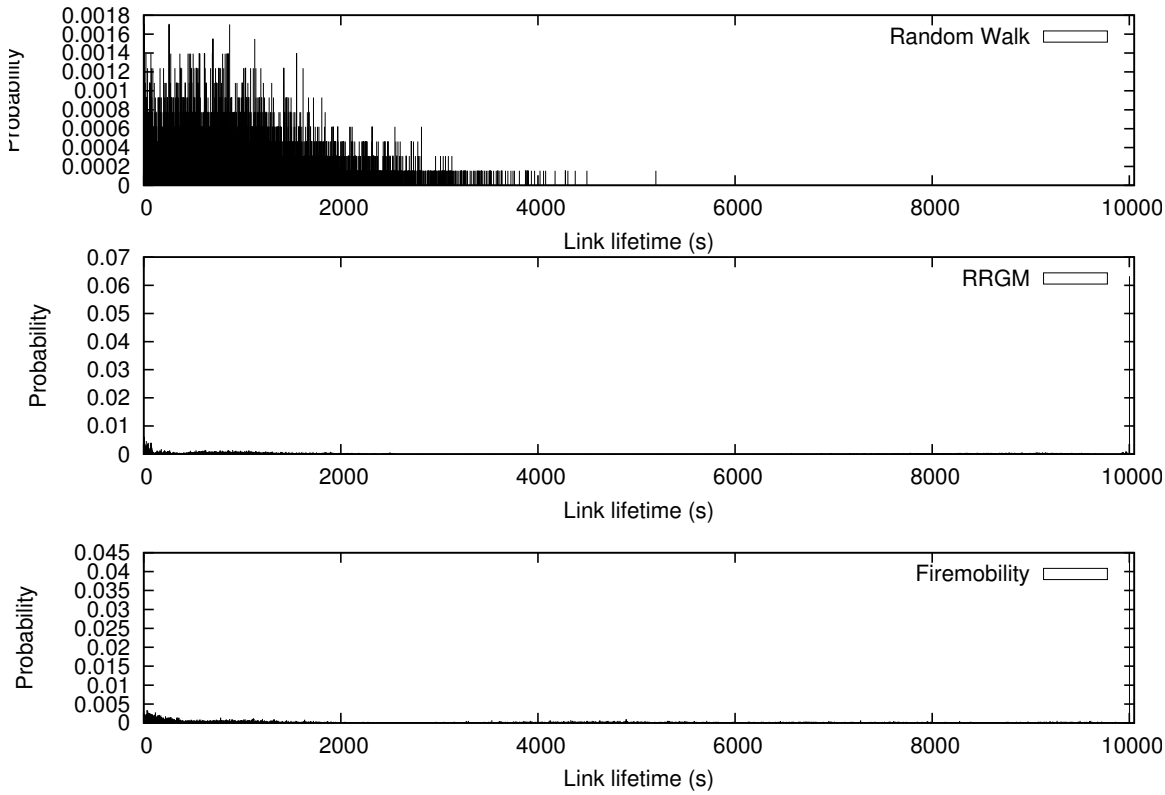


FIG. 4.14 – Distribution de la durée de vie des liens pour *Random Walk*, RRGM et *FireMobility*. La portée radio est fixée à 70 m.

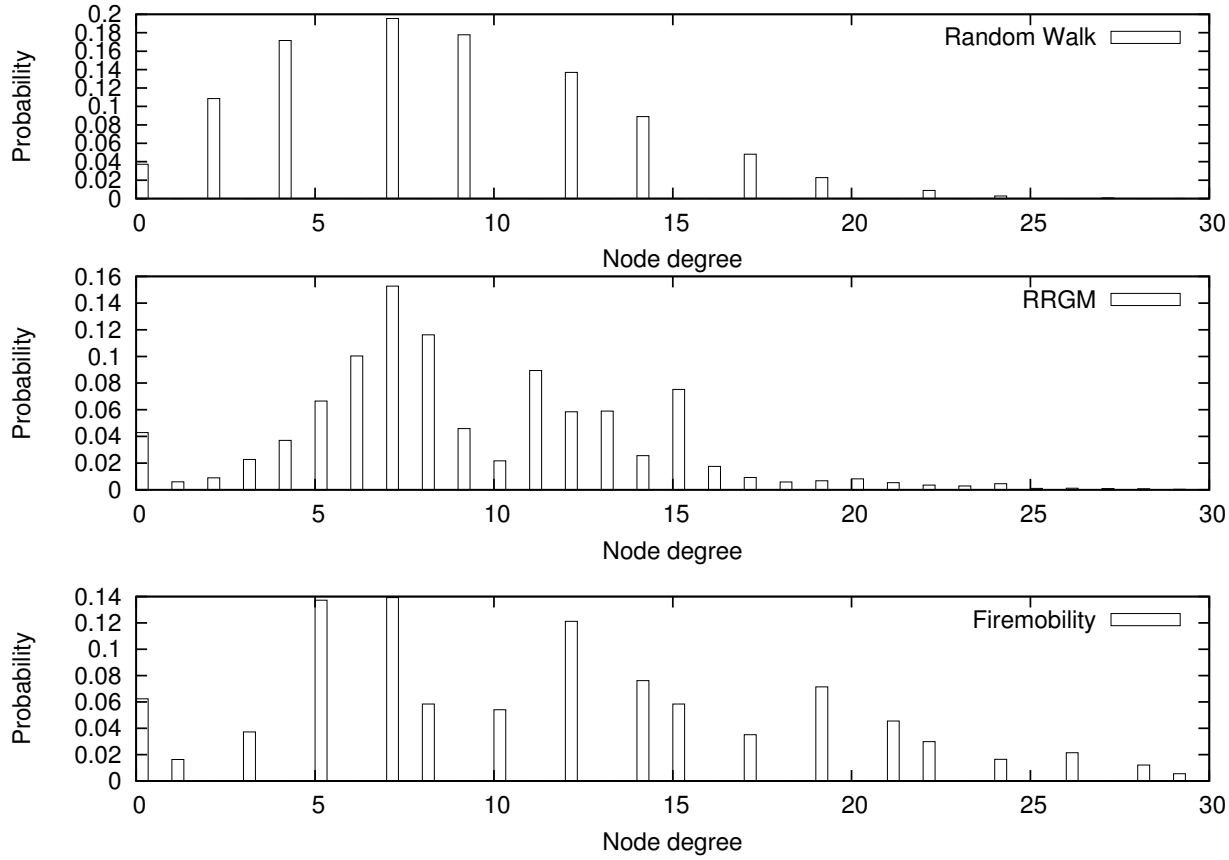


FIG. 4.15 – Distribution du degré des noeuds pour *Random Walk* (haut, $\bar{X} = 3.61$, $\sigma = 1.99$), RRGM (milieu, $\bar{X} = 9.16$, $\sigma = 4.87$) et *FireMobility* (bas, $\bar{X} = 6.61$, $\sigma = 4.01$). La portée radio est fixée à 70 m.

4.3.3 Importance du modèle dédié

Nous avons sélectionné deux modèles de mobilité pour les comparer à notre modèle dédié au scénario de lutte contre les feux de forêt (*FireMobility*). Le premier, *Random Walk* est un modèle d'entité qui partage avec *FireMobility* certaines propriétés temporelles, notamment la durée totale du partitionnement. Néanmoins, le caractère markovien du modèle, la répartition uniforme des noeuds sur la surface de simulation et la décorrélation des déplacements entraînent plus de différences notamment le degré des noeuds et la durée de vie des liens. Le deuxième modèle, RRGM est un modèle de groupe qui est adapté pour s'approcher de *FireMobility* avec l'introduction du feu et d'une zone d'intervention tout autour. Ces deux modèles sont similaires de point de vue fonctionnel avec la définition de

la notion de groupe, mais les résultats de simulation révèlent que leurs propriétés spatio-temporelles sont très différentes. Dans RRGM, le partitionnement est quasi-permanent quelque soit la valeur de la portée radio, alors qu'elle en dépend fortement dans *FireMobility*. Cette différence est expliquée par le regroupement des noeuds au sein d'un même groupe et la décorrélation des destinations des groupes. *FireMobility* définit des niveaux hiérarchiques et les déplacements des noeuds dépendent fortement de cette organisation.

Les propriétés spatio-temporelles étudiées interviennent dans le dimensionnement du système de télécommunications, notamment dans l'allocation de la bande passante du satellite et impactent aussi le comportement de l'algorithme de *clustering* qui sert à choisir les noeuds qui auront accès au satellite. La comparaison des trois modèles a révélé les similitudes et les différences et a surtout démontré que l'adaptation d'un modèle générique est toujours insuffisante. Il est donc indispensable de développer des modèles dédiés tels que *FireMobility* pour en dégager les propriétés du scénario considéré. Il est intéressant de noter que de toute manière, pour adapter le modèle générique au scénario étudié, nous avons besoin d'analyser en détails le fonctionnement des équipes sur le terrain ; ce qui revient en partie à développer un modèle dédié.
Pour conclure, nous choisissons le modèle de mobilité *FireMobility* pour représenter les déplacements des unités d'intervention dans le cadre d'un scénario de lutte contre les feux de forêt.

4.4 Simulation de KCMBC avec FireMobility

Dans cette section, le but est d'étudier le comportement de KCMBC, lorsque les noeuds suivent le modèle de mobilité *FireMobility*, choisi pour modéliser le scénario de lutte contre les feux de forêt. KCMBC est l'algorithme de *clustering* choisi pour résoudre le problème du choix des noeuds qui vont avoir accès au satellite.
Nous reprenons la définition des termes utilisés dans la technique de *clustering*. Le réseau est divisé en plusieurs groupes virtuels appelés *clusters* et à l'intérieur de chaque *cluster*, un noeud particulier appelé *clusterhead* coordonne les activités des autres membres. Dans le réseau, on distingue trois types de noeuds : les *clusterheads* qui sont les chefs de *clusters*, les membres qui sont les noeuds affiliés à un *clusterhead* et les orphelins (*orphans*) qui sont les noeuds qui n'ont pas encore choisi leur *clusterhead*.
Le *clustering* est composé de deux étapes : la formation des *clusters* et la maintenance de la structure du réseau. Pour la formation des *clusters*, nous avons besoin de paramétrer KCMBC et plus précisément fixer le valeur de k. Pour des raisons fonctionnelles qui seront reprises en détails dans le chapitre suivant, nous supposons que dans chaque partition, il y a un seul *clusterhead*. Pour satisfaire cette condition, nous nous basons sur l'étude

du diamètre des partitions (figure 4.16) pour fixer la valeur de k à 12. Le diamètre d'une partition présente le nombre de sauts maximal entre deux noeuds quelconques de cette partition.

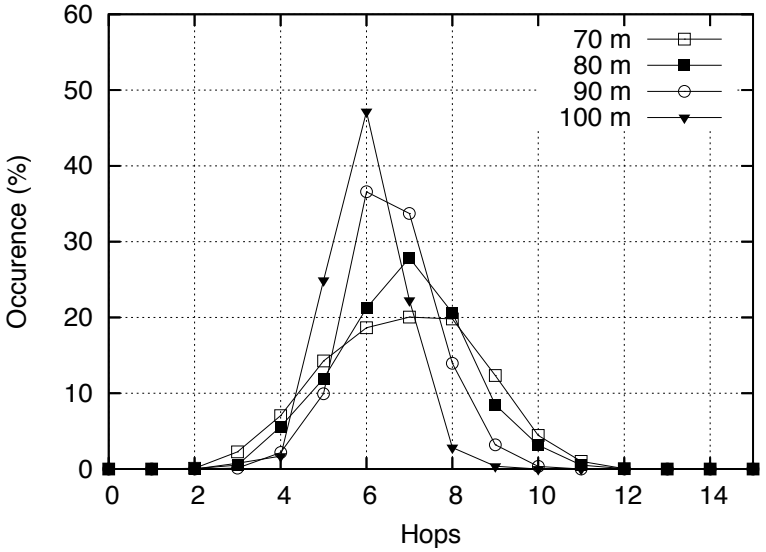


FIG. 4.16 – Distribution du diamètre des partitions pour les différentes valeurs de la portée radio.

Une fois les *clusters* formés, les changements topologiques peuvent altérer la structure des *clusters* et l'organisation du réseau. KCMBC considère quatre cas modélisant la dynamique réseau : un noeud quitte/rejoint le *cluster*, un lien apparaît/disparaît. Maintenant, la question qu'on peut se poser est : est-ce que la procédure de maintenance proposée par KCMBC et décrite dans le paragraphe 3.4.3 respecte bien la condition d'un seul *clusterhead* par partition ? Nous avons donc regardé le pourcentage du temps pendant lequel plusieurs *clusterheads* existent dans une même partition. La figure 4.17 montre que dans les meilleurs des cas, cette condition n'est pas respectée pendant à peu près 20% de la durée totale du partitionnement pour une portée radio égale à 100 m. La transgression de cette condition peut même durer pendant plus de 65% du temps pour une portée radio égale à 70 m.

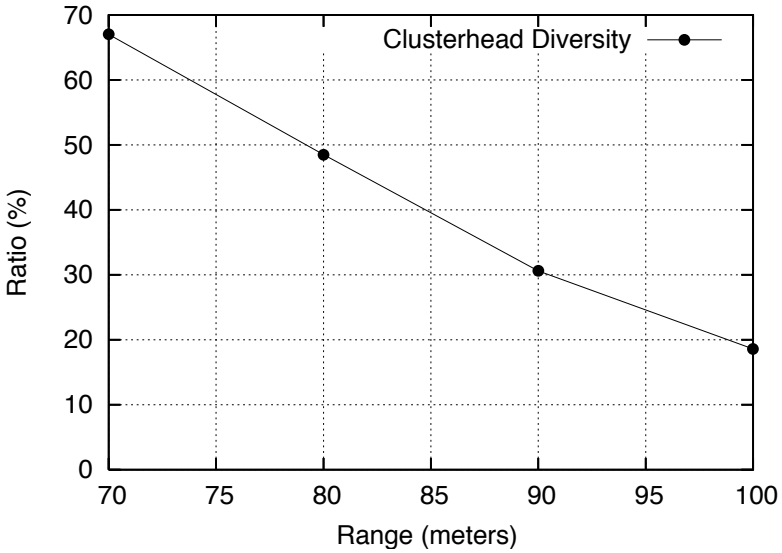


FIG. 4.17 – Durée pendant laquelle la condition d'un seul *clusterhead* par partition n'est pas respectée (pourcentage de la durée totale du partitionnement), en fonction de la portée radio.

KCMBC a proposé une classification des changements topologiques qui est plus adaptée à un réseau connexe qu'à un réseau partitionné. Il est donc indispensable de proposer une nouvelle classification qui prend en considération la dynamique des partitions. Pour comprendre cette dynamique, nous considérons dans un premier temps, l'évolution du nombre des partitions dans la réseau. On appelle la division de partition, la fragmentation d'une partition en plusieurs sous-partitions. La fusion de partitions définit l'union d'une ou de plusieurs partitions pour ne former qu'une seule. La figure 4.18 montre que le nombre de fusions et de divisions durant un évènement de partitionnement, diminue en fonction de la portée radio. Mais le plus intéressant c'est que le nombre de fusions et le nombre de divisions sont équivalents. Cela veut dire que durant un évènement de partitionnement, les partitions oscillent entre la division et la fusion. Pour la maintenance des *clusters*, ceci signifie qu'il faut considérer le cas où des *clusters* différents se rapprochent et aussi le cas les membres se trouvent déconnectés de leur *clusterhead*. Nous appelons ces deux cas l'agrégation des *clusters* et la perte du *clusterheads* et ils formeront la base pour une nouvelle procédure de maintenance décrite dans le chapitre suivant.

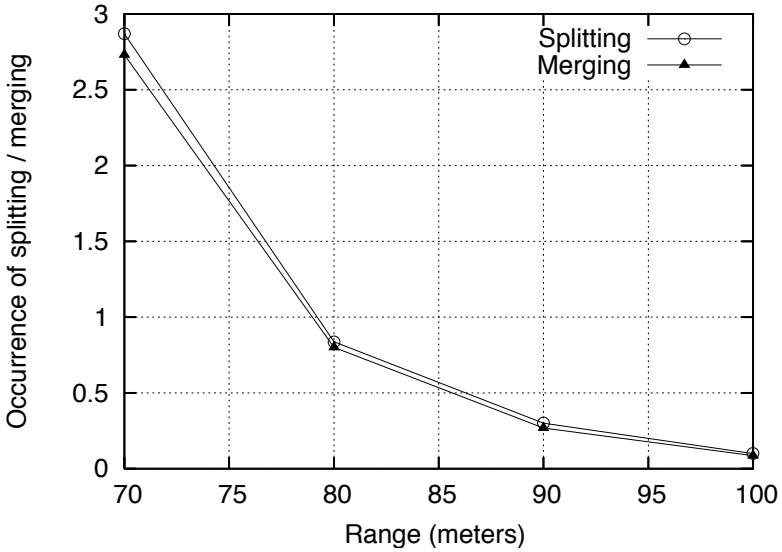


FIG. 4.18 – Moyenne du nombre de division (*splitting*) et de fusion (*merging*) de partitions durant un évènement de partitionnement en fonction de la portée radio.

La dynamique du réseau partitionné ne se limite pas à la dynamique des partitions. La structure du réseau peut changer sans que le nombre de partitions change. Les noeuds peuvent quitter leurs partitions pour aller rejoindre d'autres. On appelle ce phénomène la migration des noeuds. Nous nous intéressons particulièrement à la migration des membres. La figure 4.19 montre que la structure du réseau est de plus en plus stable en augmentant la valeur de la portée radio. Le plus important à retenir est que les membres peuvent perdre leur *clusterhead*, non pas parce que le *clusterhead* a quitté la partition mais parce qu'eux sont allés rejoindre un nouveau *cluster*.

La simulation de tout le système avec KCMBC comme algorithme de *clustering* et *FireMobility* comme modèle de mobilité permet premièrement de valider l'architecture présentée de manière théorique au début de cet exposé. Deuxièmement, elle nous a permis de comprendre la dynamique du réseau et aidé pour proposer une nouvelle procédure de maintenance adaptée à un réseau partitionné.

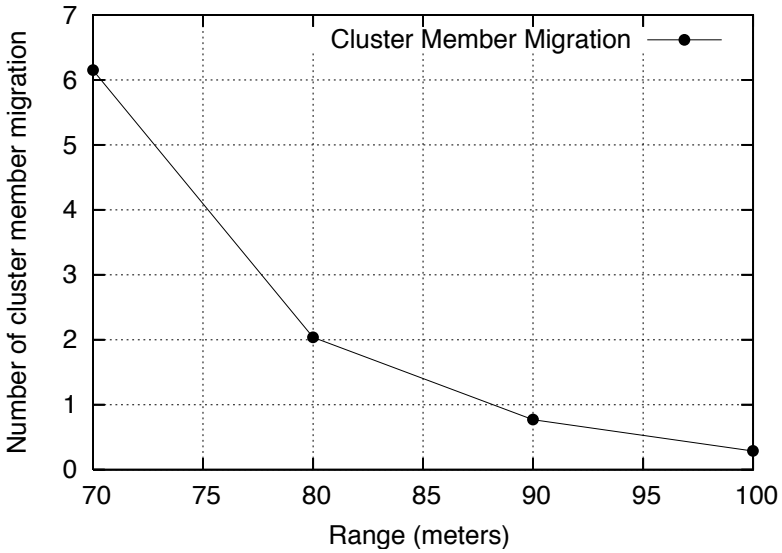


FIG. 4.19 – Moyenne du nombre de migration des membres durant un évènement de partitionnement en fonction de la portée radio.

4.5 Conclusion intermédiaire des chapitres 3 et 4

Cette conclusion couvrira aussi les chapitres précédents pour expliquer toute la démarche suivie. Le but de cette thèse est la proposition, l'étude et la validation d'une architecture permettant la restauration de la connexité dans un réseau déployé lors des opérations feux de forêt. Nous avons proposé d'utiliser le satellite comme solution pour le rétablissement du service de communications dans le réseau et l'interconnexion des partitions. Parmi les problèmes soulevés par cette solution, nous nous intéressons au choix des noeuds qui vont avoir accès au satellite. Dans le chapitre 3 nous avons étudié les approches qui permettent de structurer un réseau ad hoc mobile. Nous avons comparé les approches basées sur l'ensemble dominant connecté à celles basées sur l'ensemble dominant indépendant, communément connues sous le nom de *clustering*. Étant donné la dynamique du réseau, nous avons conclu que ces dernières sont plus adaptées à notre problème. Parmi les algorithmes proposés dans la littérature, nous avons choisi l'algorithme de *clustering* à multi-sauts KCMBC basé sur l'approche Max-Min.

L'évaluation de KCMBC dans un environnement mobile a posé le problème de la modélisation mathématique des déplacements des noeuds sur la surface de simulation. Cette problématique a fait l'objet d'une étude exposée dans ce dernier chapitre. Un modèle de mobilité peut être soit générique en s'appliquant à un large spectre de scénarios, soit dédié à un scénario

bien particulier. Dans les modèles génériques, on distingue deux catégories : les modèles de mobilité d'entité et les modèles de mobilité de groupe. Dans les modèles d'entité, la position de chaque noeud est définie indépendamment des autres, tandis que dans les modèles groupe les noeuds entretiennent entre eux certaines relations de dépendance spatiale. Dans la première catégorie, on trouve les modèles *Random WayPoint*, *Random Walk* et *Boundless Simulation Area* et dans la deuxième les modèles de Sanchez et *Reference Region Mobility Model*.

Avec les modèles de mobilité de groupe, on commence à avoir des modèles qui font apparaître des propriétés particulières et imaginer des scénarios d'applications spécifiques. Le *Reference Region Mobility Model* par exemple tente de reproduire le mode de fonctionnel des équipes de secours (pompiers, médecins, etc.). Mais il reste tout de même un modèle générique. Avec les modèles dédiés, on part du scénario et on essaye de définir l'ensemble de ses propriétés mathématiques. Les modèles avec des restrictions géographiques traduisent les contraintes de l'environnement par des conditions sur les trajectoires, en introduisant des obstacles (modèles de Johansson) ou en traçant les chemins que les noeuds peuvent emprunter (*Pathway Mobility Model*). Cependant, ces modèles gardent encore une grande part d'aléatoire pour les appliquer. Certains travaux proposent des modèles dédiés à l'urgence médicale et aux premiers secours, mais ils se sont essentiellement focalisés sur la partage de l'espace des opérations et l'attribution des rôles. Il leur manque le concept de coordination. Or quand on parle de la mise en place de communications d'urgence, on parle de coordination des efforts et de coopération des équipes. Nous avons donc utilisé un modèle de mobilité, *FireMobility* qui définit des niveaux hiérarchiques, où les déplacements d'un noeud peuvent affecter la mobilité des noeuds appartenant à des niveaux inférieurs.

Dans ce chapitre nous avons comparé *FireMobility* à un modèle d'entité (*Random Walk*) et un modèle de groupe *Reference Region Mobility Model* (RRGM) et à travers une caractérisation spatio-temporelle, nous avons démontré les différences fondamentales dans le mode fonctionnel de chaque modèle. Les propriétés étudiées (durée du partitionnement, degré du noeud et la durée de vie des liens) interviennent dans le dimensionnement du système de télécommunications, notamment dans l'allocation de la bande passante du satellite et impactent aussi le comportement de l'algorithme de *clustering* qui sert à choisir les noeuds qui auront accès au satellite. La comparaison des trois modèles a révélé les similitudes et les différences et a surtout démontré que l'adaptation d'un modèle générique est toujours insuffisante. Il est indispensable de développer des modèles dédiés pour en dégager les propriétés du scénario considéré.

Le dernier point abordé dans ce chapitre était la validation de toute l'architecture formée

par KCMBC et *FireMobility*. La simulation du système présenté de manière théorique au début nous a permis de valider cette architecture et de comprendre la dynamique du réseau. L'analyse de tout le système est le point de départ pour proposer une nouvelle procédure de maintenance adaptée à un réseau partitionné. Cette nouvelle procédure sera exposée dans le chapitre suivant.

Chapitre 5

Maintenance des clusters

Sommaire

Le processus du *clustering* est composé de deux phases : la formation des *clusters* et la maintenance de la structure du réseau. La première phase est accomplie en choisissant l'ensemble des noeuds qui vont jouer le rôle de *clusterheads*. La deuxième phase est nécessaire pour maintenir l'organisation du réseau en présence de la mobilité des noeuds. Nous avons vu dans le chapitre 3 que les algorithmes de *clustering* à multi-sauts se sont essentiellement focalisés sur la définition des règles qui préviennent la rupture de la structure des *clusters*. Ces règles gèrent le comportement des noeuds dans quatre cas modélisant la dynamique du réseau : un noeud rejoint le *cluster*, un noeud quitte le *cluster*, un lien

apparaît et un lien disparaît. Mais comme l'ont indiqué les auteurs de k-lowestID [58], définir les règles n'est qu'une étape. Il est indispensable de déployer des techniques plus spécialisées et plus performantes en termes de qualité des *clusters* et surtout en termes de surcoût en signalisation.

Dans le cadre de cette thèse, nous nous intéressons aux approches permettant de réduire le surcoût en signalisation engendré par la maintenance des *clusters*. En effet, [51] rapporte que la signalisation requise pour maintenir une bonne qualité de *clustering* est la principale source de critiques adressées au *clustering* à multi-sauts.

5.1 Stabilité des clusters

A cause de la mobilité des noeuds, il peut y avoir de fréquents évènements déclenchant la réorganisation des *clusters*. Ces évènements incluent la transition des membres d'un *cluster* à un autre (réaffiliation), la démission des *clusterheads* et le *reclustering*. Le *reclustering* a lieu quand les noeuds reprennent le processus du *clustering* dès la phase de formation des *clusters*. Dans le cas de KCMBC par exemple, le *reclustering* consiste à exécuter toutes les étapes à partir de FloodMax jusqu'à l'affiliation des membres à leur *clusterhead* (paragraphe 3.3.3.4). Dans certains cas, un simple changement peut déclencher une réaction en chaîne de changements de rôle. De tels phénomènes dégradent considérablement les performances du réseau car une bonne partie de l'énergie des noeuds est consommée pour le traitement des messages de contrôle générés.

Les algorithmes à faible coût de maintenance proposent comme solutions la relaxation des hypothèses sur le *clustering* [51] et/ou la formation de *clusters* stables dès la phase initiale du *clustering* [98]. La notion de stabilité des *clusters* est définie par le nombre d'évènements entraînant la réorganisation des *clusters* et les changements de rôle des noeuds.

5.1.1 Relaxation des hypothèses de clustering

LCC [53] est l'un des premiers algorithmes à s'intéresser à la stabilité des *clusters*. Le *clusterhead* n'est pas tenu à avoir le meilleur poids dans son voisinage. Il peut garder son statut de *clusterhead* tant que ses voisins possédant de meilleurs poids que le sien sont tous des membres. C'est uniquement quand deux *clusterheads* deviennent voisins ou un noeud n'arrive pas à joindre un *cluster* que LCC exige l'initiation du processus du *reclustering* dans tout le réseau.

3hBAC [54] propose de réduire l'effet du *reclustering* grâce au concept du membre invité. Un noeud dont tous les voisins directs sont des membres peut s'affilier à un *cluster* comme membre invité. La distance entre lui et son *clusterhead* est alors égale à deux sauts. Le concept du *reclustering* local est une deuxième mesure pour réduire le nombre de messages de contrôle. Quand deux *clusterheads* deviennent voisins, un *reclustering* global n'est pas nécessaire. Les deux *clusters* concernés par ce changement initient le processus du *reclustering*.

Afin de limiter les réaffiliations des noeuds et leur transition d'un *cluster* à un autre, CSC [70], un algorithme de *clustering* à k sauts, propose de relaxer la condition sur le nombre maximal de sauts entre un membre et son *clusterhead*. Le noeud peut donc se retrouver à k' sauts de son *clusterhead*, avec $k' > k$. La seule condition imposée est que les *clusters* ne doivent pas se chevaucher.

Toutes ces approches réduisent les évènements conduisant à la réorganisation des *clusters*, lors de la phase de maintenance. Une deuxième catégorie d'approches s'attaque au problème dès la phase de formation des *clusters*.

5.1.2 Formation de clusters stables

La stabilité des *clusters* peut être améliorée grâce à la définition d'un ensemble de critères qui allongent la durée de vie des *clusterheads*. Plus la durée de vie des *clusterheads* est longue, moins de messages de signalisation sont échangés dans le réseau. La disparition d'un *clusterhead* à cause de l'épuisement de ses batteries par exemple, nécessite la réélection d'un nouveau *clusterhead* et par conséquent l'échange de messages de signalisation supplémentaires. Une première approche consiste alors à éviter l'élection des noeuds à faible autonomie de batteries [59] [67].

Étant donné que le *clusterhead* possède des fonctionnalités supplémentaires par rapport aux membres, il risque d'épuiser rapidement ses batteries. La répartition de charge est la distribution équitable des activités de traitement et de communication entre les différents *clusterheads*. La charge du *clusterhead* dépend fortement de la taille de son *cluster*. Créer des *clusters* de tailles équivalentes permet de bien répartir la charge globale du réseau et éviter la favorisation d'un *clusterhead* au détriment d'un autre [99].

On ne peut parler de stabilité sans évoquer la mobilité. La stabilité d'un noeud n'est pas mesurée uniquement grâce à sa vitesse mais grâce aux caractéristiques globales de son schéma de mobilité. Le noeud stable est le noeud qui est capable de garder ses voisins le plus longtemps possible ; d'où la définition de la mobilité relative. Grâce à la mesure

de la mobilité relative, on s'assure que le noeud choisi comme *clusterhead* est le noeud qui maintient la même organisation du *cluster*, pour une durée plus longue que les autres noeuds [98] [65]. Un aperçu des méthodes de mesure de la mobilité relative est donné dans le paragraphe 3.3.1.3.

Une première manière d'aborder le problème du surcoût en signalisation de la maintenance est de réduire les évènements conduisant à la réorganisation des *clusters*. Cette approche est plutôt préventive. La relaxation des conditions sur le *clustering* et la création des *clusters* stables limitent l'appel à la procédure de maintenance et évitent au maximum l'utilisation des messages de contrôle. Mais la question qui se pose est quels sont les messages de contrôle utilisés.

5.2 Schémas de maintenance

La plupart des algorithmes de *clustering* se sont focalisés sur la description du comportement les noeuds face à un changement topologique donné. Ils précisent par exemple comment le noeud doit réagir quand il entre dans le réseau ou quand il perd son *clusterhead*. Ils spécifient aussi dans quels cas un processus de *reclustering* est déclenché. Mais pour prendre une décision, chaque noeud doit disposer d'un certain nombre d'informations qui lui permettent d'appliquer correctement les règles de maintenance. Si on suppose que le noeud doit connaître uniquement l'état de ses voisins directs, un échange périodique d'informations à un seul saut est suffisant. Mais si on suppose que le noeud doit connaître tous les membres de son *cluster*, une autre procédure d'échange d'informations est nécessaire : toute disparition ou apparition de noeud doit être signalée dans le *cluster* [100]. Ces deux schémas sont très différents et engendrent des surcoûts en signalisation très différents aussi.

La définition du comportement des noeuds lors de la phase de maintenance est une première étape. L'étape qui suit est de préciser quels messages utiliser et à quelle fréquence. Nous allons voir dans la suite les quelques efforts qui ont été faits dans cette direction.

5.2.1 Clustering à un seul saut

FWCA (*Flexible Weight Based Clustering Algorithm*) [66] s'intéresse particulièrement à la maintenance des *clusters*. Il a l'avantage non seulement de préciser les messages utilisés mais aussi de définir leurs structures. L'état de chaque noeud est défini par un 5-tuple :

(id_{node}, id_{CH}, *weight*, *counter*, n_{max}), où id_{node} est l'identifiant du noeud, id_{CH} l'identifiant de son *clusterhead*, *weight* son poids, *counter* la taille actuelle du *cluster* et n_{max} la taille maximale du *cluster* (le seuil). Grâce à un échange périodique de messages HELLO, chaque noeud sauvegarde l'état de tous ses voisins directs.

La plupart des algorithmes décrivent le comportement des noeuds en énumérant les quatre principaux cas qui modélisent les changements topologiques : un noeud arrive/disparaît et un lien apparaît/disparaît. FWCA procède différemment et décrit le comportement des noeuds suivant leur état : nouveau, *clusterhead* ou membre.

5.2.1.1 Comportement d'un nouveau noeud

Quand un noeud entre dans le réseau, il diffuse un message JOIN_REQUEST et attend la réponse des *clusterheads* existants. Un *clusterhead* peut accepter la requête par un message WELCOME_ACK ou la décliner par un message WELCOME_NACK. Si le noeud reçoit plusieurs réponses positives, il choisit le *clusterhead* possédant le plus petit poids et lui envoie un JOIN_ACCEPT. L'affiliation du noeud n'est définitive que lorsqu'il reçoit un message CH_ACK de la part du *clusterhead*.

5.2.1.2 Comportement d'un clusterhead

Le comportement du *clusterhead* est dicté par l'arrivée et le départ des ses membres et aussi par la qualité de ses voisins. Le *clusterhead* maintient deux tables, une pour les membres (TB_M) et une pour les *clusterheads* voisins (TB_{CH}). La réponse du *clusterhead* à un message JOIN_REQUEST dépend de la taille actuelle du *cluster*, de ses ressources disponibles et de la durée de vie du lien.
A l'instar de l'admission d'un nouveau noeud, le *clusterhead* est informé du départ d'un ses membres grâce à un message LEAVE. A la réception de ce message, le *clusterhead* met à jour la table TB_M et en informe ses membres pour décrémenter de un la variable *counter*.
Si l'un de ses voisins possède un poids plus petit que le sien et que ce voisin est stable, il lui envoie toutes ses bases de données (DATABASE_INFO). A la réception d'un acquittement (DATABASE_ACK), il met à jour son état, informe les membres du *cluster* qu'un nouveau *clusterhead* est choisi (CH_CHANGE) et envoie un CH_INFO à ce dernier pour confirmer le changement de rôle.

5.2.1.3 Comportement d'un membre

Un membre du *cluster* réagit à un message DATABASE_INFO par l'envoi d'un message DATABASE_ACK. Il attend la réception d'un message CH_INFO pour devenir un *cluste-*

rhead. L'algorithme permet à un membre de quitter son *cluster* et aller rejoindre un autre si le nouveau *clusterhead* est plus performant. Il informe son ancien *clusterhead* de son départ et procède de la même manière qu'un nouveau noeud pour son admission dans le nouveau *cluster*.

5.2.1.4 Structure des messages échangés

Dans ce paragraphe, nous détaillons la structure des messages envoyés lors de la phase de maintenance, ainsi que leur fonction :

* HELLO (id_{node}, id_{CH}, $weight$, $counter$, n_{max}) : la mise à jour des tables.
* Join_REQUEST (id_{node}, id_{CH}) : une demande d'affiliation à un *clusterhead*.
* WELCOME_ACK (id_{node}, id_{CH}, $weight$) : une réponse positive à une demande d'affiliation.
* WELCOME_NACK (id_{node}, id_{CH}, $weight$) : une réponse négative à une demande d'affiliation.
* JOIN_ACCEPT (id_{node}, id_{CH}, $weight$, $counter$, n_{max}) : la confirmation par un membre d'un WELCOME_ACK.
* CH_ACK (id_{node}, id_{CH}, $weight$, $counter$, n_{max}) : le *clusterhead* ajoute le noeud à son *cluster*.
* DATABASE_INFO () : le *clusterhead* actuel envoie sa base de données au nouveau *clusterhead*.
* Database_ ACK (id_{node}, id_{CH}, $weight$, $counter$, n_{max}) : le nouveau *clusterhead* envoie l'accusé de réception de la base de données.
* CH_CHANGE (id_{CH}) : le *clusterhead* informe les membres du *cluster* de la présence d'un nouveau *clusterhead*.
* CH_INFO (id_{node}, id_{CH}, $weight$, $counter$, n_{max}) : la confirmation du changement de rôle (de l'ancien *clusterhead* au nouveau *clusterhead*).
* LEAVE (id_{node},id_{CH}) : le noeud quitte le *cluster*.

La valeur ajoutée de FWCA est la description détaillée de la structure des messages envoyés. Mais son inconvénient majeur est le mode de fonctionnement de la procédure, centré sur le *clusterhead*. Ce dernier est consulté dans toutes les décisions telles que l'affiliation ou le départ d'un membre et l'élection d'un nouveau *clusterhead*. Cette approche peut créer des goulots d'étranglement et épuiser les ressources des *clusterheads*, notamment dans les réseaux à très forte mobilité. En dépit de cet inconvénient, la spécification des messages de signalisation permet de caractériser quantitativement, le surcoût engendré

lors de la phase de maintenance.

Dans le paragraphe suivant, nous allons prendre l'exemple de l'algorithme de *clustering* à multi-sauts KCMBC. Nous allons voir, à travers la description de la procédure de maintenance que la caractérisation du coût de la signalisation ne peut être faite que qualitativement.

5.2.2 Clustering à multi-sauts

KCMBC[63] est l'algorithme de *clustering* qui a été choisi pour résoudre le problème de choix des noeuds qui vont assurer l'interconnexion entre les différentes partitions du réseau. De la même manière, KCMBC considère quatre cas modélisant les changements topologiques dans le réseau : activation d'un noeud ou d'un lien et disparition d'un noeud ou d'un lien.

5.2.2.1 Activation d'un noeud ou d'un lien

Chaque noeud envoie périodiquement un message HELLO à ses voisins à un seul saut. Quand un noeud orphelin i reçoit des messages HELLO, il essaye de rejoindre un *clusterhead* situé à moins de k sauts. Si plusieurs *clusterheads* sont candidats, il s'affilie au *cluster* du voisin possédant le plus grand temps d'expiration. Pour la décision d'affiliation, chaque noeud i doit donc connaître l'identifiant de chacun de ses voisins j situés à un seul saut, l'identifiant du *clusterhead* de j, le temps d'expiration de j et le nombre de sauts entre j et son *clusterhead* (celui de j). Si le noeud orphelin détecte plus de D_r voisins orphelins, il initie le processus de *reclustering* pour former un nouveau *cluster*.

La création d'un nouveau lien entre deux noeuds i et j peut changer la structure du *cluster*. Si i et j sont des *clusterheads*, les deux *clusters* fusionnent et le *clusterhead* i, à plus petit temps d'expiration rejoint avec ses membres le nouveau *cluster* j. Si les deux noeuds appartiennent à deux *clusters* différents, aucune réaffiliation n'est initiée s'ils sont à moins de k sauts de leurs *clusterheads*. Pour prendre une telle décision, le noeud doit donc connaître le nombre de sauts qui le séparent de son *clusterhead*.

5.2.2.2 Disparition d'un noeud ou d'un lien

La structure du réseau est dynamique avec le départ des noeuds et la disparition des liens. Ces changements peuvent entraîner la perte du chemin vers le *clusterhead* ou son allongement à plus de k sauts. Dans ce cas, le noeud devient orphelin et suit la procédure d'activation d'un noeud décrite dans le paragraphe précédent. Suite à ces évènements, la structure du *cluster* peut devenir de mauvaise qualité : tous les membres sont à un

seul saut de leur *clusterhead*. Dans ce cas, le *cluster* fusionne avec les *clusters* voisins. En outre, quand le *clusterhead* prédit la disparition de son dernier voisin, il choisit un nouveau noeud pour prendre le relais. Les noeuds qui ne sont pas dans le voisinage à k sauts du nouveau *clusterhead* essayent de rejoindre d'autres *clusters*.

5.2.3 Synthèse sur les schémas de maintenance

Dans le cas où les membres sont à un seul saut de leur *clusterhead*, la maintenance est plus facile à mettre en oeuvre. Les auteurs de FWCA proposent un schéma de maintenance où ils décrivent le comportement de chaque noeud suivant son rôle dans le *cluster*. Certes, leur procédure peut créer des goulots d'étranglement au niveau des *clusterheads*, particulièrement dans des réseaux à forte mobilité, mais elle a l'avantage de décrire la structure des messages échangés. Cette description permet de caractériser quantitativement l'effet de la signalisation sur les performances du réseau. KCMBC, un algorithme de *clustering* à multi-sauts, se contente de décrire les règles qui régissent le réseau sans préciser comment les mettre en pratique. Ce manque de précision sur la structure des messages et leur fréquence ne permet qu'une analyse qualitative de la procédure de maintenance proposée.

Nous nous intéressons dans la suite, à la conception d'une procédure de maintenance pour le *clustering* à multi-sauts. La maintenance dépend fortement du contexte et des caractéristiques du réseau. Nous allons donc commencer par détailler un cahier de charges qui prend en considération le contexte des communications d'urgence et le modèle de mobilité *FireMobility*.

5.3 Maintenance dans un réseau partitionné

Nous définissons tout d'abord les besoins de notre application en termes d'informations requises pour réagir aux changements topologiques. Nous comparons ensuite deux différentes approches de maintenance : *Periodical Broadcast* et *Passive Maintenance*.

5.3.1 Cahier des charges

Le cahier des charges décrit dans ce paragraphe prend en considération le contexte général qui est communications d'urgence et aussi le schéma de mobilité des noeuds qui est *FireMobility*.

5.3.1.1 Un seul clusterhead par partition

On suppose que tous les noeuds ont la capacité d'utiliser le segment spatial. Mais afin d'optimiser l'utilisation des ressources du réseau terrestre et du satellite, on suppose qu'un seul noeud par partition accède au satellite et assure l'interconnexion avec les autres partitions. [101] rapporte qu'en règle générale les réseaux d'urgence déploient une seule passerelle, afin d'optimiser les coûts. Avec un seul *clusterhead* par partition, la question qui se pose est est-ce que ce dernier pourra supporter les fonctionnalités supplémentaires qui lui sont attribuées. La réponse dépend du nombre de noeuds présents dans chaque partition. En effet, la charge du *clusterhead* dépend fortement de la taille de son *cluster*. Dans *FireMobility*, le nombre total de noeuds est égal à 37. La figure 5.1 montre que la taille des partitions est inférieure ou égale à 25 noeuds dans 74% du temps pour une portée radio égale à 100 m et dans 93% du temps pour une portée radio égale à 70 m.

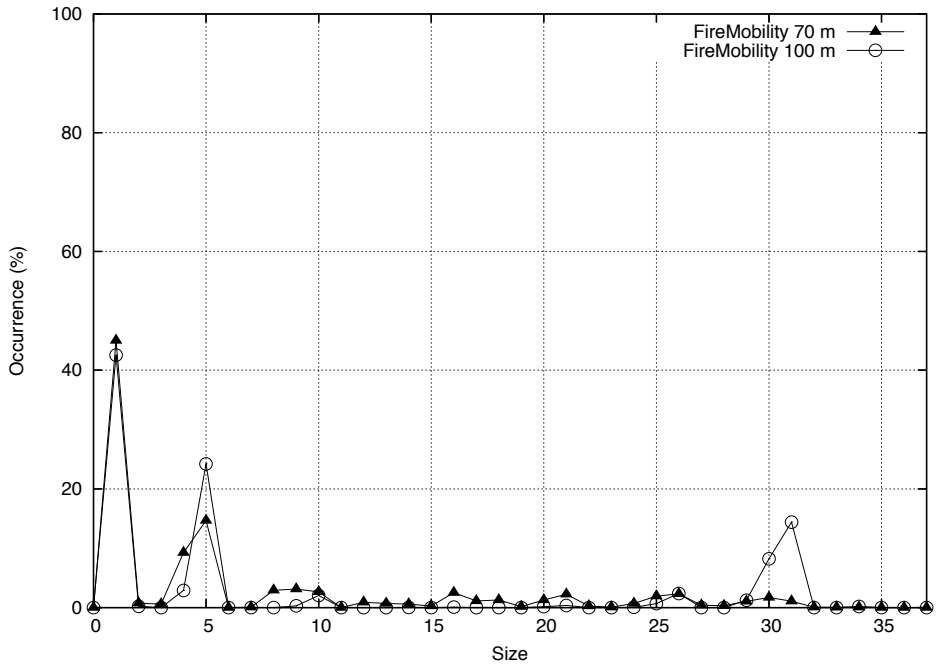


FIG. 5.1 – Distribution de la taille des partitions pour une portée radio égale à 70 m et 100 m (modèle de mobilité : *FireMobility*).

5.3.1.2 Gestion de l'agrégation de clusters

A cause des changements topologiques, un *clusterhead* peut quitter sa partition et rejoindre une autre où il existe déjà un *clusterhead*. Afin de respecter la première condition

(un *clusterhead* par partition), le *clusterhead* i possédant le temps d'expiration le plus petit doit démissionner et rejoindre le *cluster* représenté par le *clusterhead* j. Les membres du *cluster* i procèdent de la manière et rejoignent le *cluster* j. Pour prendre une telle décision, un noeud doit connaître non seulement les identifiants des *clusterheads* présents dans le voisinage à k saut, mais aussi leurs temps d'expiration pour pouvoir les comparer.

5.3.1.3 Gestion de la perte du clusterhead

Afin de déclencher l'évènement de *reclustering*, un noeud orphelin doit détecter plus de D_r autres noeuds orphelins dans son voisinage. Donc chaque noeud doit connaître les identifiants des *clusterheads* de ses voisins directs. En outre, afin de calculer le temps d'expiration, chaque noeud diffuse périodiquement sa position (x,y) à ses voisins. Ces deux informations (la position et l'identifiant du *clusterhead*) sont incluses dans les messages HELLO envoyés périodiquement toutes les *Hello_Period* secondes.

Mais avant de devenir orphelin, le noeud a besoin de détecter la perte de son *clusterhead*. Ce dernier doit donc périodiquement annoncer sa présence à ses membres. Au bout d'un certain temps (T_{out}), si le noeud ne reçoit aucune annonce, il considère que son *clusterhead* pourrait être parti. La manière dont le *clusterhead* annonce sa présence dans le *cluster* est le sujet de discussion des paragraphes suivants.

5.3.2 Periodical Broadcast

Bellavista et Magistretti proposent dans [102] d'étudier les différents aspects de la maintenance, indépendamment de la formation des *clusters*. Ils s'intéressent particulièrement à l'effet de la mobilité sur la qualité du *clustering* à multi-sauts. Pour leur étude, ils proposent d'utiliser un protocole très simple : chaque *clusterhead* diffuse dans son voisinage à k sauts un message REFRESH toutes les *Refresh_Period* secondes. Ce protocole sera appelé *Periodical Broadcast* pour mettre en évidence le caractère périodique de la diffusion de l'état du *clusterhead*.

Dans la version originelle de *Periodical Broadcast*, le choix des *clusterheads* est basé sur les identifiants des noeuds. Le noeud possédant le plus petit identifiant est choisi comme *clusterhead*. Nous le modifions pour prendre en considération le critère de choix implémenté dans KCMBC. Le *clusterhead* inclut donc dans ses messages REFRESH, la valeur de son temps d'expiration en plus de son identifiant. Chaque membre maintient une table où il sauvegarde l'identifiant et le temps d'expiration de chaque *clusterhead* situé dans son voisinage à k sauts. La table est mise à jour à la réception d'un message REFRESH de la part d'un *clusterhead* et un temporisateur est associé à chaque entrée, afin de supprimer

les données obsolètes.

Les changements topologiques qui peuvent se produire dans un réseau partitionné sont traités de la manière suivante :

1. Un membre ne reçoit pas de message REFRESH de la part de son *clusterhead* : si toutes les entrées de la table de *clusterheads* ont expiré, il devient orphelin et vérifie s'il existe D_r autres noeuds orphelins dans son voisinage. Si c'est le cas, il initie le processus de *reclustering*.

2. Un membre déjà affilié à *clusterhead* A, reçoit un message REFRESH de la part d'un autre *clusterhead* B : si le temps d'expiration de B est supérieur à celui de A, le noeud quitte le *clusterhead* A et rejoint le *clusterhead* B.

3. Un *clusterhead* A reçoit un message REFRESH de la part d'un autre *clusterhead* B : si le temps d'expiration de B est supérieur au sien, il change de rôle et devient un membre. Grâce à la règle précédente, tous les membres de son (ancien) *cluster* procèdent de la même manière et s'affilient à B.

Ces trois règles permettent de garantir un seul *clusterhead* par partition et gérer la perte du *clusterhead* et l'agrégation des *clusters*. Le protocole *Periodical Broadcast* répond donc au cahier des charges relatif à la maintenance dans un réseau déployé dans des situations d'urgence et régi par le modèle de mobilité *FireMobility*. Il utilise deux types de messages :

1. HELLO (TP, id_i, id_{ch}, x, y) : TP est le type du message, id_i l'identifiant du noeud i, id_{ch} l'identifiant de son *clusterhead*, x et y sa position. Ce message est envoyé périodiquement par i à ses voisins directs.

2. REFRESH (TP, id_i, T_i) : TP est le type du message, id l'identifiant du *clusterhead* i source du message et T_i son temps d'expiration. Il est envoyé périodiquement par chaque *clusterhead*.

Le caractère périodique dans l'envoi des messages REFRESH peut considérablement augmenter le surcoût en signalisation et dégrader les performances du réseau. Le protocole utilise déjà des messages HELLO qui contiennent la position du noeud et l'identifiant du *clusterhead*. Ces messages incluent donc une information décrivant l'état du *cluster* qui est l'identifiant du *clusterhead*. Nous partons de cette constatation pour proposer une nouvelle procédure de maintenance appelée *Passive Maintenance*.

5.3.3 Passive Maintenance

Nous allons exploiter le fait que les noeuds communiquent l'identifiant de leurs *clusterheads* à leurs voisins pour proposer un nouveau protocole de maintenance. La maintenance et la découverte du voisinage deviennent alors corrélées. L'information incluse dans les messages HELLO, peut devenir obsolète au bout d'un certain temps à cause de la dynamique du réseau. On suppose qu'un groupe de noeuds se retrouve déconnecté de son *clusterhead*. En communiquant uniquement l'identifiant de leur *clusterhead*, ces noeuds ne pourront jamais se rendre compte que ce dernier a déjà quitté la partition. La solution consiste à ajouter un champ supplémentaire caractérisant la validité de l'information incluse dans les messages HELLO. La technique d'horodatage (*time stamping*) permet de préciser à quel instant le dernier message HELLO a été envoyé par le *clusterhead*. L'utilisation de la technique de l'horodatage suppose que le réseau est synchrone. Il est important de préciser que le *clusterhead* envoie un message HELLO à ses voisins directs uniquement et non pas à ses voisins à k sauts. Ainsi de proche en proche, l'information sur l'état du *clusterhead* est propagée dans le *cluster*. Le champ supplémentaire (*ch_last_sent_hello_msg*) ne peut à lui seul garantir l'unicité du *clusterhead* par partition. Nous proposons d'utiliser des messages dédiés et envoyés uniquement en cas de besoin : CH_REQ et CH_REPLY.

Dans la suite, nous décrivons comment ces messages seront utilisés pour répondre au cahier des charges. Nous classifions les changements topologiques en deux catégories : l'agrégation des *clusters* et la perte du *clusterhead*.

5.3.3.1 Agrégation des clusters

L'agrégation des *clusters* a lieu quand deux groupes de noeuds [1] appartenant initialement à deux partitions différentes donc à deux *clusters* différents, fusionnent pour former une seule partition. Trois cas de figures peuvent de présenter : dans la nouvelle partition, il peut y avoir un seul, deux ou aucun *clusterhead*. Mais les noeuds n'ont aucun moyen de le savoir. Ils disposent néanmoins d'une information importante qui leur permettra de détecter l'agrégation de *clusters* de manière locale. Chaque noeud, grâce aux messages HELLO, connaît ses voisins et les identifiants de leurs *clusterheads*. Un noeud appartenant à un *cluster* A et possédant des voisins appartenant à un *cluster* B détecte le rapprochement de deux *clusters*. Mais dans le cas général, il est incapable de connaître le nombre de *clusterheads* présents dans la partition. Il a donc besoin de les solliciter à annoncer leur présence. On définit un nouveau paramètre d, similaire au paramètre D_r défini dans

[1] Nous prenons l'exemple de deux groupes pour faciliter la description du mécanisme, mais ils peuvent être plusieurs groupes qui se rapprochent.

KCMBC. Si un noeud détecte plus de d voisins appartenant à un *cluster* différent, il diffuse dans le réseau un message CH_REQ, à la recherche de *clusterheads* potentiels. Si un *clusterhead* reçoit ce message, il répond par un message CH_REPLY en indiquant son temps d'expiration. Les membres se contentent de relayer le message. A la réception d'un ou plusieurs messages CH_REPLY, chaque noeud, y compris les *clusterheads*, rejoint le *clusterhead* possédant le plus grand temps d'expiration et met à jour son temporisateur (*ch_last_sent_hello_msg*).

Comme nous l'avons indiqué au début de ce paragraphe, les deux groupes qui se sont rapprochés peuvent être formés uniquement de membres et aucun *clusterhead* n'est présent dans la partition. Un temporisateur est associé à l'envoi de la requête CH_REQ et à l'expiration de ce temporisateur, si aucune réponse CH_REPLY n'est reçue, le processus de *reclustering* est déclenché et un nouveau *clusterhead* est sélectionné. La figure 5.2 montre l'ordinogramme du protocole pour la gestion de l'agrégation des *clusters*.

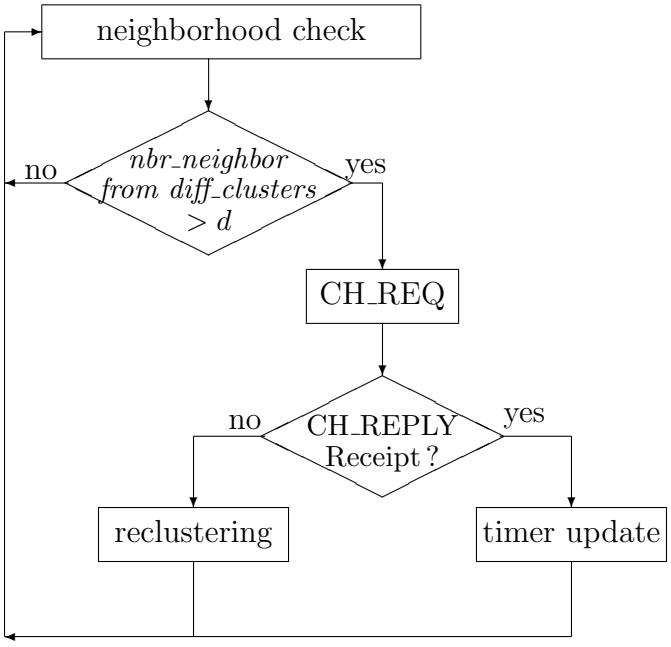


FIG. 5.2 – Ordinogramme pour la gestion de l'agrégation des *clusters*.

5.3.3.2 Perte du clusterhead

Le deuxième changement topologique qui pourrait perturber la structure du *cluster* est la perte du *clusterhead*. La partition est divisée en plusieurs sous-partitions qui, sauf une, se retrouvent dépourvues de *clusterhead*. On suppose qu'aucune autre partition ne se rapproche d'une des sous-partitions. Sinon, on se retrouve dans le cas d'agrégation de *clusters*, décrit dans le paragraphe précédent. Grâce à la technique d'horodatage, le noeud possède des informations sur l'instant d'émission par le *clusterhead* du dernier message HELLO. A l'expiration du temporisateur (*ch_last_sent_hello_msg*) associé au *clusterhead*, le noeud présume la perte de son *clusterhead* et devient orphelin. Nous utilisons le terme "présumer" car à cause de la mobilité, l'information sur l'état du *clusterhead* peut ne pas arriver à temps entraînant de fausses expirations du temporisateur. Si le noeud a plus de D_r voisins orphelins, ils procède de la même manière que dans l'agrégation des *clusters*, en sollicitant le *clusterhead* par l'envoi d'un message CH_REQ. Le *clusterhead*, s'il est encore dans la partition, est invité à confirmer sa présence dans le *cluster*. Si c'est une fausse expiration du temporisateur, le *clusterhead* répond par un message CH_REPLY et les membres mettent à jour leurs données. Si au bout d'un certain temps aucune réponse n'est reçue, alors la perte du *clusterhead* et confirmée et le processus de *reclustering* est déclenché.

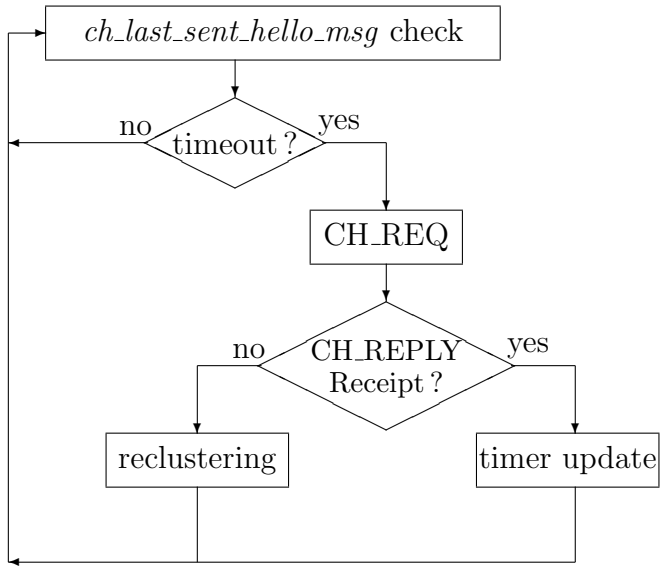


FIG. 5.3 – Ordinogramme pour la gestion de la perte du *clusterhead*.

Étant donné que ce protocole est basé sur la diffusion dans toute la partition des messages de contrôle, deux mesures sont prises pour éviter les tempêtes de *broadcast* et la contention. Afin d'éviter d'inonder le réseau par des messages CH_REQ et CH_REPLY, un noeud n'est permis d'envoyer ou de relayer qu'un seul message CH_REQ durant une période de *Hello_Period* secondes. De la même manière, un *clusterhead* n'est permis d'envoyer qu'un seul CH_REPLY durant une période de *Hello_Period* secondes. En outre, l'envoi des messages CH_REQ est différé d'une certain temps $T_{backoff}$ (exprimé en secondes) :

$$T_{backoff} = id_i/n,$$

où id_i est l'identifiant du noeud i and n est la taille du réseau.

Avant d'expliquer comment initialiser la valeur *ch_last_sent_hello_msg*, nous détaillons la structure des messages utilisés dans ce protocole.

1. HELLO (TP, id_i, id_{ch}, *ch_last_sent_hello_msg*, x, y) : TP est le type du message, id_i l'identifiant du noeud i, id_{ch} l'identifiant de son *clusterhead*, *ch_last_sent_hello_msg*, le champs supplémentaire ajouté, x et y sa position. Ce message est envoyé périodiquement par i à ses voisins directs.
2. CH_REQ (TP) : TP type du message. Ce message est envoyé en cas de perte du *clusterhead* ou en cas d'agrégation de *clusters*.
3. CH_REPLY (TP, id_i, T_i) : TP est le type du message, id l'identifiant du *clusterhead* i source du message et T_i son temps d'expiration. Il est envoyé par le *clusterhead* à la réception d'un message CH_REQ.

Tous les messages utilisés dans *Passive Maintenance* et *Periodical Broadcast* contiennent un champ TTL (*Time-To-Life*) et un numéro de séquence. Les deux messages utilisent la même structure des messages lors de la formation des *clusters* et du *reclustering* : FORMATION (TP, *source*, *winner*, *degree*, T_{winner}), où TP est le type du message, *source* source du message, *winner* l'identifiant du gagnant [2], *degree* le degré du gagnant et T_{winner} son temps d'expiration.

5.3.3.3 Phase d'initialisation

Le but de cette phase est d'initialiser la valeur du temporisateur *ch_last_sent_hello_msg*. Après la détection du partitionnement du réseau, on suppose que la formation d'un nouveau *cluster* est déclenchée à l'instant t_0. L'information sur le premier message HELLO envoyé par le *clusterhead* met du temps pour atteindre un membre du *cluster*. Ce temps est proportionnel au nombre de sauts qui sépare ce noeud à son *clusterhead* et à la période

[2]La signification du mot gagnant est donnée dans le paragraphe 3.3.3.4 : formation des *clusters*.

des messages HELLO. Ces messages véhiculent de proche en proche, sur une échelle spatiale et temporelle, l'information sur le *clusterhead*.

Pour illustrer la propagation de l'information sur le *clusterhead*, nous prenons l'exemple d'un *cluster* formé par quatre noeuds : un *clusterhead* (CH) et quatre membres (A, B, C et D) (figure 5.4). On note I l'état initial du réseau. Pour chaque membre, la variable *ch_last_sent_hello_msg* est égale à U (*Unknown*) puisqu'une information n'est disponible. A l'instant t_1 :
- CH envoie un message HELLO à ses voisins A et D.
- A envoie un message HELLO à CH et B.
- B envoie un message HELLO à A et C.
- C envoie un message HELLO à B.
- D envoie un message HELLO à CH.

A la fin de cet échange, la valeur de *ch_last_sent_hello_msg* est égale à t_1 pour A et U pour B et C. C'est uniquement à l'instant t_3 que C va apprendre que CH a envoyé un message HELLO à l'instant t_1.

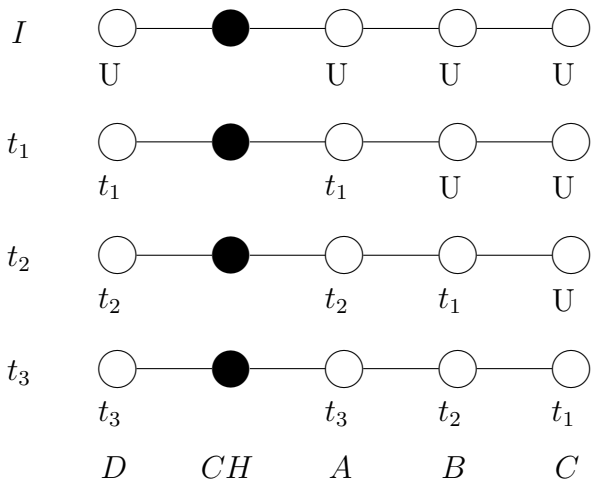


FIG. 5.4 – Diffusion de l'information sur l'état du *clusterhead* grâce aux messages HELLO et la technique d'horodatage.

Mais grâce à la mobilité des noeuds et la dynamique du réseau, l'information peut être propagée plus vite. On suppose qu'entre les deux instants t_3 et t_4, le noeud D change de position et devient voisin à C (figure 5.5). Le noeud C va recevoir des messages HELLO

de la part de B et D : B lui informe que la dernière information qui lui est parvenue de la part du *clusterhead* (CH) date de t_2 et D de t_3. Donc C choisit l'instant le plus proche et met à jour la variable *ch_last_sent_hello_msg* à t_3.

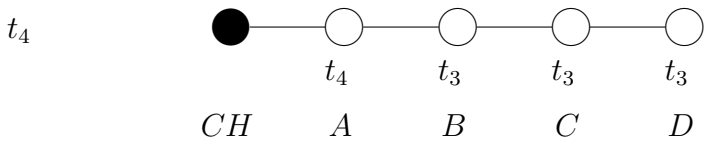


FIG. 5.5 – Amélioration de la diffusion de l'information sur l'état du *clusterhead* grâce à la mobilité des noeuds.

Lors de la phase de formation des *clusters*, KCMBC permet à chaque noeud u de connaître le nombre de sauts vers son *clusterhead* v. Afin d'éviter de fausses expirations, le temporisateur *ch_last_sent_hello_msg* est initialisé de la manière suivante :

$$ch\_last\_sent\_hello\_msg_{-0} = t_0 + h(u,v) \times \ Hello\_Period,$$

où $h(u,v)$ est le nombre de sauts entre u et v.

5.3.3.4 Validation par un réseau de Petri

Les protocoles de communication font partie des systèmes dynamiques à évènements discrets dont le comportement peut être modélisé grâce à un réseau de Petri (RdP) [103]. C'est un outil présenté par Carl Adam Petri en 1962, qui permet de décrire les relations existantes entre les évènements et les états du système.

Un RdP est composé de places représentées par des cercles, de transitions représentées par des rectangles et d'arcs orientés reliant les places et les transitions. Il est dit *graphe biparti alterné*, car tout arc relie une place à une transition ou une transition à une place et jamais deux places ou deux transitions. Un RdP utilise un mécanisme de marquage pour définir l'état du système à un instant donné. Le marquage consiste à attribuer un nombre entier (positif ou nul) de jetons dans chaque place du graphe. L'évolution temporelle de l'état système est modélisée par le passage des jetons d'une place à une autre. Cette opération de passage est appelée franchissement de la transition. L'évolution du système

est régie par deux règles principales, modélisant la concurrence et la synchronisation dans le système :

1. Le réseau évolue par le franchissement d'une seule transition à la fois.
2. On ne peut franchir une transition que lorsque chacune des places en amont possède au moins un jeton.

Le RdP modélisant le protocole *Passive Maintenance* est illustré dans le figure 5.6. Les termes *split* et *merge* désignent la perte du *clusterhead* et l'agrégation des *clusters* respectivement.
La modélisation grâce à un RdP permet d'analyser le fonctionnement du protocole et en déduire ses propriétés. Nous avons utilisé l'outil PIPE (*Platform Independant Petri net Editor*) [104] dans sa version 3.0 pour modéliser *Passive Maintenance* et les fonctionnalités offertes par le logiciel nous ont permis de valider les propriétés suivantes [3] :

État d'accueil : le réseau admet un état d'accueil M_a si pour tout état accessible du réseau, il existe une séquence de franchissements qui permet de retourner à M_a. Le réseau est dit réversible ou réinitiable si l'état initial est lui même l'état d'accueil. Pour *Passive Maintenance*, le réseau admet l'état "Affiliated" comme état d'accueil, représenté par l'état S_0 dans la figure 5.7 qui représente le graphe d'atteignabilité du réseau (*reachability*). Cette propriété correspond au caractère périodique de la procédure de maintenance.

Quasi-vivacité : depuis le marquage initial, toute transition peut être franchie au moins une fois. Toute transition qui n'est pas quasi-vivante est inutile.

Vivacité : cette propriété peut être démontrée grâce aux relations entre les deux premières propriétés. Un réseau quasi-vivant et réversible est aussi vivant [105]. Un réseau est vivant, si quelque soit l'évolution du réseau à partir du marquage initial, le franchissement à terme de toute transition est toujours possible. Le concept de vivacité est fortement lié à l'absence de blocage et démontre que le réseau a toujours la possibilité d'évoluer.

Distance synchrone : elle mesure le degré de dépendance mutuelle entre deux évènements. Une distance infinie entre l'évènement "perte du *clusterhead* " et l'évènement "agrégation des *clusters* " montre la cohérence du protocole.

Periodical Broadcast est un protocole de maintenance qui répond au cahier des charges. Il permet de gérer les changements topologiques et garantir l'unicité du *clusterhead* dans

[3] Quand on parle de "réseau", on désigne le réseau de Petri représentant le protocole *Passive Maintenance*.

la partition. Mais son caractère périodique peut engendrer un surcoût important en signalisation. La manière dont le *clusterhead* annonce sa présence dans la partition peut être améliorée en exploitant le fait que les noeuds communiquent l'identifiant de leur *clusterhead* à leurs voisins directs. Nous avons donc proposé un nouveau protocole de maintenance *Passive Maintenance* qui respecte le cahier de charges et dont le fonctionnement est validé grâce à un réseau de Petri. Nous allons démontrer par simulation, que notre protocole génère moins de trafic de signalisation dans le réseau que *Periodical Broadcast*.

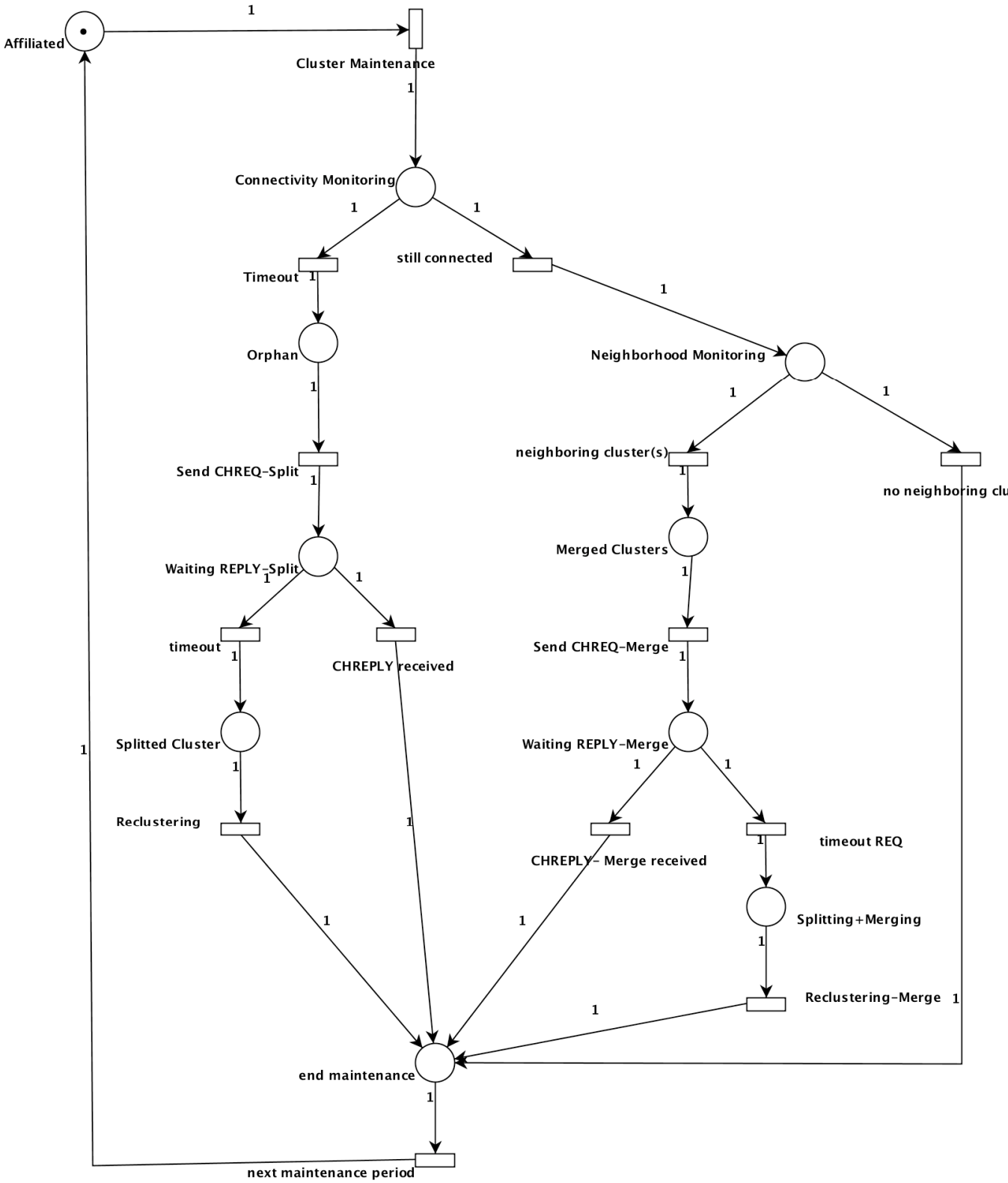


FIG. 5.6 – Modélisation de *Passive Maintenance* à l'aide d'un réseau de Petri.

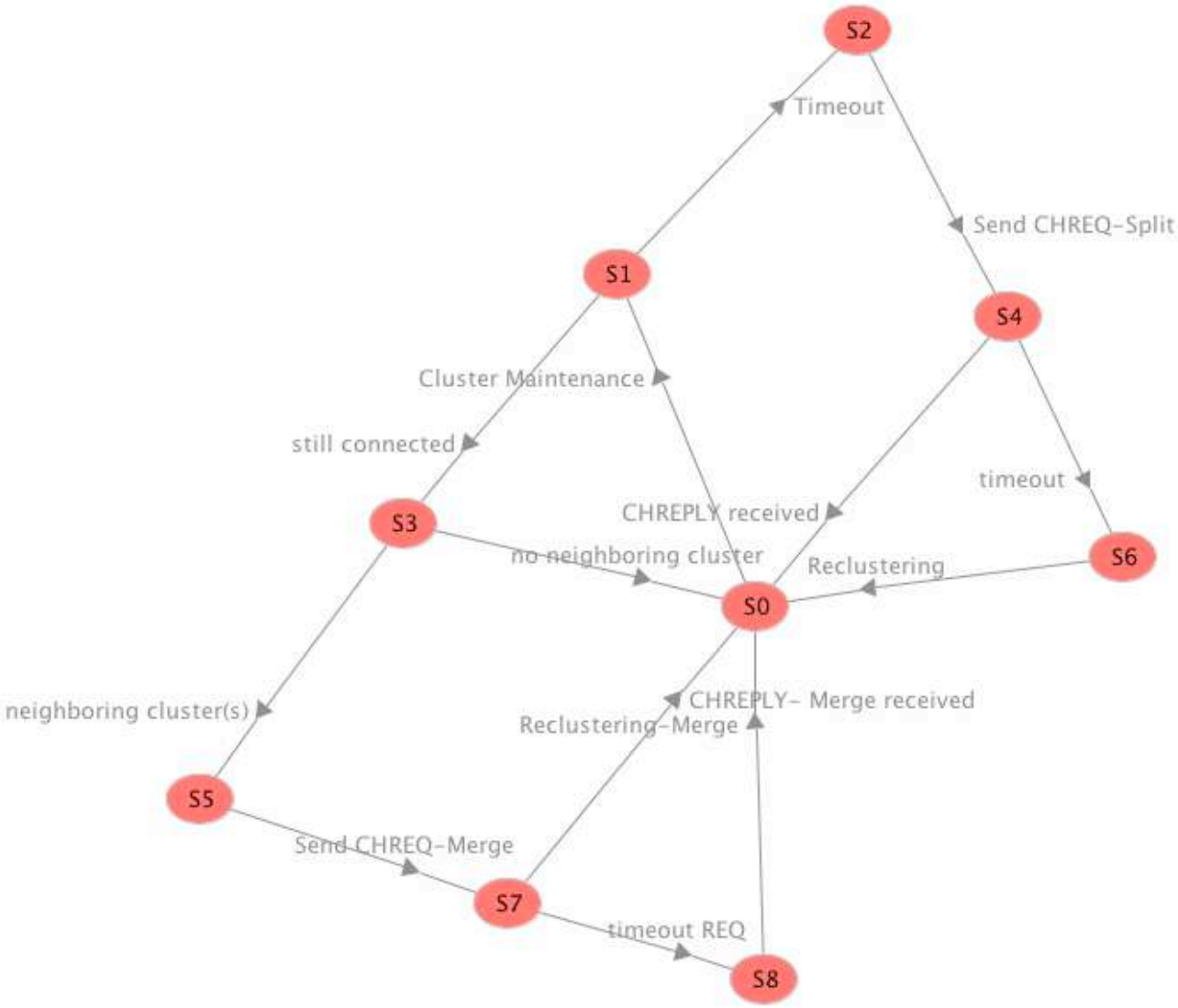


FIG. 5.7 – Graphe d'atteignabilité (*reachability*) du réseau modélisant *Passive Maintenance*.

5.4 Résultats de simulation

Nous commençons tout d'abord par comparer les deux protocoles en termes de qualité du *clustering* et du surcoût en signalisation. Ensuite, nous analysons le comportement de *Passive Maintenance*, en fonction des différents paramètres du modèle.

Les résultats de simulation sont moyennés sur 10 réalisations dont chacune dure 10000 s. Le tableau 5.1 illustre les paramètres de simulation communs aux deux protocoles de maintenance.

Paramètre	**Valeur**
Nombre de noeuds	37
Vitesse moyenne des noeuds (m/s)	4.4
Surface de simulation $(m \times m)$	1000×1000
Portée radio (m)	$[70, 100]$
Refresh_Period & *Hello_Period* (s)	1
k	12

TAB. 5.1 – Paramètres de simulation pour les protocoles de maintenance.

5.4.1 Comparaison de Periodical Broadcast et Passive Maintenance

5.4.1.1 Incohérence du clustering : définition

La modélisation qualitative des deux protocoles, présentée dans la section précédente, permet de vérifier la justesse logique de leurs modes de fonctionnement. Nous avons maintenant besoin de valider et comparer leurs performances. La qualité du *clustering* est définie grâce aux deux conditions suivantes :

1. Dans chaque partition, un seul noeud assure le rôle de *clusterhead*.
2. Chaque noeud doit être affilié à un *clusterhead*.

La première phase du *clustering* (formation des *clusters*) assure l'existence et l'unicité du *clusterhead* dans la partition. Mais lors de la deuxième phase (maintenance), ces deux conditions ne sont pas toujours respectées, notamment avec la perte du *clusterhead* et l'agrégation des *clusters*. La structure du *cluster* devient alors incohérente. Pour la comparaison des deux protocoles, nous définissons l'incohérence comme étant le pourcentage du temps durant lequel une des deux conditions définissant la cohérence du *clustering*

n'est pas respectée. Nous fixons les paramètres de chaque protocole de manière à avoir des niveaux d'incohérence équivalents, ce qui garantit la justesse de la comparaison d'un point de vue signalisation. Nous rappelons que D_r est utilisé dans les deux protocoles comme seuil pour le déclenchement du processus de *reclustering* par les noeuds orphelins. Le paramètre d est défini uniquement pour *Passive Maintenance* et est utilisé comme seuil pour l'envoi des messages CH_REQ en cas d'agrégation des *clusters*. La valeur de T_{out} indique au bout de combien de temps, un membre considère que son *clusterhead* a quitté la partition, si aucune information sur son état n'est reçue. Le tableau 5.2 résume les paramètres de chaque protocole de maintenance et la figure 5.8 montre le niveau d'incohérence dans les deux protocoles en fonction de la portée radio.

Paramètre	***Periodical Broadcast***	***Passive Maintenance***
D_r	1	1
d	non défini	1
T_{out} (s)	3	2

TAB. 5.2 – Paramétrage des deux protocoles de maintenance dans le but d'avoir des niveaux d'incohérence équivalents.

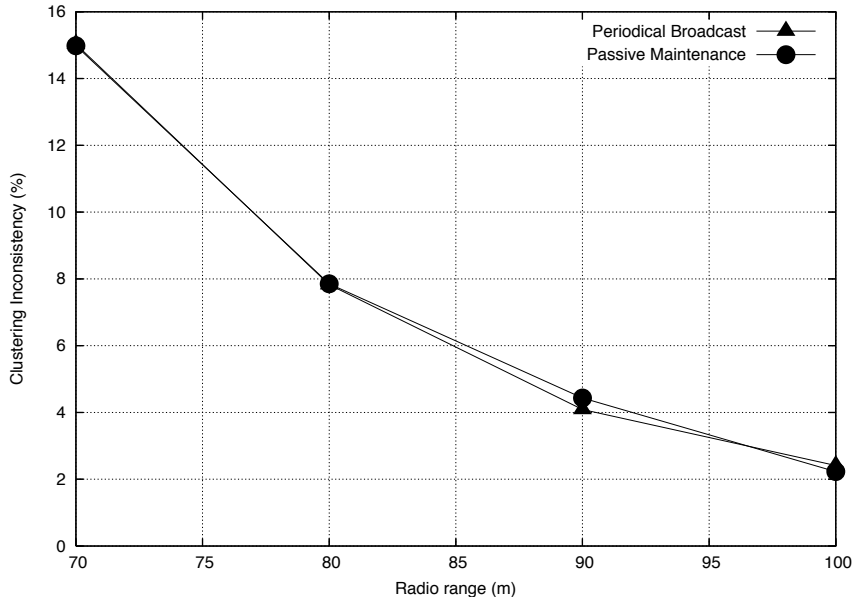


FIG. 5.8 – Niveaux d'incohérence dans les deux protocoles (en fonction de la portée radio).

5.4.1.2 Performance des protocoles

A des niveaux de qualité similaires, nous pouvons correctement comparer les deux protocoles en termes du surcoût en signalisation. Pour cette comparaison, nous considérons indépendamment les deux parties constituant un message : l'en-tête et la charge utile. La taille de l'en-tête dépend de la pile protocolaire, du modèle et du mécanisme d'encapsulation des paquets. Pour la couche transport par exemple, les protocoles UDP (*User Datagram Protocol*) et TCP (*Transmission Control Protocol*) ajoutent une entête de 8 et 20 octets respectivement. Ce datagramme est ensuite encapsulé dans un paquet IP (*Internet Protocol*) où 20 octets sont ajoutées pour l'entête, et ainsi de suite jusqu'aux couches les plus basses. Étant donné que les paquets de signalisation sont transmis sans subir de fragmentation, la taille de l'entête est la même pour tous les messages quelque soit leur type. Pour plus de généralité, nous considérons donc le nombre de messages transmis comme indicatif du surcoût induit par la partie en-tête et les encapsulations multiples qui en découlent. La figure 5.9 représente le nombre de messages envoyés par noeud en fonction de la portée radio. Ces messages incluent les messages FORMATION (formation des *clusters* et *reclustering*) pour les deux protocoles, REFRESH pour *Periodical Broadcast* et CH_REQ et CH_REPLY pour *Passive Maintenance*. Le surcoût en signalisation causé par les messages HELLO n'est pas considéré ici car le nombre de messages HELLO envoyés est dicté par la durée totale du partitionnement qui dépend uniquement du modèle de mobilité (*FireMobility* est utilisé pour la simulation des deux protocoles) [4]. La première remarque concerne l'allure générale des deux courbes. Le nombre de messages envoyés est une fonction décroissante de la portée radio. En effet, le nombre de messages dépend essentiellement de la durée pendant laquelle l'algorithme de *clustering* est exécuté. Nous avons vu dans la section 3.1 que la communication avec le satellite doit être coupée dès le rétablissement de la connexité dans le réseau. L'algorithme de *clustering*, dont le but est le choix des noeuds qui ont accès aux communications satellite, n'est alors exécuté que lorsque le réseau est partitionné. Or, la durée pendant laquelle le réseau est partitionné décroît graduellement en fonction de la portée radio 4.11. La deuxième remarque se rapporte aux performances des deux protocoles. Dans *Passive Maintenance*, un noeud consomme en moyenne 75% moins de messages que dans *Periodical Broadcast*, pour une portée radio égale à 70 m et ce gain peut atteindre 84% pour une portée radio égale à 100 m. Cette amélioration est due au fait que l'envoi des messages est sollicité uniquement en cas d'un changement topologique. Tant que le réseau est stable et qu'il n'y a ni perte de *clusterhead* ni agrégation de *clusters*, les noeuds n'envoient pas de messages de maintenance.

[4] Par contre, le surcoût en signalisation causée par la charge utile des messages HELLO sera bien pris en compte ultérieurement.

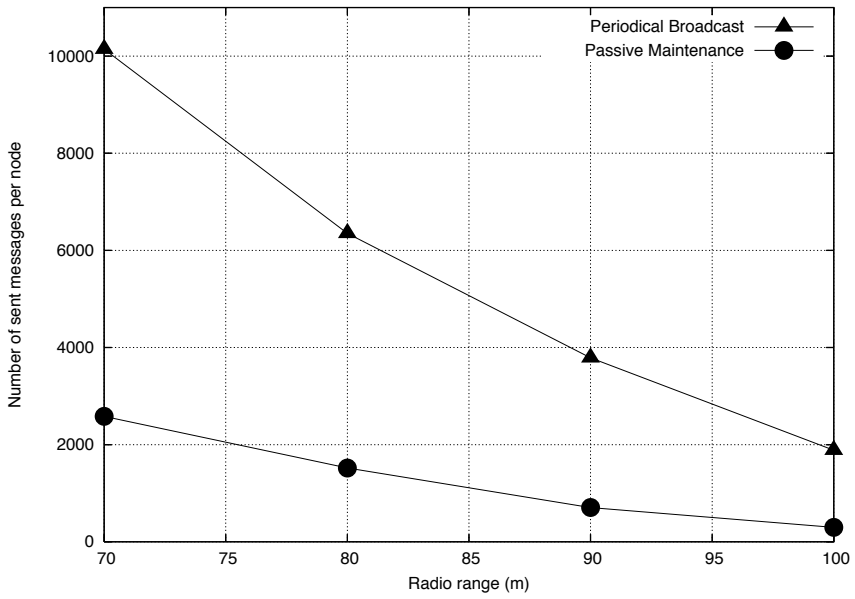


FIG. 5.9 – Comparaison de *Periodical Broadcast* et *Passive Maintenance* : nombre de messages de signalisation envoyés par noeud durant toute la simulation.

Afin de valider les résultats de simulations et de la même manière que dans le chapitre 4, nous avons examiné les intervalles de confiance à 95% pour le nombre de messages envoyés (Figure 5.10). Nous avons remarqué que les intervalles de confiance sont suffisamment petits pour affirmer la représentativité des résultats et la prééminence de *Passive Maintenance*.

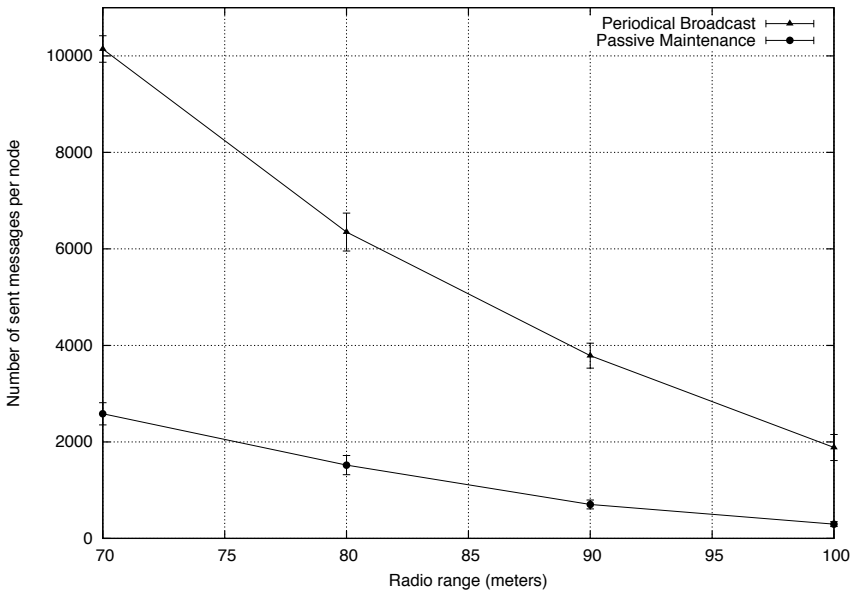


FIG. 5.10 – Validation des résultats pour le nombre de messages envoyés : intervalles de confiance à 95%.

Le deuxième volet de cette comparaison de performances traite le surcoût engendré par la partie charge utile des messages. La taille de la charge utile dépend des informations transportées et de leur type. La description de structure des messages utilisés dans chaque protocole est donnée en détail dans le paragraphe 5.3.2 pour *Periodical Broadcast* et le paragraphe 5.3.3 pour *Passive Maintenance*. Nous considérons qu'un entier est codé sur un octet. Les adresses des noeuds et les nombres réels sont codés sur quatre octets. Étant donné que *Passive Maintenance* change la structure des messages HELLO, nous considérons dans ce paragraphe le surcoût engendré par le champ supplémentaire (*ch_last_sent_hello_msg*). Ce champ est utilisé pour l'horodatage des messages HELLO envoyés par les *clusterheads*. La figure 5.11 montre qu'ici aussi *Passive Maintenance* réalise une meilleure performance. Mais le gain est moins important par rapport à celui réalisé sur le nombre de messages envoyés : 45% pour une portée radio égale à 70 m et 51% pour une portée radio égale à 100 m. Certes *Passive Maintenance* n'utilise des messages dédiés à la maintenance qu'en cas de besoin, mais cette procédure ne peut éviter une certaine périodicité dans le transfert de l'information sur le *clusterhead*. Le champ *ch_last_sent_hello_msg* est inclus dans les messages HELLO dont l'envoi est périodique.

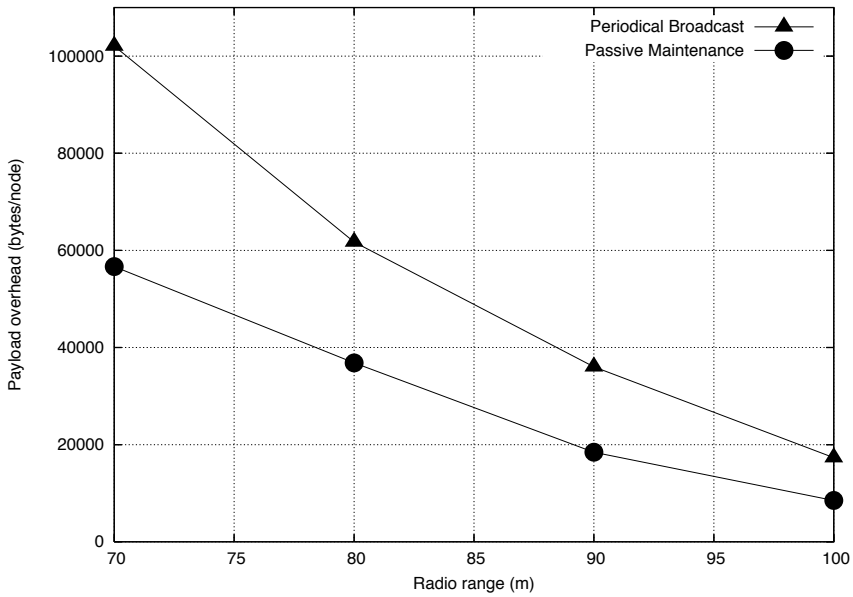


FIG. 5.11 – Comparaison de *Periodical Broadcast* et *Passive Maintenance* : la surcharge induite par la partie charge utile des messages de signalisation en octets par noeud (durant toute la simulation).

La question qu'on pourrait se poser est : est-ce que cette amélioration reste valable si on change le paramétrage du modèle ? Nous avons essayé de répondre à cette question, en modifiant la valeur de T_{out} comme suit : elle est maintenant à égale à 6 s pour *Periodical Broadcast* et 4 s pour *Passive Maintenance*. Nous procédons de la même manière en comparant le niveau d'incohérence (figure 5.12), le nombre de messages envoyés par noeud (5.13) et la surcharge induite par la partie charge utile des messages (5.14). Les résultats de simulation montrent qu'en changeant le paramétrage du modèle, *Passive Maintenance* réalise encore une meilleure performance que *Periodical Broadcast*. En augmentant la valeur de T_{out}, on constate que le niveau d'incohérence augmente mais le surcoût en signalisation diminue. On remarque aussi que la valeur de T_{out} a un plus grand impact sur *Passive Maintenance* que sur *Periodical Broadcast*. Pour une portée radio égale à 70 m, le nombre de messages par noeud diminue de 790 pour *Passive Maintenance* et de 33 pour *Periodical Broadcast* et pour la charge utile, on gagne 3460 octets par noeud pour *Passive Maintenance*, tandis que le gain n'est que de 265 octets pour *Periodical Broadcast*. Cette différence est expliquée par le fait que l'envoi des messages dans *Passive Maintenance* est lié à la détection d'un changement topologique. La valeur de T_{out} contrôle la détection de la perte du *clusterhead*. La structure et la fréquence des messages dans *Periodical Broadcast* sont pratiquement indépendantes des changements topologiques.

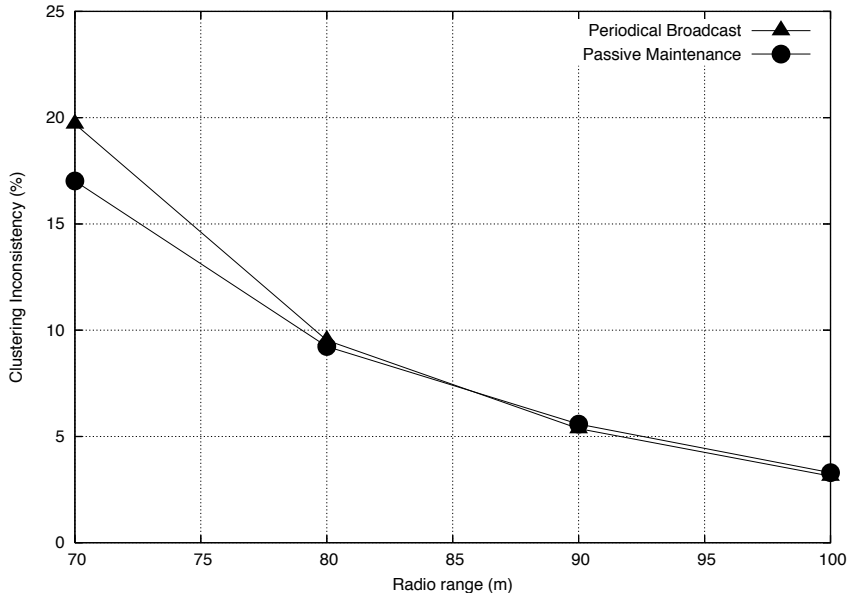


FIG. 5.12 – Niveaux d'incohérence dans les deux protocoles (en fonction de la portée radio).

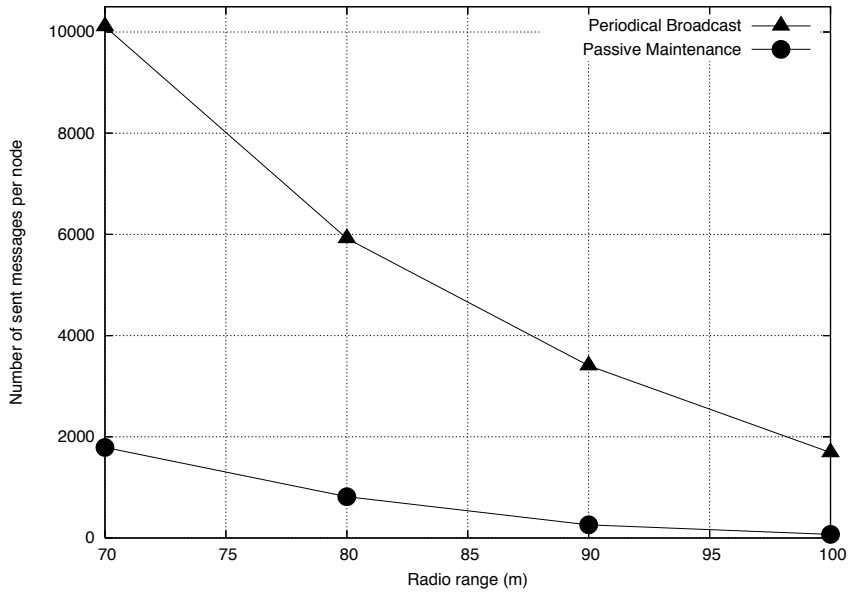


FIG. 5.13 – Comparaison de *Periodical Broadcast* et *Passive Maintenance* : nombre de messages envoyés par noeud durant toute la simulation.

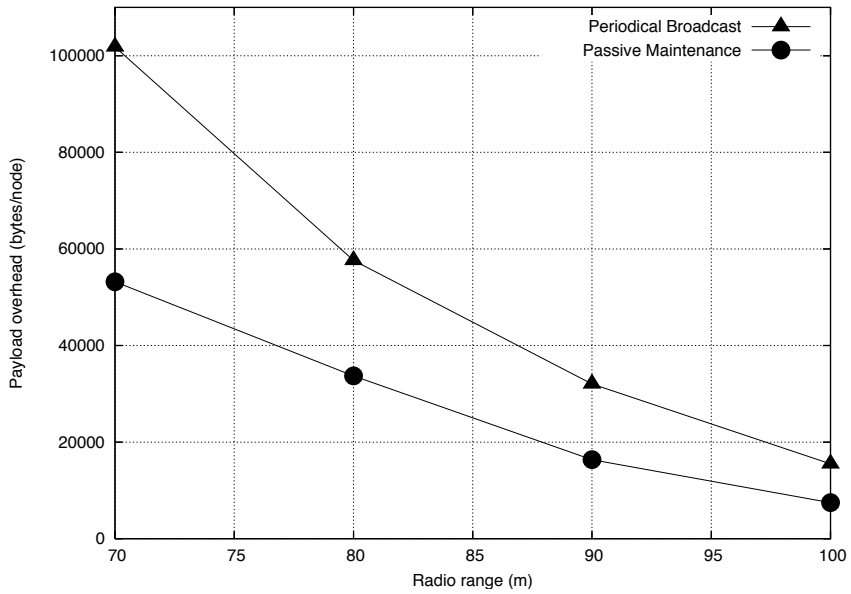


FIG. 5.14 – Comparaison de *Periodical Broadcast* et *Passive Maintenance* : la surcharge induite par la partie charge utile des messages en octets par noeud (durant toute la simulation).

La comparaison entre *Periodical Broadcast* et *Passive Maintenance* n'a pas abordé uniquement les différences au niveau du nombre de messages envoyés lors de la phase de maintenance, mais aussi le surcoût engendré par la partie charge utile des messages. En gardant des niveaux de qualité équivalents, notre protocole, *Passive Maintenance* réalise de meilleures performances que *Periodical Broadcast*. La prééminence de *Passive Maintenance* dans le cas général est démontrée grâce à la variation du paramétrage des modèles.

Nous nous intéressons par la suite, à l'analyse du *Passive Maintenance* et l'évaluation de l'impact des différents paramètres sur les performances du protocole.

5.4.2 Analyse du modèle de Passive Maintenance

Le modèle de *Passive Maintenance* est défini grâce à trois paramètres : D_r, d et T_{out}. Dans cette section, le but est d'analyser le comportement du modèle en fonction de ces paramètres et caractériser leur impact sur les performances du protocole. Pour cette étude nous fixons la valeur de la portée radio à 70 m, étant donné que l'impact de la portée radio a été évoqué dans la section précédente.

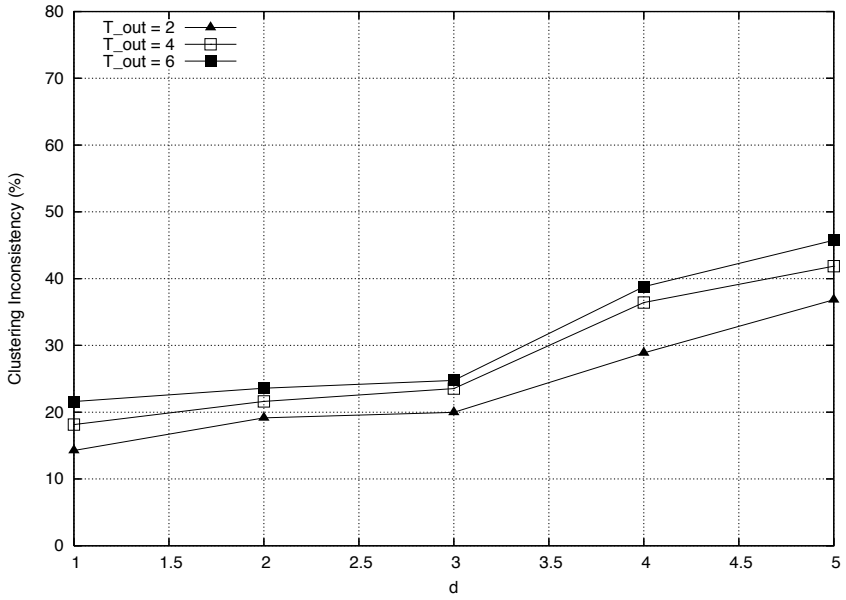
5.4.2.1 Impact des relations de voisinage

Nous considérons tout d'abord l'impact de D_r et d sur les performances du protocole. Ces deux paramètres illustrent les relations de voisinage dans le réseau, étant donné que l'envoi des messages est conditionné par le nombre de voisins répondant à certains critères. La détection de la perte du *clusterhead* est liée au nombre d'orphelins dans le voisinage qui doit être supérieur à D_r. Dans le cas d'agrégation des *clusters*, c'est le nombre de voisins appartenant à des *clusters* différents qui doit être supérieur à d. On rappelle que le degré moyen d'un noeud dans *FireMobility* est égal à 6.6 pour une portée radio égale à 70 m. La figure 5.15 montre que le niveau d'incohérence augmente avec l'accroissement de D_r et d. Quand on augmente la valeur de D_r et de d, on diminue le nombre de noeuds capables de détecter la perte du *clusterhead* et l'agrégation des *clusters* respectivement. Or la conception de *Passive Maintenance* est fortement liée à ces changements topologiques. Si le réseau n'arrive pas à les détecter, aucun message CH_REQ n'est envoyé et l'incohérence s'accentue.

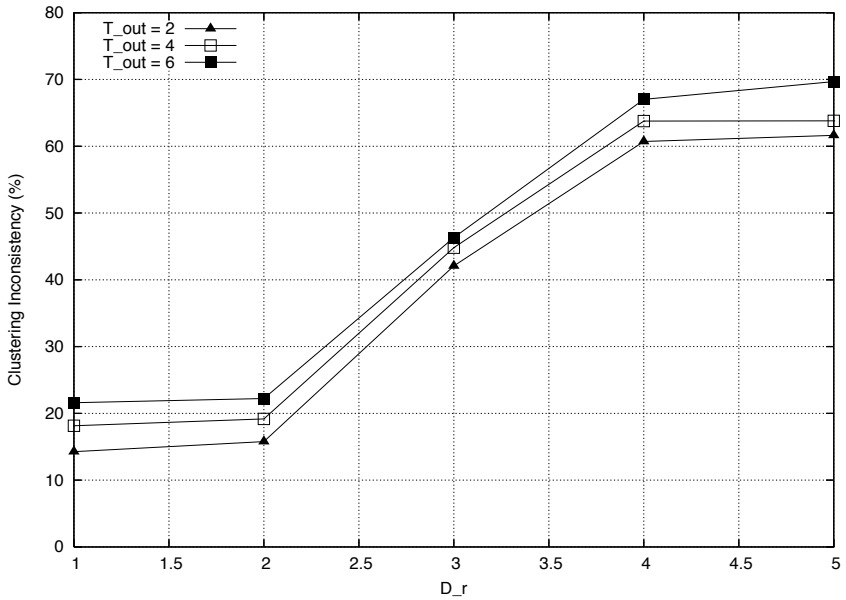
On remarque que la dégradation causée par D_r est plus importante que celle causée par d. Pour $T_{out} = 4\ s$ par exemple, l'incohérence passe de 21.6% à 36.4% (+14.8%) quand d augmente de 2 à 4 et de 19.1% à 63.7% (+44.6%) quand D_r augmente de 2 à 4. Ceci est expliqué par l'intervention du temporisateur *ch_last_sent_hello_msg* dans la réorganisation des *clusters*. Quand la valeur de d est grande, les noeuds se retrouvent incapables d'envoyer des message CH_REQ et l'agrégation des *clusters* n'est pas signalée dans la partition. Mais les noeuds qui viennent rejoindre la partition sont déconnectés de leur *clusterhead*. A l'expiration du temporisateur, les noeuds deviennent orphelins et utilisent dans ce cas le mécanisme de perte de *clusterhead* pour envoyer des messages CH_REQ et solliciter le *clusterhead* de la partition.
Moins d'évènements topologiques détectés signifie aussi moins de messages envoyés dans le réseau. La figure 5.16 montre que l'augmentation de d et de D_r réduit le nombre de messages envoyés. Mais, comme pour la comparaison des niveaux d'incohérence, cette réduction est plus importante avec l'augmentation de D_r qu'avec celle de d.

Le choix de paramétrage est un compromis entre la qualité de service souhaitée en vue

de la cohérence de la structure des *clusters* et le surcoût en signalisation engendré par la procédure de maintenance. Accroître la valeur de d et/ou D_r détériore la qualité du *clustering* mais diminue le nombre de messages échangés dans le réseau.

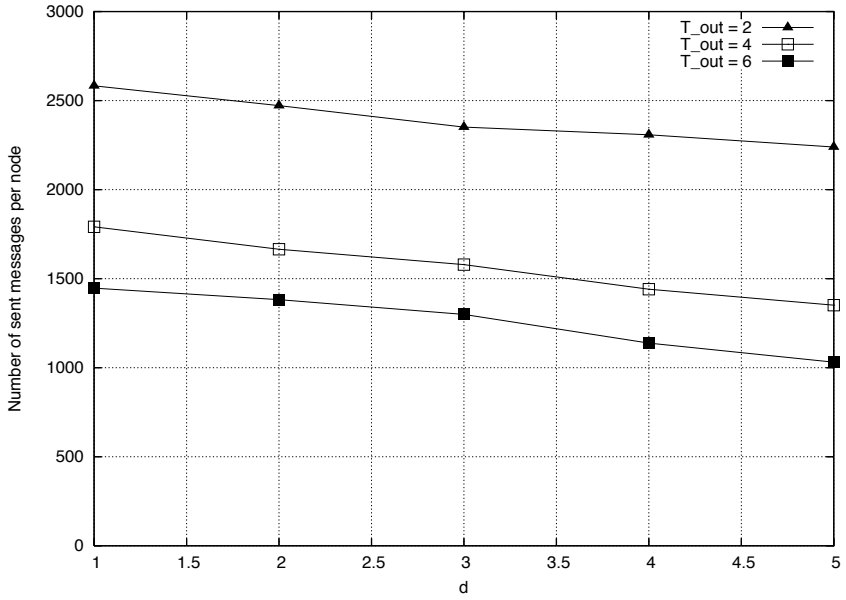


(a)

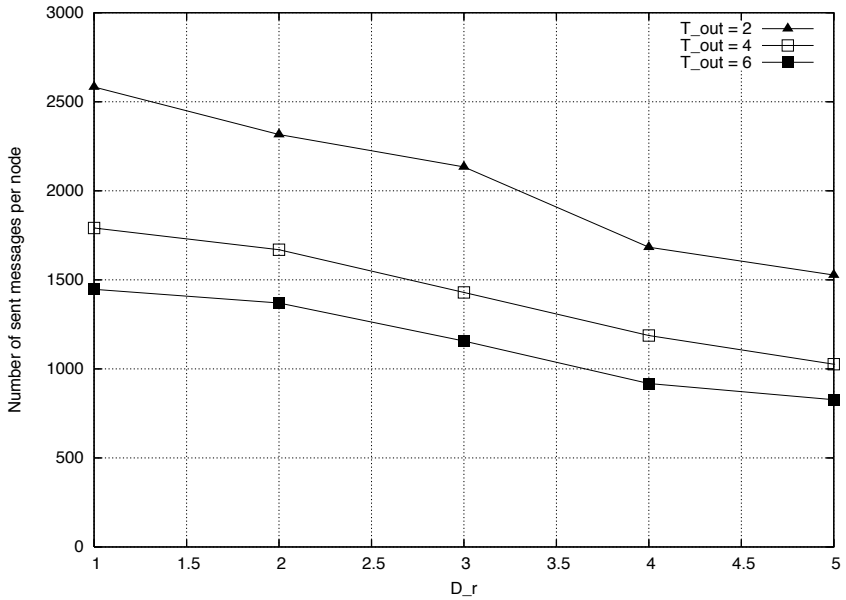


(b)

FIG. 5.15 – Niveau d'incohérence en fonction de d avec D_r=1 (a) et en fonction de D_r avec d=1 (b).



(a)



(b)

FIG. 5.16 – Nombre de messages envoyés par noeud en fonction de d avec D_r=1 (a) et en fonction de D_r avec d=1 (b).

5.4.2.2 Impact du temporisateur

La seconde partie de cette évaluation considère l'impact du paramètre T_{out}. Ce dernier caractérise la perte du *clusterhead* : au bout de T_{out} secondes, si aucune nouvelle information sur l'état du *clusterhead* n'est arrivée, le noeud présume que ce dernier n'est plus dans la même partition que la sienne. Nous soulignons l'usage du mot *présumer* car à cause de la dynamique du réseau, de fausses expirations du temporisateur peuvent avoir lieu. Nous considérons alors le pourcentage de ces fausses expirations par rapport au nombre de total d'expirations. La figure 5.17 montre que l'augmentation de la valeur de T_{out} diminue le pourcentage des fausses expirations. Plus la valeur de T_{out} est grande, plus on laisse du temps à l'information pour arriver jusqu'au noeud et au réseau pour traiter des situations de blocage causées par la mobilité des noeuds. Il intéressant de noter que la pente de la courbe représentant les fausses expirations est plus importante que celle des courbes représentant le niveau d'incohérence et le nombre de messages envoyés. Par exemple, si on fixe T_{out} à 6 s, on obtient 6.4% de fausses expirations au lieu de 20.6% avec T_{out} égale à 5 s. Au même temps, le nombre de messages chute de 94 messages et passe de 1541 à 1447. Le niveau d'incohérence augmente uniquement de 0.56% et passe de 21.04% à 21.6%.

Les paramètres liés aux relations de voisinage d et D_r agissent différemment sur les performances du réseau à cause de l'interférence du temporisateur associé au *clusterhead*. Si l'augmentation de leurs valeurs fait chuter le nombre de messages envoyés, elle détériore e n même temps la qualité du *clustering*. L'expiration du temporisateur *ch_last_sent_hello_msg* n'est pas toujours équivalente à une perte de *clusterhead*. Donc l'analyse de ce paramètre a nécessité l'ajout d'un troisième critère qui est le pourcentage de fausses expirations. De la même manière qu'avec d et D_r, l'augmentation de T_{out} détériore la qualité du *clustering* mais elle fait chuter le nombre de messages envoyés grâce à la réduction du pourcentage des fausses expirations.

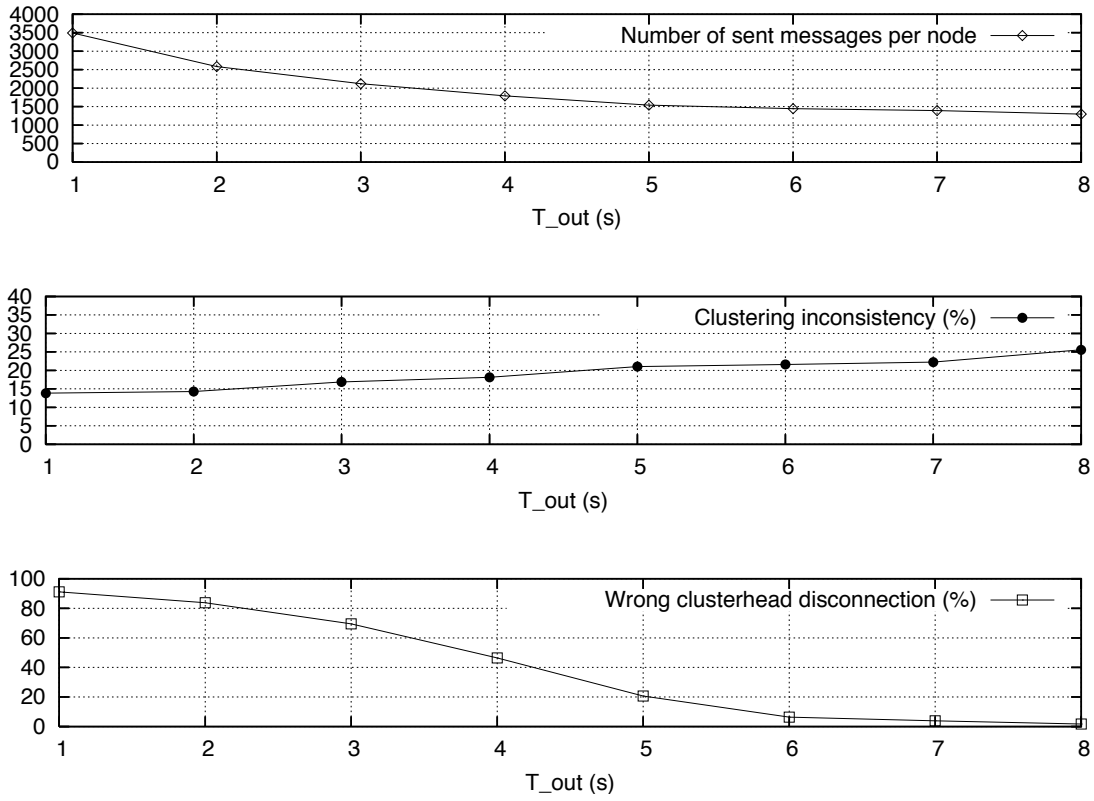


FIG. 5.17 – Impact de T_{out} sur le nombre de messages envoyés, l'incohérence du *clustering* et le pourcentage de fausses expirations du temporisateur (avec $d = 1$ et $D_r = 1$).

5.5 Conclusion

Dans ce chapitre, nous avons tout d'abord passé en revue les différentes approches qui permettent de réduire le surcoût en signalisation de la maintenance. Une première manière d'aborder le problème est de réduire les évènements conduisant à la réorganisation des *clusters*. Cette approche est plutôt préventive. La relaxation des conditions sur le *clustering* et la création des *clusters* stables limitent l'appel à la procédure de maintenance et évitent au maximum l'utilisation des messages de contrôle. Mais elles ne s'intéressent pas aux messages de contrôle utilisés. Certaines contributions tels que FWCA font l'effort de préciser la structure des messages utilisés et leur fréquence. Ces contributions se focalisent essentiellement sur le *clustering* à un seul saut. Dans le cas du *clustering* à multi-sauts, la principale préoccupation des algorithmes de *clustering* est de considérer les cas qui altèrent la qualité du *clustering* et définir le comportement des noeuds face aux changements topologiques. Dans les deux cas, l'évaluation et la réduction surcoût en

signalisation n'est pas le problème primordial.

Bellavista et Magistretti avec leur protocole *Periodical Broadcast* ont essayé d'évaluer l'impact de la mobilité sur la qualité du *clustering*. Leur protocole répond à un cahier des charges développé spécialement pour la maintenance dans un réseau dynamique où plusieurs partitions peuvent se former. Nous partons de ce cahier de charges et exploitons les particularités de maintenance dans un réseau partitionné par rapport à un réseau complètement connexe, pour proposer un nouveau protocole de maintenance, *Passive Maintenance*.

Nous avons démontré grâce aux simulations que *Passive Maintenance* réalise de meilleures performances que *Periodical Broadcast*. L'évaluation a comporté deux niveaux de comparaison : l'entête et la charge utile des messages échangés dans le réseau. Nous nous sommes ensuite intéressés à l'analyse du comportement de *Passive Maintenance* en fonction des différents paramètres du modèle. Si l'amélioration du niveau de cohérence dans le réseau passe forcément par l'utilisation de plus de messages, un compromis est à trouver entre une bonne qualité du *clustering* et le nombre de messages envoyés dans le réseau. Une bonne qualité de *clustering* signifie une réactivité plus rapide du réseau face aux changements topologiques. Mais un important surcoût en signalisation nécessite plus d'énergie pour le traitement des messages et signifie plus de collision et d'interférences dans un environnement sans fil.

Chapitre 6

Structuration des réseaux ad hoc mesh

Sommaire

La technologie *mesh* permet la connexion et la déconnexion des noeuds de manière automatique et sans interruption du service. La structure d'un réseau *mesh* est présentée dans le paragraphe 2.2.2. Cette technologie a été conçue au départ pour répondre aux attentes des armées qui, en terre étrangère, n'ont aucune infrastructure de communication sur les champs de batailles. Le réseau déployé peut ainsi couvrir des zones géographiques étendues sans avoir besoin de relier les noeuds par des liaisons filaires contraignantes.

Cette technologie a gagné de l'intérêt dans plusieurs applications civiles. Elle est par exemple, utilisée pour étendre la couverture réseau aux zones blanches où le déploiement d'une infrastructure terrestre est très coûteux. Grâce à la flexibilité du réseau avec les propriétés d'auto-réparation et auto-configuration, les réseaux *mesh* sont aussi d'un grand intérêt pour les communications d'urgence.

Dans ce chapitre, nous étudions une architecture similaire à celle proposée dans le cadre du projet SAVION (2005 - 2007) [106]. Ce projet s'intéresse à fournir des services de communications dans les scénarios d'urgence, grâce à un système intégré composé d'un segment satellite et un réseau ad hoc. Le but du projet SAVION est la conception des terminaux qui jouent le rôle de passerelle entre le réseau ad hoc et le segment satellite. Nous considérons une autre problématique liée à cette architecture qui est le positionnement de ces passerelles. On considère un réseau ad hoc de type *mesh* où un certain nombre de noeuds assurent, via le satellite, l'interconnexion avec le centre des opérations et de contrôle. Le but est de minimiser le nombre de ces noeuds, tout en assurant une certaine qualité de service. Nous appelons ce problème *Satellite Capability Placement Problem* ou SCPP. Dans la littérature, nous avons identifié deux problématiques qui se rattachent à notre étude : le *Gateway Placement Problem* et le *Discrete Network Design Problem*.

6.1 Gateway Placement Problem

Dans un réseau *mesh*, certains routeurs possèdent des fonctionnalités supplémentaires : ils assurent l'interconnexion avec l'infrastructure existante et le routage vers un réseau extérieur tel qu'Internet. Contrairement aux réseaux ad hoc mobiles, les communications sont à destination ou en provenance de ces routeurs particuliers appelés passerelles. Le placement de ces derniers a une grande influence sur les performances du réseau. Le *Gateway Placement Problem* s'intéresse à la sélection du nombre optimal de passerelles qui permet de garantir une qualité de service minimale [101]. La figure 6.1 montre un exemple d'un réseau *mesh* où une passerelle (*gateway*) joue le rôle de pont entre le réseau *mesh* et le monde IP extérieur.

La structuration du réseau est fondamentale, car elle détermine sa capacité en termes de débits disponibles. Étant donné que tout le trafic vers l'extérieur passe par ces passerelles, elles peuvent être des goulots d'étranglement si leur emplacement n'est pas optimisé. Le *Gateway Placement Problem* s'intéresse donc à la sélection d'un sous-ensemble de routeurs *mesh* pour assurer le rôle de passerelles. Ce problème peut être formulé, dans un premier temps, sous la forme d'un problème d'optimisation non-linéaire en nombres entiers (*Integer Programming Problem*) [36].

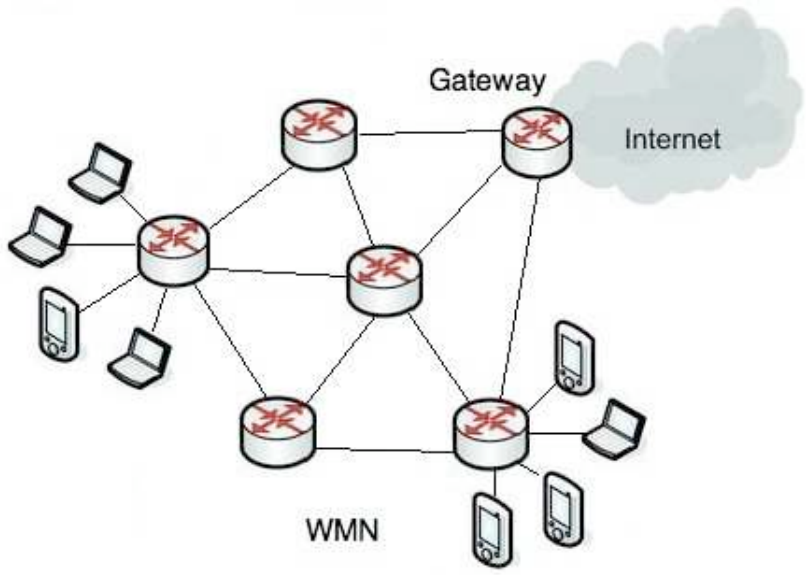


FIG. 6.1 – *Gateway Placement Problem* dans un réseau *mesh*.

6.1.1 Optimisation non-linéaire en nombres entiers

Le premier objectif du problème est de minimiser le nombre de passerelles afin de minimiser les coûts d'installation. Le second objectif est de minimiser le nombre de sauts requis pour établir une connexion entre un routeur et une passerelle. En effet, la probabilité de perte et les délais de transmission sont proportionnels à la longueur du chemin parcouru.

6.1.1.1 Modèle du réseau

Une passerelle est aussi un routeur. Mais dans ce paragraphe, quand on parle de routeur, on désigne un noeud sans accès direct à Internet. Le terme "noeud" désigne indifféremment un routeur ou une passerelle.

On note $V = \{v_1, \cdots, v_n\}$ l'ensemble des n noeuds du réseau, où il y a m noeuds qui seront configurés comme passerelles ($m < n$). On note cet ensemble $I = \{I_1, \cdots, I_m\}$ et on utilise la variable binaire χ_i pour l'identifier.

$$\chi_i = \begin{cases} 1 & \text{si le noeud } v_i \in I \\ 0 & \text{sinon} \end{cases}$$

On utilise une deuxième variable binaire $\varpi_{(i,j)}$ pour représenter la relation d'affiliation :

$$\varpi_{(i,j)|(i\neq j)} = \begin{cases} 1 & \text{si le routeur } \upsilon_j \text{ dirige son trafic vers la passerelle } \upsilon_i \\ 0 & \text{sinon} \end{cases}$$

Le choix d'une passerelle dépend fortement de sa capacité à gérer le trafic provenant des autres routeurs en direction d'Internet et inversement. Cette capacité dépend du nombre d'interfaces radio, du nombre de canaux disponibles et finalement du débit sur chaque canal. On note donc :

* $\rho(\upsilon_i) = \{1, 2, \cdots, |\rho(\upsilon_i)|\}$: l'ensemble des interfaces radio du noeud υ_i.
* $CHA = \{1, 2, \cdots, c\}$: l'ensemble des canaux radio existant dans le système.
* dtr_i (*data transfert rate*) : le débit sur le canal $i \in CHA$, exprimé en bits/s.

6.1.1.2 Modélisation de la capacité du réseau

Pour cette modélisation, on considère que tout le trafic généré par les noeuds du réseau est à destination du monde extérieur et transite nécessairement par les passerelles. On distingue deux types de trafic Internet dans un routeur :
- Un trafic local $T_l(\upsilon_i)$: c'est le trafic généré par les clients *mesh* attachés au routeur υ_i.
- Un trafic relayé $T_r(\upsilon_i)$: c'est le trafic généré par un routeur quelconque et relayé par le routeur υ_i à destination d'une passerelle.

Un routeur doit pouvoir gérer son propre trafic généré par ses clients *mesh* et aussi le trafic de relais généré par les autres routeurs. Cette condition se traduit par :

$$DTR_{\upsilon_i} \geq T_l(\upsilon_i) + T_r(\upsilon_i),$$

où DTR_{υ_i} est la capacité du routeur υ_i, exprimé en bits/s.

De la même manière, la passerelle choisie doit satisfaire la demande en termes de trafic des routeurs qui lui sont associés. Étant donnée que la passerelle est connectée à l'infrastructure existante, elle dispose d'une interface supplémentaire par rapport aux autres routeurs, dont la débit est noté $DTR_{passerelle}$. A un instant donné, le débit maximal est exprimé comme suit :

$$DTR_{passerelle} = \sum_{i=1}^{|\rho(\upsilon_i)|} dtr_i.$$

6.1.1.3 Modèle mathématique

Notations utilisées :

* $h(i,j)$: le nombre de sauts entre les noeuds υ_i et υ_j.
* $e=(j,k)$: le lien entre les noeuds υ_j et υ_k.
* $f^i_{j,e}$: le trafic généré par le routeur υ_j et routé sur le lien e à destination de la passerelle υ_i.
* R_{max} : le nombre maximal de sauts entre un routeur et une passerelle.
* $\lambda^i_{j,k}$: la variable binaire qui représente la relation de voisinage entre les noeuds υ_j et υ_k, relativement à la passerelle υ_i. On a $\lambda^i_{j,k}=1$, si $h(i,j)-1=h(i,k)$ et $h(j,k)=1$; c'est-à-dire : si υ_i est choisi comme passerelle, υ_k est le prochain saut à partir de υ_j vers υ_i.

Le modèle mathématique pour résoudre le problème d'emplacement des passerelles est le suivant :

$$\min \sum_{i}^{n} \chi_i, \tag{6.1}$$

$$min \sum_{i}^{n} \sum_{j}^{n} h(i,j).\varpi_{(i,j)} \tag{6.2}$$

sous les conditions suivantes :

$$\sum_{\upsilon_i \in I} \varpi_{(i,j)} = 1, \qquad \forall \upsilon_j \in V \tag{6.3}$$

$$\varpi_{(i,j)} \leq \chi_i, \qquad \forall \upsilon_j \in V, \upsilon_i \in I \tag{6.4}$$

$$\sum_{\upsilon_i \in I} \varpi_{(i,j)}.h(i,j) \leq R_{max}, \qquad \forall \upsilon_j \in V \tag{6.5}$$

$$\sum_{\upsilon_j \in I} \varpi_{(i,j)}.T_l(\upsilon_j) \leq DTR_{passerelle}, \qquad \forall \upsilon_i \in I \tag{6.6}$$

$$\sum_{\upsilon_l \in V, e=(j,k)} f^i_{l,e}.\lambda^i_{j,k} + T_l(\upsilon_k) \leq DTR_k, \qquad \forall \upsilon_k \in V, \forall \upsilon_i \in I \tag{6.7}$$

$$\sum_{\upsilon_l \in V, e=(j,k)} f^i_{l,e}.\lambda^i_{j,k} + T_l(\upsilon_k) = \sum_{\upsilon_l \in V, e=(k,p)} f^i_{l,e}.\lambda^i_{k,p}, \qquad \forall \upsilon_k \in V, \forall \upsilon_i \in I \tag{6.8}$$

$$\sum_{e=(j,k)} f^i_{j,e}.\lambda^i_{j,k} = \sum_{e=(l,i)} f^i_{j,e}.\lambda^i_{l,i} = T_l(\upsilon_j), \qquad \forall \upsilon_j \in V, \upsilon_i \in I \tag{6.9}$$

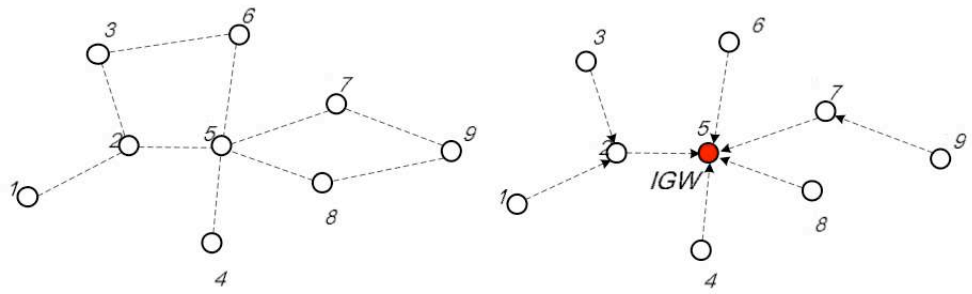
$$\chi_i \in \{0,1\}, \qquad \forall \upsilon_i \in V$$
$$\varpi_{(i,j)} \in \{0,1\}, \qquad \forall \upsilon_i \in V, \forall \upsilon_j \in V$$

Nous donnons, dans la suite, la signification de ces équations :
(6.1) : Le premier objectif est de minimiser le nombre des passerelles.
(6.2) : Le deuxième objectif est de minimiser le nombre de sauts entre tous les routeurs et leurs passerelles.
(6.3) : Un routeur est attaché à une seule passerelle.
(6.4) : Le noeud u_i auquel est attaché le routeur u_j est attaché, est une passerelle.
(6.5) : Le nombre de sauts entre un routeur et sa passerelle est inférieur à R_{max}.
(6.6) : La passerelle doit satisfaire la demande en termes de trafic.
(6.7) : Un routeur u_k doit satisfaire son trafic local et le trafic qui l'utilise comme relais.
(6.8) : La conservation du flux au niveau du routeur u_k.
(6.9) : Conservation des flux : le flux généré par tous les routeurs est le même qui entre dans les passerelles.

Le modèle mathématique présenté ci-dessus est un modèle général prenant en compte l'ensemble des contraintes et objectifs impliqués dans une telle problématique. Mais, un problème d'optimisation non-linéaire en nombres entiers est un problème NP-difficile ; donc il ne peut être résolu qu'en un temps exponentiel [37]. Plusieurs algorithmes ont été proposées pour trouver une solution dans un temps polynomial. Une première approche consiste à exploiter des algorithmes issus de la théorie de graphes et une deuxième à ramener le *Gateway Placement Problem* à un problème de localisation des installations dans les systèmes de distribution. Nous exposons dans les deux sections suivantes, deux algorithmes adoptant chacun une approche différente.

6.1.2 D-GCP et W-GCP

B. He [107] a développé deux algorithmes basées sur GDTSP (*Greedy Dominating Tree Set Partitioning*). Dans son approche, le graphe initial est divisé en plusieurs arbres enracinés disjoints. Un arbre enraciné (*rooted tree*) est un graphe orienté où tous les liens sont orientés vers un noeud déterminé, qui est dans ce contexte une passerelle. Cette architecture présente plusieurs avantages tels que la réduction de l'*overhead* de routage et l'agrégation efficace des flux. La figure 6.2 montre un exemple de construction d'un arbre enraciné (le noeud υ_5 est la passerelle).
On va décrire dans la suite les étapes de sélection des passerelles, illustrées dans la figure 6.3.



(a) Topologie initiale du réseau (b) L'arbre enraciné au noeud v_5

FIG. 6.2 – Construction d'un arbre enraciné.

La première étape consiste à collecter des informations sur le nombre des noeuds, leurs positions et leurs relations de voisinage. Grâce à ces informations, l'algorithme génère une matrice de voisinage à un seul saut qui va être utilisée, dans la suite pour créer un graphe connecté à R sauts. On définit R comme le nombre maximal de sauts entre un routeur et sa passerelle.

6.1.2.1 Formation du graphe connecté à R sauts

Un graphe résiduel $G_{residual} = (V', E')$ est un graphe qui exclut les noeuds qui sont déjà attribués à un arbre enraciné donné.
Étant donné le graphe initial $G = (V, E)$ ou le graphe résiduel $G_{residual} = (V', E')$, cette étape consiste à générer un graphe connecté à R sauts ($G_R = (V_R, E_R)$). Dans ce graphe, deux noeuds sont considérés comme voisins si la distance entre eux, en nombre de sauts, est inférieure ou égale à R.

6.1.2.2 Sélection de la passerelle Internet

Dans cette étape, une passerelle est choisie à partir du graphe résiduel $G_R = (V_R, E_R)$. Deux critères de choix sont proposés : le degré de connectivité et le poids du noeud (*weight*), correspondant aux deux algorithmes D-GCP[1] et W-GCP[2], respectivement.

[1] *Degree based Greedy Cluster Partitioning*
[2] *Weight based Greedy Cluster Partitioning*

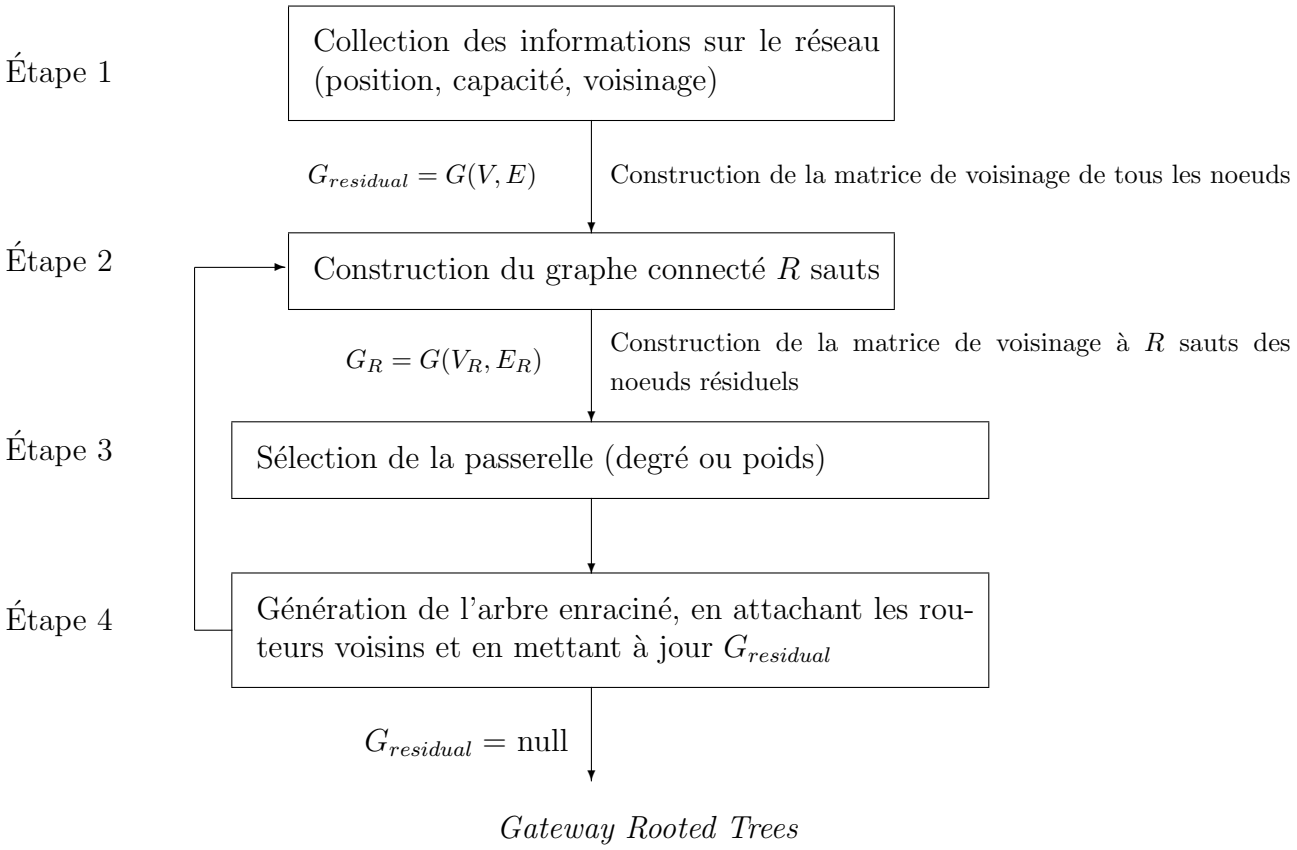


FIG. 6.3 – Les étapes de l'algorithme du placement des passerelles.

6.1.2.2.1 Choix basé sur le degré La passerelle est choisie en se basant sur son degré de connectivité à R sauts. Ce dernier est égal au nombre total de voisins dans le graphe $G_R = (V_R, E_R)$. L'algorithme sélectionne le noeud qui a le plus grand degré de connectivité.

6.1.2.2.2 Choix basé sur le poids Le poids de chaque noeud υ_i est calculé de la manière suivante :

$$W(\upsilon_i, R) = \sum_{\forall \upsilon_j \in N_R(\upsilon_i)} \frac{1}{hop(\upsilon_i, \upsilon_j)},$$

où $hop(\upsilon_i, \upsilon_j)$ est la longueur du plus court chemin entre les noeuds υ_i et υ_j et $N_R(\upsilon_i)$ est l'ensemble des voisins de υ_i dans le graphe G_R. Le noeud qui a la plus grande valeur de $W(\upsilon_i, R)$ est choisi comme passerelle.

Le choix basé sur le degré de connectivité traite tous les voisins à R sauts de la même manière, alors que le choix basé sur le poids fait la différence entre un voisin situé à un seul saut et un autre situé à deux sauts. Pour illustrer la différence entre ces deux critères de choix, on prend l'exemple de la figure 6.2 (a) et on suppose que $R = 3$. Les noeuds υ_2 et υ_5 ont le même degré de connectivité à 3 sauts, égal à 8. En revanche $W(\upsilon_2, 3) = 3 \times \frac{1}{1} + 4 \times \frac{1}{2} + \frac{1}{3} = 5.3$ et $W(\upsilon_5, 3) = 5 \times \frac{1}{1} + 3 \times \frac{1}{2} = 6.5$. Le noeud υ_5 est alors sélectionné comme passerelle car $W(\upsilon_5, 3) > W(\upsilon_2, 3)$.

6.1.2.3 Génération de l'arbre enraciné

Après le choix de la passerelle, un arbre enraciné est généré en sélectionnant les routeurs subordonnés. L'algorithme de parcours en largeur (*Breadth First Search*) est utilisé pour sélectionner les routeurs situés dans le voisinage à R sauts de la passerelle. Donc les routeurs les plus proches sont attribués à cette passerelle avant les plus éloignés.
Étant donné que le choix d'une passerelle est soumis à des contraintes de capacité, l'affiliation d'un routeur doit satisfaire les conditions imposées sur la capacité des passerelles et des routeurs.

Chaque noeud sélectionné comme passerelle ou attaché à une passerelle sera supprimé du graphe du réseau et le graphe $G_{residual}$ sera mis à jour. Les étapes décrites précédemment sont répétées jusqu'à ce que $G_{residual}$ ne contienne plus de noeuds ; c'est-à-dire un noeud du graphe initial est soit sélectionné comme passerelle, soit attaché à une passerelle.

Pour conclure, D-GCP et W-GCP sont deux algorithmes qui exploitent le concept d'arbres enracinés pour résoudre le problème d'emplacement des passerelles dans les réseaux *mesh*. D'autres concepts issus de la théorie du graphe ont été utilisés pour résoudre le *Gateway Placement Problem*. Dans [38], le choix des passerelles repose sur la recherche de l'ensemble dominant minimal (*Minimum Dominating Set*). LPGPA[3] [108] construit l'ensemble dominant minimal de poids maximal (*Minimum Dominating Set with Maximal Weight*). Dans les deux cas, un noeud appartenant à l'ensemble dominant est configuré comme passerelle.

6.1.3 MinHopCount et MinContention

Le choix de l'emplacement des passerelles Internet dans un réseau *mesh* partage des points communs avec la localisation des installations dans les systèmes de distribution. Dans les deux cas, le but est de minimiser les coûts d'acquisition des ressources tout en répondant aux besoins des utilisateurs. Ce rapprochement entre les deux problématiques est le point

[3] *Load Balanced Gateway Placement Algorithm*

de départ dans la conception des deux algorithmes *MinHopCount* et *MinContention* proposés par Robinson [37].

6.1.3.1 MinHopCount

Dans cet algorithme, le principal objectif est de minimiser le nombre de sauts entre les routeurs et les passerelles. De ce point de vue, le *Gateway Placement Problem* a un lien fort avec une problématique de la Recherche Opérationnelle : le *Facility Location Problem*. L'objectif de cette problématique est de déterminer un sous-ensemble de sites à ouvrir pour servir de dépôts, de manière à minimiser les distances à parcourir pour atteindre le dépôt le plus proche, tout en minimisant les coûts d'installation. Les sites ouverts doivent satisfaire l'ensemble de la demande.

On note :

* V représente l'ensemble des routeurs du réseau (passerelle ou pas).
* n est le nombre de routeurs.
* v est un routeur ; $v \in V$.
* Ω est un sous-ensemble de routeurs ($\Omega \subset V$).
* I représente l'ensemble des passerelles.
* Γ est un vecteur tel que $\Gamma[i] = 1$, si $i \in I$.
* $c[v]$ est la capacité du routeur v en absence de toute contention.
* $c'[v]$ est la capacité effectivement disponible du routeur v. Elle dépend de la matrice de routage de tout le réseau, donc la localisation des passerelles.

MinHopCount est un algorithme de recherche locale qui part d'une solution initiale et l'améliore en explorant de manière itérative, son voisinage immédiat. L'algorithme consiste en trois opérations :

1. *add(v)* évalue le coût d'installer une passerelle au noeud v.
2. *open(v, Ω)* installe une passerelle au noeud v et retire les passerelles de l'ensemble Ω, en réaffectant les noeuds servis par Ω à v.
3. *close(v, Ω)* retire la passerelle installée à v et installe des passerelles dans tous les noeuds de l'ensemble Ω, en réaffectant les noeuds servis par v à Ω.

A partir d'une solution initiale du problème, l'algorithme exécute les trois opérations, *add()*, *open()* et *close()*, pour améliorer la qualité de la solution obtenue. Afin de garantir un temps d'exécution polynomial, chaque étape doit réduire le coût total de la solution par un facteur minimal égal à p. La fonction coût dans *MinHopCount* est le nombre de

sauts entre les routeurs et les passerelles. La table 6.1 donne un aperçu des étapes de l'algorithme *MinHopCount*.

Pseudo-code de *MinHopCount*

Initialiser le vecteur de capacité **c**.
Faire {
  Commencer par une solution valide Γ et calculer son coût $F(\Gamma)$.
  Faire {
    Pour tout $\upsilon \in V$:
      Trouver une opération *add(υ)* valide.
      Trouver une opération *open(υ, Ω)* valide : Ω est une solution au problème du sac à dos (*knapsack*) avec la taille du sac à dos égale à $c[\upsilon]$.
      Trouver une opération *close(υ, Ω)* valide.
    Calculer Δ coût pour toute opération valide.
    Appliquer à Γ l'opération qui fournit le meilleur Δ coût.
  } **tant que** (Δ coût $\geq F(\Gamma)/p$).
  Γ est un solution optimale locale.

  Calculer le vecteur de capacité **c'**.
  Si c' $<$ $\mathbf{c}_{precedent}$, **alors** mettre à jour le vecteur $\mathbf{c}_{actuel}$.
} **tant que** $((\sum_{i=1}^{n} \mathbf{c}_{precedent}[i]$ - $\mathbf{c}_{actuel}[i]\ ) \geq \alpha\ )$.

Retourner Γ.

TAB. 6.1 – Pseudo-code de *MinHopCount*.

6.1.3.2 MinContention

Le problème cette fois, est de minimiser la contention dans tous les noeuds du réseau, ce qui peut être assimilé au problème des k-médians [109].
Le problème des k-médians est issu de la théorie de la localisation discrète où on peut installer uniquement k ressources. Une résolution du problème consiste à échanger p ($p < k$) ressources 'installées' (Ψ) contre p autres ressources (Ω). Cette opération est notée *swap*(Ψ,Ω).

Dans *MinContention*, le coût d'une configuration donnée est la somme des poids des liens actifs. Le poids d'un lien actif est le nombre total de routeurs qui sont dans sa zone de

contention. Les passerelles seront donc placées près des routeurs avec plus de demande en termes de trafic vers le monde IP extérieur. La table 6.2 donne un aperçu des étapes de l'algorithme *MinContention*.

Pseudo-code de *MinContention*
Trouver une solution valide.
Faire {
Trouver toutes les opérations *swap*(Ψ,Ω) valides.
où Ψ est l'ensemble des p passerelles à ouvrir.
et Ω est l'ensemble des p passerelles à fermer.
Calculer le Δ coût de chaque opération.
Appliquer l'opération *swap* avec le plus large Δ coût positif.
} **tant que** (Δ coût $\geq F(\Gamma)/p$)
Retourner

TAB. 6.2 – Pseudo-code de *MinContention*.

Pour conclure, outre la prise en compte de la contention dans le réseau, l'originalité du travail de J. Robinson décrit ci-dessus provient de la validation des deux algorithmes, *MinHopCount* et *MinContention* dans des scénarios réels, tels que le réseau GoogleWiFi[4].

Le *Satellite Capability Placement Problem* partage plusieurs points communs avec le *Gateway Placement Problem*. Dans les deux cas, le but est de trouver un compromis entre la minimisation du nombre de passerelles et le respect de certains critères de performance tels que les délais de transmission et la charge des noeuds. Pour résoudre le *Gateway Placement Problem*, D-GCP et W-GCP exploitent les algorithmes de la théorie des graphes et *MinHopCount* et *MinContention* ramènent le problème initial à un problème de localisation des installations dans un système de distribution.

Les algorithmes *MinHopCount* et *MinContention* ont ramené le problème du choix des passerelles à un autre problème bien connu qui est le *facility location*. De la même manière, nous proposons donc de ramener le *Satellite Capacity Location Problem* à un *Discrete Network Design Problem* qui est un problème de mise à niveau des réseaux de transport. Dans la section suivante, nous allons montrer la corrélation entre les deux problèmes (le

[4]http ://wifi.google.com

Satellite Capacity Location Problem et le *Discrete Network Design Problem*) et ensuite comparer les performances de la solution proposée aux approches basées sur la théorie des graphes tels que W-GCP.

6.2 Discrete Network Design Problem

Le *Discrete Network Design Problem* (DNDP) concerne la mise à niveau des réseaux de transports par l'ajout de nouvelles artères. L'enjeu décisionnel de la problématique est de déterminer quelles voies de transport ouvrir. Si on assimile le réseau de transport à un réseau de télécommunications, le DNDP a un lien fort avec la problématique posée dans ce chapitre. Le nombre de nouvelles routes doit être suffisamment petit pour respecter la limite budgétaire et suffisamment grand pour satisfaire la demande des usagers. Par analogie, le nombre de passerelles doit être suffisamment petit pour minimiser les coûts d'installation et suffisamment grand pour satisfaire la qualité de service.

Le *Discrete Network Design Problem* fait intervenir deux niveaux de décision : le premier niveau est formé par le planificateur appelé meneur ou *leader*. Le deuxième niveau est formé par les usagers de la route appelés suiveurs ou *followers*. Le meneur décide des routes qui seront construites de façon à optimiser le cumul des temps de parcours de tous les usagers. Quant au suiveur, il réagit aux décisions prises par le meneur de façon à optimiser son propre temps de parcours. La fonction objectif du meneur est donc déterminée en partie par le comportement des suiveurs.

Pour modéliser l'interaction hiérarchique meneur-suiveur, le DNDP peur être formulé sous la forme d'un problème à deux niveaux (*Bilevel Programming*).

6.2.1 Bilevel Programming

Le *Bilevel Programming* s'intéresse aux problèmes d'optimisation, où deux agents entretiennent une relation hiérarchique. Les décisions du premier agent influencent les politiques de décision du deuxième et vice versa. Plus précisément, le meneur choisit une stratégie y dans un ensemble $Y \in \mathbb{R}^n$ et les suiveurs possèdent un ensemble des stratégies $X(y) \in \mathbb{R}^m$ correspondant à chaque $y \in Y$. On a donc, Y est le vecteur de design et X est le vecteur de réponse. L'objectif est de minimiser certaines fonctions $f, \varphi : \mathbb{R}^n \times \mathbb{R}^m \longrightarrow \mathbb{R}$.

De manière générale, le problème est décrit par le modèle mathématique suivant : [110]

$$\min_y f(x, y),$$

$$\text{avec } (x, y) \in Z,$$

$$\min_{x \in X(y)} \varphi(x, y),$$

où $Z \subseteq \mathbb{R}^n \times \mathbb{R}^m$ est l'ensemble des décisions et de réponses possibles.
Dans le contexte d'optimisation d'un réseau de transports, $\min_y f(x, y)$ décrit l'objectif du meneur, Y le vecteur des décisions prises par ce dernier, $\min_{x \in X(y)} \varphi(x, y)$ le problème d'affectation de trafic et X le vecteur des flux à l'équilibre.

L'affectation de trafic consiste à répartir entre les différents itinéraires possibles l'ensemble de flux des usagers désirant se rendre d'une origine à une destination. Elle simule, face à une modification du réseau (nouvelle route, restriction locale, etc.), la réaction des utilisateurs qui tentent de minimiser leur temps de parcours.

Les variables de décision du niveau supérieur peuvent être discrètes ou continues en fonction du problème à résoudre. Les problèmes à deux niveaux sont intrinsèquement difficiles. Les recherches se sont focalisées sur les problèmes qui ont de bonnes propriétés telles que la continuité des variables. Dans ce cas, plusieurs méthodes ont été proposées : *Complementary Pivoting* par Bialas en 1980 [111], *Descent Methods* par Kolstad en 1990 [112] et *Penalty Function Methods* par Ishizuka en 1992 [113]. Le DNDP est un problème d'optimisation combinatoire où les composantes du vecteur Y prennent des valeurs dans l'ensemble $\{0, 1\}$. La discrétisation des variables rend le problème plus difficile à résoudre.
On s'intéresse maintenant à la reformulation mathématique du DNDP.

6.2.2 Modèle mathématique du DNDP

Notations utilisées :

* Y est le vecteur décisions, où y_{ij} est égal à 1 si le lien (i, j) est ajouté au réseau, 0 sinon.
* E_y est l'ensemble des liens candidats.
* E_{y_1} est l'ensemble des liens ajoutés : $E_{y_1} = \{(i, j) \in E_y; y_{ij} = 1\}$.
* X est le vecteur de réponses, où $x_{ij}(y)$ est le flux du lien (i, j).
* $t_{ij}(x_{ij}(y))$ est le temps de parcours du lien (i, j).
* RS définit l'ensemble de toutes les paires origine-destination.
* f_p^{rs} est le flux du chemin p entre l'origine r et la destination s.
* P_{rs} est l'ensemble de tous les chemins entre l'origine r et la destination s.

* d_{rs} est demande de trafic entre l'origine r et la destination s.
* $\zeta_{ij,p}^{rs}$ est une variable binaire, égale à 1 si le chemin p entre l'origine r et la destination s contient le lien (i,j), 0 sinon.
* c_{ij} est le coût de construction du lien (i, j).

Le DNDP est modélisé par [114] :

$$\min_{y} f(x, y) = \sum_{(i,j)\in E\cup E_{y_1}} t_{ij}(x_{ij}^*(y))x_{ij}^*(y) \tag{6.10}$$

avec

$$y_{ij} \in \{0, 1\}, (i, j) \in E_y \tag{6.11}$$

$$P(t_{rs} \leq t_m) > R, \forall rs \in RS \tag{6.12}$$

$$\sum_{(i,j)} \in E_{y_1} c_{ij} y_{ij} < B, \tag{6.13}$$

où x^* est la solution au problème d'affectation de trafic suivant. Pour tout vecteur Y :

$$\min_{x} \sum_{(i,j)\in E\cup E_{y_1}} \int_0^{x_{ij}} t_{ij}(u)du \tag{6.14}$$

avec

$$\sum_{p\in P_{rs}} f_p^{rs} = d_{rs}, \forall rs \in RS \tag{6.15}$$

$$x_{ij} = \sum_{rs\in RS} \sum_{p\in P_{rs}} f_p^{rs} \zeta_{ij,p}^{rs}, \forall (i, j) \in E \cup E_{y_1} \tag{6.16}$$

Le niveau supérieur, illustré par les équations 6.10 – 6.13 représente les objectifs et les contraintes du meneur. Le but, de point de vue système, est de minimiser le temps total de parcours (6.10). Le meneur doit satisfaire les contraintes de fiabilité exprimées par l'inégalité 6.12. La fiabilité est la probabilité que le temps de parcours entre une origine et une destination données soit inférieur à un certain seuil t_m. Finalement, l'inégalité 6.13 représente les contraintes budgétaires. Les variables R et B représentent respectivement, la borne inférieure de la fiabilité et la borne supérieure du budget.

Le niveau inférieur, illustré par les équations 6.14 – 6.16 décrit un modèle d'affectation de trafic. Les utilisateurs réagissent aux modifications décidées par le meneur en choisissant les routes qui minimisent leurs temps de parcours. Dans un modèle où le temps de parcours dépend du flux, l'équilibre se calcule par approximations successives, à l'aide d'un algorithme itératif. Schittenhelm [115] propose de considérer chaque origine et d'effectuer au début, une recherche de plus court chemin entre cette origine et toutes les destinations. Ensuite, il applique le principe d'égalisation qui consiste à transférer du flux depuis le chemin chargé vers le chemin de coût minimal jusqu'à ce qu'il y ait égalisation du temps de parcours entre les deux chemins. En répétant cette opération sur tous les chemins deux à deux, il parvient à minimiser et égaliser les temps de parcours des chemins utilisés entre l'origine et la destination. Le système ainsi obtenu décrit un équilibre de Wardrop [116], où les temps de parcours des chemins utilisés sont minimaux.

Nous avons présenté dans cette section, le modèle mathématique du DNDP. Ce modèle sera utile dans la suite pour montrer l'analogie entre le DNDP et le SCPP. Pour établir cette analogie, nous avons besoin aussi de présenter le modèle mathématique du SCPP. Dans la section suivante, nous allons exposer le modèle mathématique du SCPP et grâce au modèle du DNDP présenté précédemment, nous allons démontrer l'équivalence entre les deux problèmes. Cette démonstration est une étape nécessaire pour justifier le choix de l'algorithme de résolution. Finalement, nous allons adapter l'algorithme qui a été proposé pour résoudre le DNDP pour la résolution du SCPP.

6.3 Résolution du SCPP

Dans ce travail, nous adoptons les conditions sur la qualité de service proposées par Aoun [38]. La qualité de service dans le réseau peut être traduite par deux conditions. La première condition porte sur la longueur du chemin entre un routeur et la passerelle à laquelle il est affilié. En effet, les délais de transmission et la probabilité de perte de messages sont proportionnels à la longueur du chemin parcouru. La deuxième condition porte sur la charge des noeuds. Un noeud a une capacité maximale de transmission ; au-delà de cette capacité, le noeud se transforme en un goulot d'étranglement augmentant ainsi la probabilité de collision et d'abandon des messages.

6.3.1 Modèle mathématique du SCPP

Les SCPP peut être, à l'instar du DNDP modélisé sous la forme d'un problème à deux niveaux (*Bilevel Programming*). Le meneur est le planificateur du réseau et les suiveurs

sont les routeurs qui cherchent à diriger leur trafic vers la passerelle la plus proche. Dans le modèle mathématique, on utilise les notations suivantes :

* n est le nombre total de routeurs.
* Y est le vecteur de décision, y_i est égal à 1 si le routeur i est configuré comme passerelle, 0 sinon.
* I est l'ensemble des passerelles.
* $L(i)$ est la charge du noeud i.
* l_i^{rs} est le volume de trafic généré par le routeur r à destination de la passerelle s et qui passe par le routeur i.
* $h(r, s)$ est la longueur du plus court chemin entre la source r et la destination s.

On modélise la topologie du réseau par un graphe $G = (V, E)$, où $V(G)$ représente l'ensemble des routeurs du réseau et $E(G)$ correspond à l'ensemble des liens. Le SCPP est modélisé par :

$$\min \sum_{i=0}^{n} y_i \tag{6.17}$$

avec

$$h(r, s) \leq R_{max}, \quad \forall r \in V(G), \forall s \in I \tag{6.18}$$

$$L(i) \leq L_{max}, \quad \forall i \in V(G) \tag{6.19}$$

$$y_i \in \{0, 1\}, \quad \forall i \in V(G) \tag{6.20}$$

$L(i)$ et $h(r, s)$ sont les résultats au problème d'affectation suivant. Pour tout vecteur Y :

$$\min_{r \in V, s \in I} h(r, s) \tag{6.21}$$

$$L(i) = \sum_{r \in V, s \in I} l_i^{rs} \tag{6.22}$$

L'équation 6.17 montre que le but du meneur est de minimiser le nombre de passerelles dans le réseau. L'inégalité 6.18 donne la borne supérieure (R_{max}) de la longueur du chemin

entre un routeur r et sa passerelle s. Finalement, l'inégalité 6.19 donne la borne supérieure (L_{max}) de la charge de tout noeud i du réseau.

Les deux équations 6.21 et 6.22 modélisent le comportement des suiveurs. La première équation montre que chaque suiveur cherche la passerelle la plus proche pour lui diriger son trafic destiné au monde extérieur. La dernière équation montre que la charge d'un noeud dépend du schéma d'affectation d'un routeur à une passerelle.

6.3.2 Équivalence entre DNDP et SCPP

Le DNDP est un problème d'optimisation où on considère l'effet de l'ajout de nouveaux liens sur la performance du réseau de transports. La discrétisation des variables du problème et l'interaction hiérarchique entre deux niveaux (meneur-suiveur) sont deux aspects qui nous ont conduit à reprendre cette problématique et l'adapter à notre problème qui est l'effet des positions des passerelles sur les performances du réseau de télécommunications.

Dans le modèle à deux niveaux du DNDP, le meneur tente de minimiser le cumul des temps de parcours sur toutes les artères. Il est soumis à des limites budgétaires et des contraintes de fiabilité sur les temps de parcours. Dans le réseau de télécommunications, le meneur tente de minimiser le nombre de passerelles. Il est mené à respecter les conditions sur la qualité de service (QoS) : la charge des noeuds et le nombre de sauts entre les routeurs et les passerelles. Le suiveur dans DNDP tente de minimiser son propre ton temps de parcours entre une origine r et une destination s données. Son temps de parcours dépend des routes disponibles entre r et s. Le suiveur de SCPP a pour but de minimiser les délais de transferts en prenant le chemin le plus court vers la passerelle. La longueur des chemins parcourus dépend du nombre de passerelles installées et de leurs positions.

La table 6.3 illustre l'équivalence entre DNDP et SCPP.

		DNDP	**SCPP**
Meneur	Objectifs	min cumul temps de parcours	min nombre passerelles
	Contraintes	fiabilité et budget	QoS
Suiveur	Objectifs	min temps de parcours	min délais de transferts
	Contraintes	routes	passerelles

TAB. 6.3 – Équivalence entre DNDP et SCPP.

6.3.3 Algorithme Génétique

Pour résoudre le DNDP, plusieurs algorithmes ont été proposés : la méthode de *branch-and-bound* [117], la relaxation de Lagrange, la procédure de double ascension (*dual ascent*) [118] et le concept de la fonction de soutien (*support function*) [119]. Les méthodes proposées sont fortement liées aux fonctions modélisant le meneur et les suiveurs, correspondant aux fonctions $\min_{y} f(x,y)$ et $\min_{x \in X(y)} \varphi(x,y)$ du modèle *Bilevel Programming* du DNDP (section 6.2.1). D'où la nécessité d'autres approches qui peuvent être décrites de manière abstraite sans faire appel à un problème spécifique. Les métaheuristiques sont une famille de procédures bénéficiant d'une certaine intelligence pour explorer l'espace de recherche efficacement afin de déterminer des solutions *presque* optimales [120].
L'algorithme génétique (GA) a été proposé pour résoudre le DNDP par Dimitriou en 2007 [114] et L. Xu [121] en 2008. Son avantage majeur est sa capacité à traiter les problèmes sans avoir besoin d'informations sur la nature des variables utilisées. L'algorithme génétique est basé sur des mécanismes dérivés de l'évolution naturelle, où seuls les meilleurs individus subsistent et reproduisent de nouveaux individus. Il a été proposé en 1962 par Holland [122].
Nous allons décrire dans la suite les différentes étapes de l'algorithme génétique [123].

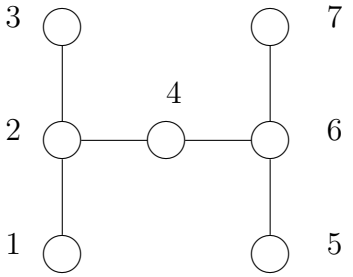
6.3.3.1 Codage

Un noeud est représenté par un gène prenant une valeur égale à 1 si le noeud correspondant est choisi comme passerelle, 0 sinon. Un individu est représenté par un chromosome qui est une collection de gènes. Finalement, une population est un ensemble d'individus.

On prend l'exemple de la figure 6.4(a). Il existe sept noeuds candidats, donc un individu est un chromosome à sept gènes. Le chromosome de la figure 6.4(c) est un exemple de solution : la passerelle vers le satellite est installée au niveau du noeud 4.

6.3.3.2 Fonction d'évaluation

Pour calculer le coût d'un point de l'espace de recherche, on utilise une fonction d'évaluation. Le résultat fourni par la fonction d'évaluation, appelé $fitness$ détermine la qualité de l'individu et son adéquation au problème. Il ne dépend pas de ceux des autres individus de la population. Le but est de ne garder que les individus les plus performants et éliminer les autres.
Le but du SCPP est de minimiser le nombre de passerelles dans le réseau. Nous proposons de définir la $fitness$ d'un individu par le nombre de passerelles satellite dans le réseau. Dans l'exemple précédent, la *fitness* de l'individu P est à 1 : un seul noeud est configuré



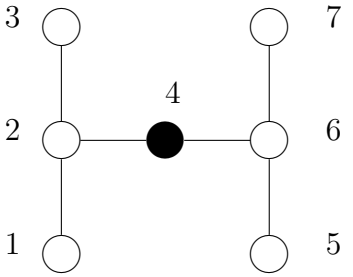
(a) Exemple de topologie

1	2	3	4	5	6	7

(b) Codage du chromosome

0	0	0	1	0	0	0

(c) Exemple de solution : P



(d) Le noeud 4 est une passerelle

FIG. 6.4 – Exemple de codage.

comme passerelle.

La fonction d'évaluation permet de sélectionner ou de refuser un individu. Dans le SCPP, la solution proposée doit satisfaire les contraintes sur la qualité de service : le nombre de sauts entre un routeur et une passerelle doit être inférieur à R_{max} et la charge d'un noeud à L_{max}. Dans l'exemple précédent pour atteindre la passerelle, on doit parcourir au maximum deux sauts. Si le nombre maximal de sauts autorisé était égal à un ($R_{max} = 1$), l'individu P qui place une passerelle au noeud 4 ne représente pas une solution valide du problème. De même, pour la condition sur la charge des noeuds. Pour le calcul de la charge, on suppose que chaque noeud a une demande égale à une unité de trafic [107]. Cette unité peut être exprimée en bits par second. Dans l'exemple précédent, les noeuds les plus chargés sont les noeuds 2 et 6 avec une charge égale à trois unités de trafic [5]. Si La capacité d'un noeud était égale à deux, l'individu P qui place une passerelle au noeud 4 ne représente pas une solution valide du problème.

Le codage et la fonction d'évaluation étant choisis, l'algorithme génétique se déroule en plusieurs étapes. En partant d'une population initiale donnée, un cycle d'opérations d'affectation de trafic, de sélection, de croisement et de mutation est répété jusqu'à ce que le critère d'arrêt soit atteint. Ces opérations servent à renouveler et améliorer la population. Le déroulement de l'algorithme génétique est illustré par l'ordinogramme 6.5.

[5]Par exemple pour le noeud 2 : les noeuds 1 et 3 passent par 2 pour communiquer avec 4. Ceci représente une charge égale à deux unités de trafic. Si on y ajoute le trafic généré par 2, on obtient une charge totale à trois unités de trafic.

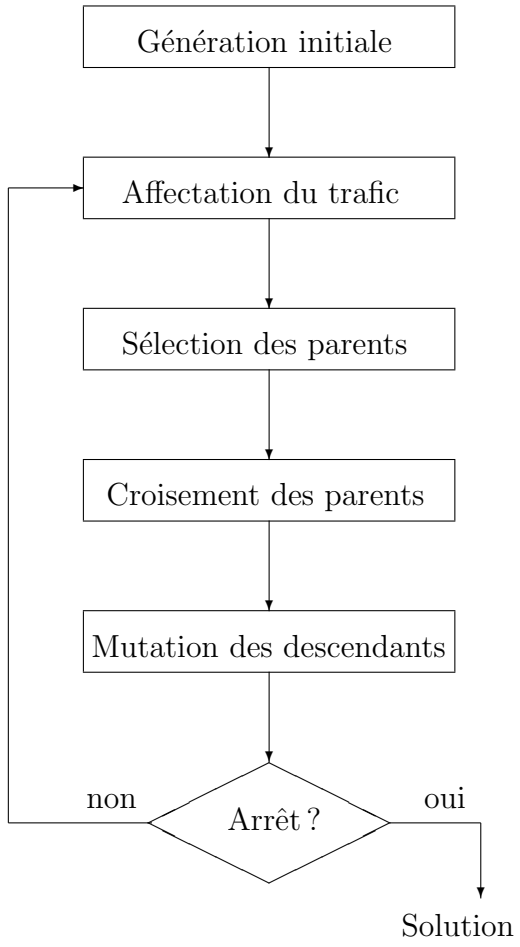


FIG. 6.5 – Ordinogramme du déroulement de l'algorithme génétique.

6.3.3.3 Initialisation

Les paramètres d'initialisation de l'algorithme génétique :

* $N_{generations}$ est le nombre de générations.
* $N_{population}$ est le nombre d'individus dans une population.
* p_c est la probabilité de croisement.
* p_m est la probabilité de mutation.

Après l'initialisation des quatre paramètres précédents, on génère de manière aléatoire une population initiale qui servira de base pour les générations futures.

6.3.3.4 Affectation de trafic

Pour chaque individu, on a besoin de vérifier s'il respecte les conditions de qualité de service imposées. L'opération d'affectation du trafic permet notamment de vérifier les conditions sur la charge des noeuds et la longueur du chemin entre tout noeud et sa passerelle.

Le mécanisme d'affectation du trafic se produit de manière itérative. On considère tour à tour chaque noeud et on cherche depuis ce noeud les plus courts chemins vers toutes les passerelles. Le noeud est alors affecté à la passerelle la plus proche.

6.3.3.5 Sélection

La sélection permet d'identifier les meilleurs individus d'une population qui seront gardés pour former la nouvelle génération et éliminer progressivement les mauvais individus pour éviter la dégénérescence. Il existe dans la littérature plusieurs méthodes de sélection. Dans la sélection par roulette (*Roulette Wheel Selection*) [124], plus l'individu est meilleur, plus la probabilité d'être sélectionné est grande. Cette méthode rencontre des problèmes quand les écarts de $fitness$ sont importants : le meilleur individu de la population est trop souvent sélectionné et ses congénères sont éliminés. Une manière de contourner ce problème est d'ordonner les individus et d'attribuer à chaque individu une nouvelle valeur de $fitness$: la $fitness$ du plus mauvais sera égal à 1 et celle du meilleur $N_{population}$. Cette méthode, appelée sélection par rang (*Rank Selection*) [125] empêche les individus les plus forts de dominer et d'éliminer les plus faibles.

Après les deux opérations de croisement et de mutation (expliquées dans le prochain paragraphe), on risque de perdre les meilleurs chromosomes. On utilise alors la méthode d'élitisme [125] qui consiste à garder une partie des meilleurs chromosomes dans la nouvelle génération. Cette méthode améliore considérablement l'algorithme génétique, car elle permet de ne pas perdre les meilleures solutions. Pour la résolution du SCPP, on utilise cette dernière méthode, car le codage du problème est très sensible au croisement et la mutation.

6.3.3.6 Opérateurs de croisement et de mutation

Les opérateurs permettent de diversifier la population au cours des générations et explorer l'espace des solutions candidates. Les deux opérateurs évolutionnaires utilisés sont le

croisement et la mutation.

6.3.3.6.1 Croisement L'opérateur de croisement est appliqué aux parents P1 et P2 avec une probabilité p_c et génère un couple d'enfants C1 et C2. Les enfants sont composés d'une partie de chacun de leurs parents. La probabilité de croisement p_c représente la fréquence à laquelle les croisements sont appliqués. La figure 6.6 illustre l'opérateur de croisement.

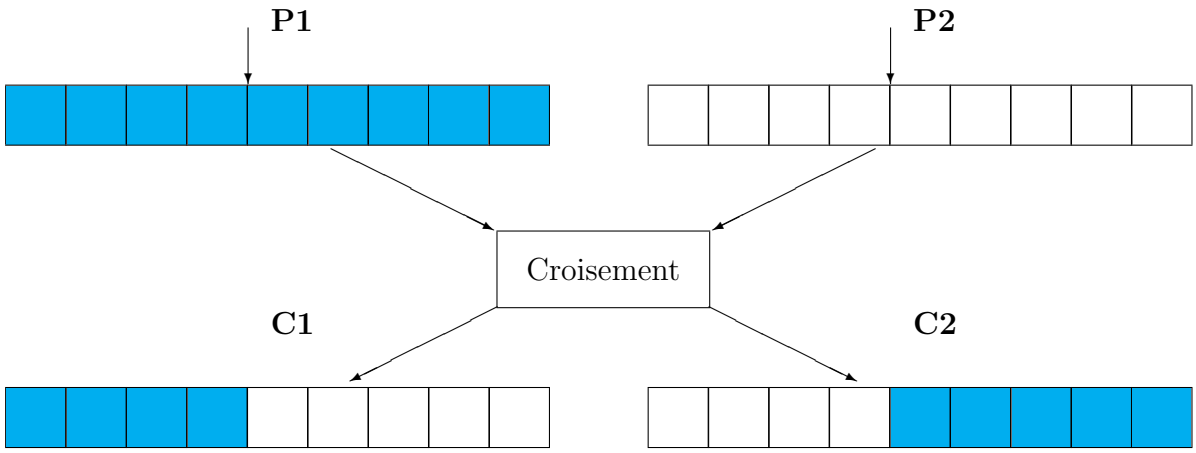


FIG. 6.6 – Croisement des parents P1 et P2 : génération des enfants C1 et C2.

6.3.3.6.2 Mutation La mutation est une source de diversité génétique. L'opérateur de mutation est appliqué à P avec la probabilité p_m et génère un individu muté P'. La probabilité de mutation représente la fréquence à laquelle les gènes d'un chromosome sont mutés. La figure 6.7 illustre l'opérateur de mutation.

6.3.3.7 Itération

A partir de la population initiale, l'algorithme génétique applique les trois opérations de sélection, croisement et mutation pour construire une nouvelle génération. A partir de cette population, on réitère le procédé $N_{generations}$ fois.

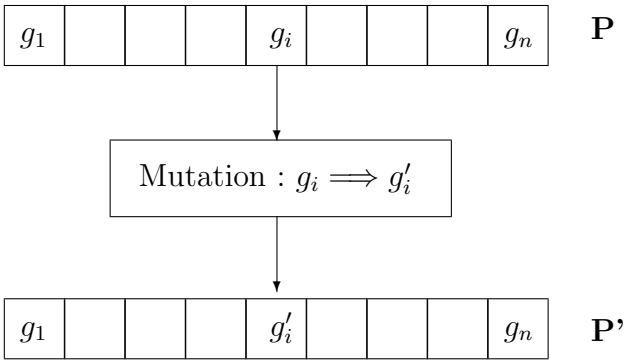
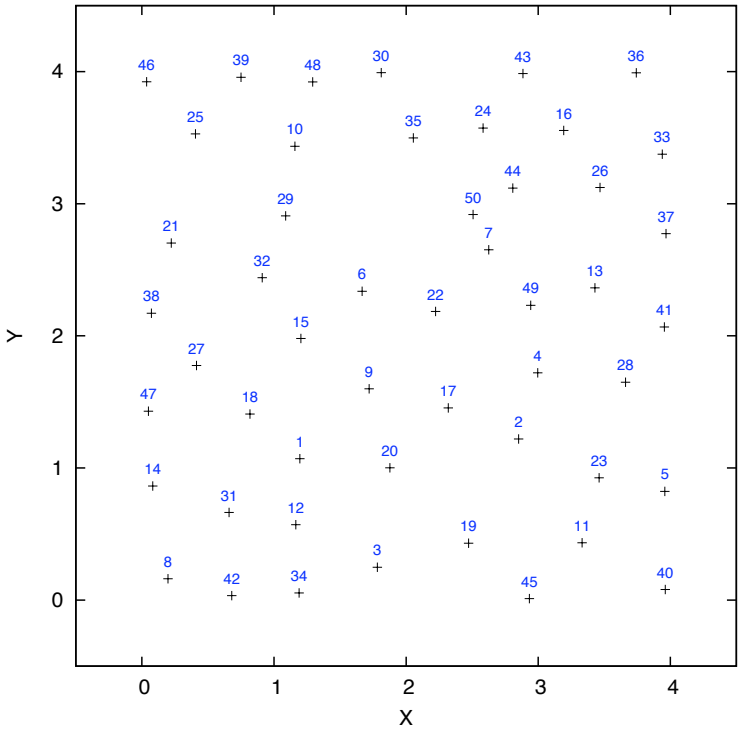


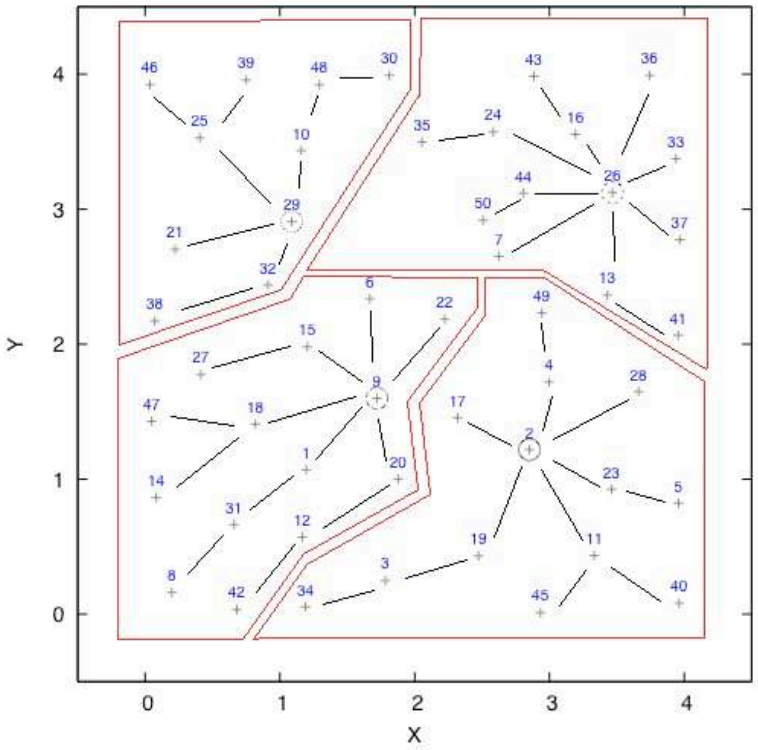
FIG. 6.7 – Mutation de l'individu P : génération de P'.

Afin d'illustrer le fonctionnement de l'algorithme génétique, nous avons pris l'exemple du réseau représenté dans la figure 6.8(a). Le réseau comporte 50 noeuds répartis sur une surface de simulation de taille 4 m × 4 m. La portée radio est fixée à 1 m. En fixant la charge maximale des routeurs à 3 unités et le nombre maximal de sauts à 3 également, l'algorithme calcule le nombre minimal de passerelles nécessaires. Quatre passerelles sont installées au niveau des noeuds 2, 9, 26 et 29 (Figure 6.8(b)). Les noeuds les plus éloignés (8, 42 et 34) sont à 3 sauts des passerelles 9 et 2, tandis que les autres noeuds sont à moins de 3 sauts de leurs passerelles. Les noeuds les plus chargés (1, 10, 11, 18, 19, 20, 25) supportent un trafic égal à 3 unités. En effet, le noeud 25 par exemple relaye le trafic provenant de ses voisins 39 et 46. En ajoutant le trafic généré par le noeud lui-même, on obtient au total 3 unités de trafic.

Nous avons établi l'équivalence entre le DNDP et le SCPP grâce à la modélisation mathématique des deux problèmes. L'algorithme génétique a été proposé pour résoudre le DNDP. Nous l'avons adapté pour la résolution du SCPP. Dans la section suivante, nous allons comparer les performances de l'algorithme génétique avec des approches basées sur la théorie des graphes. Deux approches seront utilisées pour cette approche. La première, W-GCP est proposée par B. He et al. [107] et a été exposée dans le paragraphe 6.1.2. La deuxième, *Recursive* est proposée par Aoun et al. [38]. Nous donnerons dans la section suivante un descriptif de *Recursive* et expliquerons les différences entre cette approche et W-GCP.



(a)



(b)

FIG. 6.8 – Résolution du SCPP grâce à l'algorithme génétique.

6.4 Résultats de simulation

Pour la comparaison algorithme génétique/W-GCP et la comparaison algorithme génétique/ *Recursive*, nous adoptons deux scénarios de déploiement différents. Dans le premier, les noeuds seront uniformément distribués sur la surface de simulation. Pour la seconde, les noeuds sont répartis de manière à ce que la distance minimale entre deux noeuds soit inférieure ou égale à $0.6 \times r$, où r est la portée radio. Pour les deux scénarios, on suppose que chaque noeud a une demande en trafic égale à 1 unité. Cette unité peut être exprimée en bits/s. Le tableau 6.4 illustre les paramètres de simulation dans les deux scénarios W-GCP et *Recursive* et le tableau 6.5 illustre les paramètres de simulation pour l'algorithme génétique.

Paramètre	**W-GCP**	***Recursive***
Surface de simulation ($m \times m$)	200 x 200	10 x 10
Nombre de noeuds	200	175
Portée radio (m)	30	1

TAB. 6.4 – Paramètres de simulation pour W-GCP et *Recursive*.

Paramètre	**Valeur**
$N_{generations}$	500
$N_{population}$	100
p_c	0.5
p_m	0.1

TAB. 6.5 – Paramètres de simulation pour l'algorithme génétique.

6.4.1 Comparaison de l'algorithme génétique et W-GCP

Dans le paragraphe 6.1.2, nous avons exposé deux approches W-GCP et D-GCP proposées dans le cadre du même travail [107]. Les auteurs de ces deux approches ont démontré que W-GCP réalise de meilleures performance que D-CGP. Nous allons donc comparer les performances de l'algorithme génétique à celles de W-GCP. La figure 6.9 montre le nombre de passerelles nécessaires en fonction de la capacité des routeurs. Le nombre maximum de sauts entre un routeur et une passerelle R_{max} est fixé à 4. La première remarque concerne les performances de l'algorithme génétique par rapport à W-GCP. L'algorithme génétique permet de placer moins de passerelles dans le réseau tout en respectant les conditions sur

la qualité de service. Le gain est de 4 passerelles pour une capacité égale à 4 unités et peut atteindre à peu près 6 passerelles pour une capacité égale à 20. La deuxième remarque se rapporte à l'évolution des deux courbes en fonction de la charge. Jusqu'à une charge égale à 10, les deux courbes chutent avec des pentes très similaires. Mais au delà de 10, la pente de la courbe représentant les performances de W-GCP devient très faible, tandis que le nombre de passerelles dans l'algorithme génétique continue à chuter avec le même rythme. Ce comportement peut être expliqué par la manière dont W-GCP sélectionne les passerelles. Nous rappelons que dans l'itération R, le poids $W(v_i, R)$ d'un noeud v_i est calculé comme suit :

$$W(v_i, R) = \sum_{\forall v_j \in N_R(v_i)} \frac{1}{hop(v_i, v_j)},$$

où $hop(v_i, v_j)$ est la longueur du plus court chemin entre les noeuds v_i et v_j et $N_R(v_i)$ est l'ensemble des voisins de v_i dans le graphe G_R.
Avec cette méthode de calcul, on favorise les noeuds dont les voisins sont plus regroupés. On suppose que deux noeuds v_i et v_j ont tous les deux 10 voisins à 4 sauts. Mais la moitié des voisins de v_i sont des voisins directs (à un seul saut) et v_j a uniquement 2 voisins directs et le autres sont à plus d'un saut. W-GCP sélectionne v_i plutôt que v_j. Avoir plus de noeuds à 4 sauts que de noeud à un seul saut augmente la charge des noeuds. Donc la condition sur le nombre de sauts est plus contraignante que la condition sur la capacité dans W-GCP.

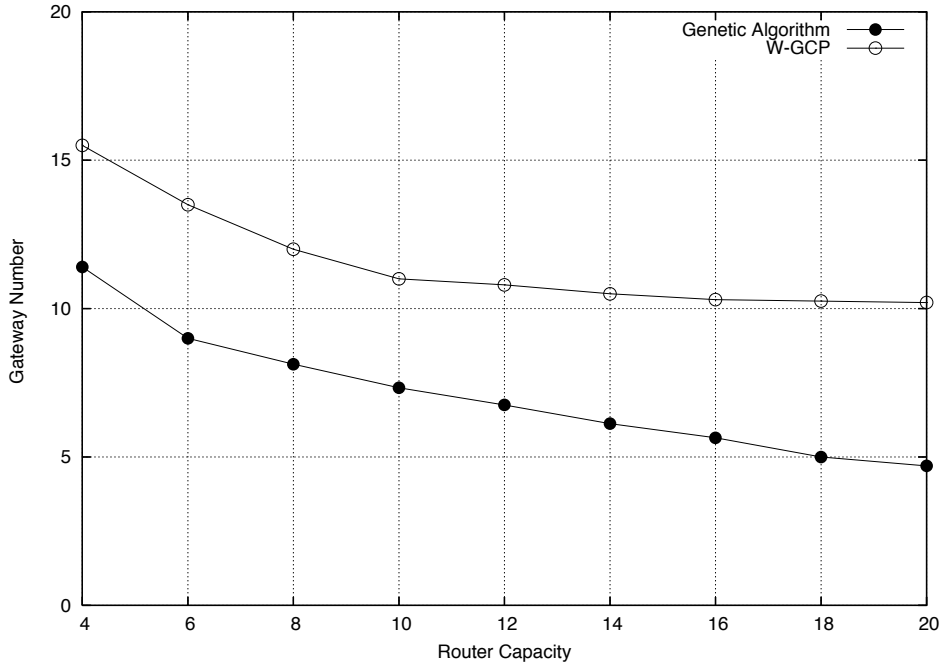


FIG. 6.9 – Nombre de passerelles nécessaires par l'algorithme génétique et W-GCP en fonction de la capacité des routeurs avec $R_{max} = 4$.

6.4.2 Comparaison de l'algorithme génétique et Recursive

L'approche proposée par Aoun [38], *Recursive* repose sur un processus itératif de sélection formé par R_{max} tours, où R_{max} est le nombre maximal de sauts entre un routeur et sa passerelle. A chaque tour i, le noeud v qui couvre le plus de routeurs est sélectionné en premier. Le nombre de sauts entre un routeur et une passerelle ne dépasse pas i. Jusqu'à ce stade, *Recursive* suit le même fonctionnement que D-GCP, basé sur le degré à i sauts. Afin de vérifier la contrainte sur la charge des noeuds, l'algorithme construit un arbre couvrant enraciné en v. Si la contrainte n'est pas satisfaite, v n'est pas sélectionné comme passerelle mais l'algorithme modifie la matrice représentant les relations de voisinage à i sauts, en supprimant une arête entre v et un de ses voisins. Dans cette matrice, deux noeuds sont considérés comme voisins si la distance entre eux est inférieure ou égale à i sauts. Le processus de sélection est répété jusqu'à tous les noeuds soient affiliés à une passerelle à i sauts. Nous parlons de i sauts car le processus est itératif et comporte en tout R_{max} tours et le i est l'index du $i$ème tour. Tout ce mécanisme est répété jusqu'à ce que tous les noeuds soient affiliés à une passerelle à R_{max} sauts. Il est très similaire à celui implémenté dans W-GCP et exposé dans le paragraphe 6.1.2. La différence entre l'approche *Recursive* et W-GCP réside essentiellement dans la manière dont les routeurs sont affiliés à une

passerelle. Dans W-GCP, chaque admission est conditionnée par la satisfaction de la condition sur la charge maximale des noeuds. A chaque fois qu'un routeur est attribué à une passerelle v, la charge des noeuds appartenant à l'arbre couvrant enraciné en v est mise à jour pour servir de base dans l'admission des autres routeurs. *Recursive* choisit une passerelle v et vérifie la condition sur la charge des noeuds en la calculant sur tout l'arbre couvrant enraciné en v et au cas où cette condition n'est pas satisfaite, il modifie la matrice de voisinage.
Comme nous l'avons mentionné auparavant, le nombre de passerelles nécessaires avec W-GCP est inférieur à celui avec D-GCP. Il est intéressant à noter que *Recursive* utilise la même critère de choix que D-GCP mais en examinant la figure 6.10 on remarque que la différence entre l'algorithme génétique et *Recursive* est moins importante que celle entre l'algorithme génétique et W-GCP. Ceci peut être expliqué par le fait que *Recursive* utilise une astuce pour contourner la condition sur la capacité en modifiant la matrice de voisinage. Cette différence pourrait aussi découler du schéma de répartition particulier adopté dans *Recursive*, mais pour confirmer ces soupçons nous aurions besoin de simuler cet algorithme avec d'autres scénarios de déploiement. Finalement, l'algorithme génétique sélectionne moins de passerelles que *Recursive* quelque soit la capacité des routeurs.

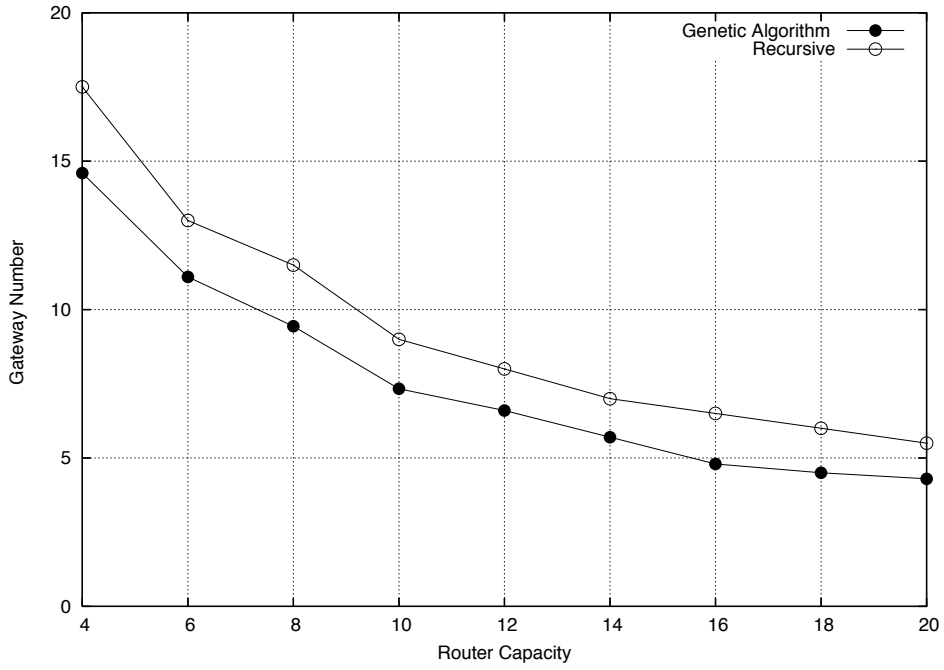


FIG. 6.10 – Nombre de passerelles nécessaires par l'algorithme génétique et Recursive en fonction de la capacité des routeurs avec $R_{max} = 6$.

La question de la représentativité des résultats se pose aussi dans le cas de l'algorithme génétique. De la même manière que dans les chapitres 4 et 5, nous avons examiné les intervalles de confiance à 95%. La figure 6.11 montre que les intervalles de confiance sont suffisamment petits pour valider les résultats de simulation.

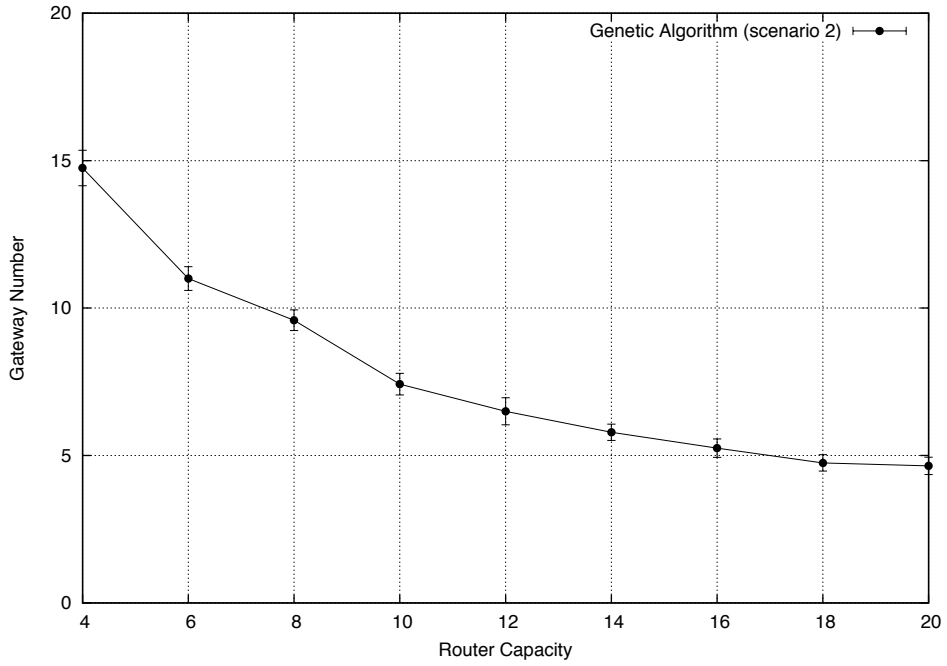


FIG. 6.11 – Validation des résultats pour l'algorithme génétique : intervalles de confiance à 95%.

Nous avons comparé les performances de l'algorithme génétique avec deux approches très similaires qui reposent sur une sélection itérative des passerelles et qui sont basées sur le concept de l'arbre couvrant enraciné. L'algorithme réalise dans les deux cas de meilleures performances en termes de nombre de passerelles nécessaires.

Le concept développé dans les algorithmes *MinHopCount* et *MinContention* est très intéressant étant donné qu'il considère les interférences entre les différentes passerelles installées. Nous proposons comme continuation à ce travail, de comparer leurs performances à celles de l'algorithme génétique.

6.5 Conclusion

Le premier scénario étudié dans le cadre de cette thèse consiste au déploiement d'un réseau ad hoc mobile pour les opérations de lutte contre les feux de forêt. A cause de la mobilité du noeuds, le réseau peut être partitionné et les communications sont donc interrompues entre les différents groupes formés. Le satellite a été proposé pour rétablir la connexité dans le réseau. Nous nous sommes particulièrement intéressés au choix des noeuds qui vont avoir accès au segment satellite et fournir le service de communications dans les partitions. Pour résoudre ce problème, nous avons proposé d'utiliser une technique appelée *clustering*, où les noeuds coopèrent pour élire un noeud particulier dans chaque partition en fonction de plusieurs critères tels que le degré et la durée de vie des liens. Nous avons validé par simulation l'adéquation de cette technique au scénario étudié.

Dans un deuxième temps, nous avons proposé de considérer un autre scénario où les noeuds sont quasi-statiques. Ils forment ainsi un réseau ad hoc *mesh*. Nous supposons que le service est déployé sur plusieurs sites éloignés et le satellite est utilisé pour les interconnecter. De la même manière que dans le premier scénario, nous nous intéressons au choix des noeuds qui vont avoir accès au satellite et fournir le service de communications dans chaque site. Pour résoudre ce problème dans un réseau *mesh*, nous proposons d'appliquer un algorithme génétique.

Ces choix techniques différents soulève deux questions : pourquoi ne pas appliquer l'algorithme génétique pour résoudre le problème du choix des noeuds dans un réseau MANET et pourquoi ne pas utiliser la technique de *clustering* pour résoudre ce même problème dans un réseau *mesh*. Pour répondre à ces deux questions, il faut considérer les propriétés de chaque scénario. La mobilité des noeuds dans un réseau MANET engendre un changement fréquent de la topologie, d'où la nécessité de mécanismes permettant de suivre la dynamique du réseau. En outre, aucune entité centrale ne doit intervenir pour effectuer le choix des noeuds. Or, l'algorithme génétique ne permet pas de s'adapter aux changements topologiques et suppose de connaître la position de tous les noeuds. Il est donc nécessairement exécuté par un noeud central. Si ces propriétés sont considérés comme des inconvénients pour un réseau MANET, elles ne perturbent pas la résolution du problème dans un réseau *mesh* puisque les noeuds sont quasi-statiques.

Pour la deuxième question, si la technique de *clustering* permet de limiter le nombre de sauts maximal entre un noeud et sa passerelle, elle ne permet pas en revanche de garantir une limite sur la charge des noeuds. Or les routeurs *mesh* sont considérés comme une dorsale et fournissent le service de communications à un ensemble de clients *mesh* qui leurs sont attachés. Leur déploiement nécessite donc de garantir une qualité de service

minimale. Pour cela, il est indispensable de connaître la topologie complète réseau. La procédure de choix est effectuée par un noeud central, lors de la phase de planification et de dimensionnement. Quant à la structure créée par la technique de *clustering*, elle émerge à partir d'interactions locales uniquement et fait participer tous les noeuds du réseau. Nous rappelons que le *clustering* a été proposé pour résoudre les problèmes de passage à l'échelle dans les réseaux MANETs et limiter les interactions entre les noeuds dans les réseaux étendus. Il est donc incohérent de supposer que les noeuds ont eu vision globale de tout le réseau.

Pour conclure, la différence des propriétés et des contraintes de chaque scénario (MANET et *mesh*) nous a emmené à considérer deux approches de résolution différentes.

Chapitre 7

Conclusion de la thèse

Dans cette thèse, nous avons traité le problème de restauration de la connexité dans un réseau ad hoc déployé dans des situations d'urgence. Dans une zone dépourvue de tout moyen de communication, le satellite a été proposé pour assurer l'interconnexion entre les différentes parties du réseau. Les noeuds peuvent être mobiles et former un réseau MANET ou quasi-statiques et former un réseau *mesh*. Dans les deux cas, le problème consiste à optimiser le choix des noeuds qui vont avoir accès au satellite.

Pour les réseaux ad hoc mobiles (MANET), nous avons étudié deux concepts différents qui permettent de structurer les réseaux ad hoc mobiles. Le premier concept s'appuie sur la recherche d'un ensemble dominant connecté pour former la dorsale du réseau. La condition sur la connexité de la dorsale introduit de la complexité, de la surcharge en signalisation et de la redondance. La relaxation de cette hypothèse nous conduit au deuxième concept qui repose sur la recherche d'un ensemble dominant indépendant. Ce dernier concept est communément connu sous le nom de *clustering*. Cette technique a été utilisée dans les réseaux ad hoc depuis leur apparition pour démontrer leur capacité d'adaptation aux changements topologiques et de passage à l'échelle. Le réseau est divisé en plusieurs groupes virtuels appelés *clusters* et dans chaque *cluster*, un noeud particulier appelé *clusterhead* coordonne les activités à l'intérieur de son *cluster*.

Le *clustering* à un seul saut se focalise sur les propriétés intrinsèques du réseau. Mais son inconvénient majeur c'est qu'il crée un grand nombre de *clusters* de taille très réduite. En outre, la structure des *clusters* change dès qu'un noeud se retrouve au delà de la portée de transmission de son *clusterhead*. A cause de la mobilité des noeuds, la topologie du réseau change fréquemment modifiant ainsi la structure des *clusters*. Pour plus de stabilité, le concept de *clustering* est généralisé pour créer des *clusters* où les noeuds sont au maximum à k sauts de leur *clusterhead* ($k \geq 1$). Le regroupement des noeuds permet plus d'évolutivité pour les réseaux MANETs. Nous avons adopté une approche basée sur

le concept Max-Min, mais dans une version plus développée avec KCMBC qui a introduit la maintenance et les propriétés intrinsèques des noeuds (le degré et le temps d'expiration). L'évaluation de KCMBC dans un environnement mobile a posé le problème de la modélisation mathématique des déplacements des noeuds sur la surface de simulation.

Un modèle de mobilité peut être soit générique en s'appliquant à un large spectre de scénarios, soit dédié à un scénario bien particulier. Dans les modèles génériques, on distingue deux catégories : les modèles de mobilité d'entité et les modèles de mobilité de groupe. Avec les modèles de mobilité de groupe, on commence à avoir des modèles qui font apparaître des propriétés particulières et imaginer des scénarios d'applications spécifiques. Mais ils restent tout de même des modèles génériques. Avec les modèles dédiés, on part du scénario et on essaye de définir l'ensemble de ses descriptions mathématiques. Les modèles dédiés à l'urgence médicale et aux premiers secours se sont essentiellement focalisés sur la partage de l'espace des opérations et l'attribution des rôles. Il leur manque le concept de coordination. Or quand on parle de la mise en place de communications d'urgence, on parle de coordination des efforts et de coopération des équipes. Nous avons donc proposé d'utiliser un modèle de mobilité, *FireMobility* qui définit des niveaux hiérarchiques, où les déplacements d'un noeud peuvent affecter la mobilité des noeuds appartenant à des niveaux inférieurs. *FireMobility* a été développé lors d'une précédente thèse dans notre laboratoire.

Nous avons comparé *FireMobility* à un modèle d'entité (*Random Walk*) et un modèle de groupe *Reference Region Mobility Model* (RRGM) et à travers une caractérisation spatio-temporelle, nous avons démontré les différences fondamentales dans le mode fonctionnel de chaque modèle. Les propriétés qui ont été étudiées interviennent dans le dimensionnement du système de télécommunications et impactent aussi le comportement de l'algorithme de *clustering* qui sert à choisir les noeuds qui auront accès au satellite. La comparaison des trois modèles a révélé les similitudes et les différences et a surtout démontré que l'adaptation d'un modèle générique est toujours insuffisante. Il est indispensable de développer des modèles dédiés pour représenter et montrer les propriétés du scénario considéré. La simulation de tout le système formée par KCMBC et *FireMobility* a permis de valider l'architecture proposée et de comprendre la dynamique du réseau. L'analyse de tout le système a été le point de départ pour proposer une nouvelle procédure de maintenance adaptée à un réseau partitionné.

Un important surcoût en signalisation nécessite plus d'énergie pour le traitement des messages et signifie plus de collisions et d'interférences dans un environnement sans fil. Pour remédier à ce problème, nous avons proposé une nouvelle procédure de maintenance *Passive Maintenance* qui permet d'améliorer les performances du réseau grâce à la réduction

du surcoût en signalisation. Cette procédure évite l'envoi périodique des informations sur l'état des *clusterheads*, en exploitant d'autres messages de signalisation envoyés par les noeuds dans KCMBC. Le fonctionnement logique de cette procédure a été validé grâce à un réseau de Petri. Ses performances ont été analysées et comparées à une autre procédure de maintenance largement utilisée dans la littérature, *Periodical Broadcast*. Les résultats de simulation ont montré que *Passive Maintenance* permet de réduire le surcoût en signalisation requis pour maintenir la structure des *clusters* ; ce qui permet de réduire la consommation en énergie et économiser de la bande passante. La qualité du *clustering* est elle inchangée.

Certes le contexte des réseaux MANET a été le coeur de nos travaux de recherche, mais nous avons consacré une partie de notre intérêt aux réseaux *mesh*, où les noeuds sont quasi-statiques. La mobilité des noeuds dans un réseau MANET engendre un changement fréquent de la topologie, d'où la nécessité de mécanismes permettant de suivre la dynamique du réseau. En outre, aucune entité centrale ne doit intervenir pour effectuer le choix des noeuds. En ce qui concerne les réseaux *mesh*, les routeurs sont considérés comme une dorsale et fournissent le service de communication à un ensemble de clients *mesh* qui leurs sont attachés. Leur déploiement nécessite donc de garantir une qualité de service minimale. La procédure de choix est effectuée par un noeud central, lors de la phase de planification et de dimensionnement. Ces différences entre les propriétés de chaque scénario, nous ont conduit à appliquer de différentes techniques de résolution. Pour les réseaux *mesh*, nous avons appliqué l'algorithme génétique dont les performances ont été comparées à celles de deux algorithmes basés sur la théorie des graphes, W-GCP et *Recursive*. Les résultats de simulation ont démontré que l'algorithme génétique permet de réduire le nombre de passerelles nécessaires dans le réseau.

Perspectives

Comme continuation à ce travail, nous proposons d'étudier l'effet du *clustering* sur le routage dans le cas du réseau MANET. En effet, le changement du *clusterhead* signifie la modification des tables de routage. Quand deux *clusterheads* A et B se retrouvent dans la même partition, on suppose que par exemple A garde son statut de *clusterhead* et B devient membre. Les noeuds qui étaient affiliés à B vont rediriger leur trafic vers A, étant donné que A assure l'interconnexion de sa partition avec les autres partitions du réseau. Certaines communications avec l'ancien *clusterhead* B peuvent être déjà en cours lorsque ce changement de *clusterheads* a lieu. Le réseau doit garantir la continuité du service aux utilisateurs : on doit absolument éviter la rupture des communications. Des mécanismes de *handover* qui garantissent la continuité du service sont à étudier. Ce problème existe aussi dans les réseaux cellulaires, où les terminaux mobiles passent d'une station de base à autre. Mais dans ces réseaux, les stations de base restent toujours actives ; c'est-à-dire une station n'arrête pas de jouer son rôle de point d'accès. Dans notre cas, le *clusterhead* qui devient membre doit arrêter sa connexion satellite. L'enjeu consiste alors à gérer la migration des flux de l'ancien *clusterhead* vers le nouveau.

Pour résoudre le problème de sélection des passerelles dans les réseaux *mesh*, nous avons proposé d'utiliser l'algorithme génétique, où les opérateurs génétiques sont appliqués à l'ensemble de la population. Nous proposons d'étudier l'impact de la structuration de la population sur les performances de l'algorithme génétique. Les algorithmes génétiques cellulaires [126] considèrent des populations dotées d'une structure topologique : les individus sont répartis sur une grille et ne sont autorisés à interagir qu'avec leurs voisins. Les algorithmes génétiques coopératifs [127] divisent la population en plusieurs sous-populations où un algorithme génétique est appliqué à chaque sous-population séparément.

Dans le chapitre 6 nous avons présenté un algorithme qui permet de résoudre le *Gateway Placement Problem* tout en prenant en considération la contention dans le réseau. On suppose que x passerelles sont installées dans le réseau. Une configuration donnée ne détermine pas uniquement la longueur des chemins et la charge des liens mais aussi le schéma des interférences. L'idée consiste à prendre en compte le niveau d'interférences et les collisions dans la sélection des passerelles et évaluer son impact sur la capacité totale du réseau. Certaines notions classiques de l'ingénierie tels que le *load balancing* et le routage adaptatif peuvent être mises en oeuvre dans l'affiliation des routeurs à une passerelle. Ce que nous appelons le routage adaptatif est le fait de prendre en considération l'état des liens et la charge des noeuds pour une utilisation équitable des ressources du réseau.

Bibliographie

[1] Monia Hamdi, Laurent Franck, and Xavier Lagrange. Réparation par satellites des réseaux ad hoc mobiles pour les communications d'urgence. In GDR Architectures Systemes et Réseaux du CNRS, editor, *RESCOM 2010*, 2010.

[2] Laurent Franck, Monia Hamdi, and Carlos Giraldo Rodriguez. Topology modeling of emergency communication networks : caveats and pitfalls. In The International Management Society, editor, *The International Emergency Management Society Workshop 2011*, 2011.

[3] Monia Hamdi, Laurent Franck, and Xavier Lagrange. Novel cluster maintenance protocol for efficient satellite integration in manets. In *International Communications Satellite Systems Conference (ICSSC)*. AIAA, November 2011.

[4] Monia Hamdi, Laurent Franck, and Carlos Giraldo Rodriguez. Modeling the topology of emergency networks : an application to forest firefighting. *International Journal of Emergency Management*, 2011.

[5] Monia Hamdi, Laurent Franck, and Xavier Lagrange. Gateway placement in hybrid manet-satellite networks. In *IEEE VTC2012-Fall*, 2012.

[6] J. Jubin and J.D. Tornow. The darpa packet radio network protocols. *Proceedings of the IEEE*, 75(1) :21 – 32, jan. 1987.

[7] I. Akyildiz and X. Wang. A survey on wireless mesh networks. *IEEE Radio Communications*, 43 :S23–S30, September 2005.

[8] A. Joshi et al. Hwmp specification. *IEEE 802.11-06/1778r1*, november 2006.

[9] C Perkins, E Royer, and S Das. Rfc 3561 - ad hoc on-demand distance vector routing (aodv), 2003.

[10] Seoung-Bum Lee, Gahng-Seop Ahn, Xiaowei Zhang, and Andrew T. Campbell. Insignia : An ip-based quality of service framework for mobile ad hoc networks. *Journal of Parallel and Distributed Computing*, 60(4) :374 – 406, 2000.

[11] Claude Chaudet and Isabelle Guérin Lassous. Bruit : Bandwidth reservation under interferences influence. In *Proc. of the European Wireless (EW02*, pages 466–472, 2002.

[12] T Clausen and P Jacquet. Rfc 3626 - optimized link state routing protocol (olsr), 2003.

[13] R. Ogier, F. Templin, and M. Lewis. Rfc 3684 - topology dissemination based on reverse-path forwarding (tbrpf), 2004.

[14] D Johnson, Y Hu, and D Maltz. Rfc 4728 - the dynamic source routing protocol (dsr) for mobile ad hoc networks for ipv4, 2007.

[15] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. a high-throughput path metric for multi-hop wireless routing. *Wireless Networks*, 11 :419–434, 2005. 10.1007/s11276-005-1766-z.

[16] Richard Draves, Jitendra Padhye, and Brian Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proceedings of the 10th annual international conference on Mobile computing and networking*, MobiCom '04, pages 114–128, New York, NY, USA, 2004. ACM.

[17] J. Wang Y. Yang and R. Kravets. Interference-aware load balancing for multihop wireless networks. *Technical Report UIUCDCS-R-2005-2526*, 2005.

[18] Y. Shinoda L. T. Nguyen, R. Beuran. A load-aware routing metric for wireless mesh networks. *IEEE Symposium on Computers and Communications. ISCC 2008*, pages 429 –435, july 2008.

[19] Yun Wang, Kai Li, and Qiang Xu. A distributed topology control algorithm for k-connected dominating set in wireless sensor networks. In *Pervasive Computing and Applications, 2007. ICPCA 2007. 2nd International Conference on*, pages 642 –646, july 2007.

[20] P. Bahl L .Li, J. Halpern, Y. Wang, and R. Wattenhofer. A cone-based distributed topology-control algorithm for wireless multi-hop networks. *IEEE/ACM Transactions on Networking*, 13 :147–159, February 2005.

[21] R. Wattenhofer and A. Zollinger. Xtc : A practical topology control algorithm for ad-hoc networks. *18th International Parallel and Distributed Processing Symposium*, April 2004.

[22] Xiuzhen Cheng, Xiao Huang, Deying Li, Weili Wu, and Ding-Zhu Du. A polynomial-time approximation scheme for the minimum-connected dominating set in ad hoc wireless networks. *Networks*, 42(4) :202–208, 2003.

[23] B. Das and V. Bharghavan. Routing in ad-hoc networks using minimum connected dominating sets. In *Communications, 1997. ICC 97 Montreal, 'Towards the Knowledge Millennium'. 1997 IEEE International Conference on*, volume 1, pages 376 –380 vol.1, jun 1997.

[24] Yao-Pin Tsai, Tzu-Ling Hsu, Ru-Sheng Liu, and Ying-Kwei Ho. A backbone routing protocol based on the connected dominating set in ad hoc networks. In *Computer Science and Information Engineering, 2009 WRI World Congress on*, volume 1, pages 14 –18, 31 2009-april 2 2009.

[25] P. J. Macdonald, E. Almaas, and A.-L. Barabasi. Minimum spanning trees of weighted scale-free networks. *EPL (Europhysics Letters)*, 72(2) :308, 2005.

[26] Xiang-Yang Li, Yu Wang, and Wen-Zhan Song. Applications of k-local mst for topology control and broadcasting in wireless ad hoc networks. *Parallel and Distributed Systems, IEEE Transactions on*, 15(12) :1057 – 1069, dec. 2004.

[27] H. Frey, F. Ingelrest, and D. Simplot-Ryl. Localized minimum spanning tree based multicast routing with energy-efficient guaranteed delivery in ad hoc and sensor networks. In *World of Wireless, Mobile and Multimedia Networks, 2008. WoWMoM 2008. 2008 International Symposium on a*, pages 1 –8, june 2008.

[28] Laurent Thomasson, Greet Verelst, Sophie Deprey, Philippe Boutry, Matteo Berioli, and Nicolas Courville. Hybrid satellite-terrestrial based solutions for rapid deployement of wireless telecommunication networks in emergency situations. *15th Annual Conference of the International Emergency Management Society (TIEMS)*, 2008.

[29] CHORIST. Publishable final activity report - sp0.r7. Technical report, Information Society Technology, 2009.

[30] Jaewon Kang, Yanyong Zhang, and B. Nath. Accurate and energy-efficient congestion level measurement in ad hoc networks. In *Wireless Communications and Networking Conference, 2005 IEEE*, volume 4, pages 2258 – 2263 Vol. 4, march 2005.

[31] L. Angrisani, A. Napolitano, and M. Vadursi. Modeling and measuring link capacity in communication networks. *Instrumentation and Measurement, IEEE Transactions on*, 59(5) :1065 –1072, may 2010.

[32] I. C. Atalay, Y. Sarikaya, O. Gurbuz, and O. Ercetin. Accurate non-intrusive residual bandwidth estimation in wmns. *Sensor, Mesh and Ad Hoc Communications and Networks Workshops, 2008. SECON Workshops '08. 5th IEEE Annual Communications Society Conference on*, pages 1–6, 16-20 June 2008.

[33] H. Ritter, R. Winter, and J. Schiller. A partition detection system for mobile ad-hoc networks. In *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, pages 489 – 497, oct. 2004.

[34] Abdelouahid Derhab, Nadjib Badache, and Abdelmadjid Bouabdallah. A partition prediction algorithm for service replication in mobile ad hoc networks. *Wireless on Demand Network Systems and Service, International Conference on*, 0 :236–245, 2005.

[35] Yan Zhang, Chor Ping Low, Jim Mee Ng, and Ting Wang. An efficient group partition prediction scheme for manets. *IEEE Wireless Communications and Networking Conference*, pages 1–6, 2009.

[36] Bing He, Bin Xie, and Dharma P. Agrawal. Internet gateway deployment optimization in a multi-channel multi-radio wireless mesh network. *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 2259–2264, march 2008.

[37] J. Robinson, M. Uysal, R. Swaminathan, and E. Knightly. Adding capacity points to a wireless mesh network using local search. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 1247 –1255, april 2008.

[38] B. Aoun, R. Boutaba, Youssef Iraqi, and G. Kenward. Gateway placement optimization in wireless mesh networks with qos constraints. *Selected Areas in Communications, IEEE Journal on*, 24(11) :2127 –2136, nov. 2006.

[39] Teresa W. Haynes, S. T. Hedetniemi, and Peter J. Slater. *Fundamentals of Domination in Graphs*. A series of Monographs and text books, 1998.

[40] Jie Wu. Extended dominating-set-based routing in ad hoc wireless networks with unidirectional links. *Parallel and Distributed Systems, IEEE Transactions on*, 13(9) :866 – 881, sep 2002.

[41] Jie Wu and Hailan Li. On calculating connected dominating set for efficient routing in ad hoc wireless networks. In *Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications*, DIALM '99, pages 7–14, New York, NY, USA, 1999. ACM.

[42] Hong-Yen Yang, Chia-Hung Lin, and Ming-Jer Tsai. Distributed algorithm for efficient construction and maintenance of connected k-hop dominating sets in mobile ad hoc networks. *Mobile Computing, IEEE Transactions on*, 7(4) :444 –457, april 2008.

[43] K. Alzoubi, P.-J. Wan, and O. Frieder. New distributed algorithm for connected dominating set in wireless ad hoc networks. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 9 - Volume 9*, HICSS '02, pages 297–, Washington, DC, USA, 2002. IEEE Computer Society.

[44] Zeng Yuanyuan, Xiaohua Jia, and He Yanxiang. Energy efficient distributed connected dominating sets construction in wireless sensor networks. In *Proceedings of the 2006 international conference on Wireless communications and mobile computing*, IWCMC '06, pages 797–802, New York, NY, USA, 2006. ACM.

[45] Ning Li, J.C. Hou, and L. Sha. Design and analysis of an mst-based topology control algorithm. *Wireless Communications, IEEE Transactions on*, 4(3) :1195 – 1206, may 2005.

[46] P. Teymoori and N. Yazdani. Local reconstruction of virtual backbone to support mobility in wireless ad hoc networks. In *Telecommunications, 2008. IST 2008. International Symposium on*, pages 382 –387, aug. 2008.

[47] D. Baker and A. Ephremides. The architectural organization of a mobile radio network via a distributed algorithm. *Communications, IEEE Transactions on*, 29(11) :1694 – 1701, nov 1981.

[48] A. Ephremides. Design concepts for a mobile-user radio network. *Computers and Electrical Engineering*, 10(3) :127 – 135, 1983.

[49] A. Ephremides, J.E. Wieselthier, and D.J. Baker. A design concept for reliable mobile radio networks with frequency hopping signaling. *Proceedings of the IEEE*, 75(1) :56 – 73, jan. 1987.

[50] V. S. Anitha and M. P. Sebastian. Scenario-based diameter-bounded algorithm for cluster creation and management in mobile ad hoc networks. In *Proceedings of the 2009 13th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications*, DS-RT '09, pages 97–104, Washington, DC, USA, 2009. IEEE Computer Society.

[51] J.Y. Yu and P.H.J. Chong. A survey of clustering schemes for mobile ad hoc networks. *Communications Surveys Tutorials, IEEE*, 7(1) :32 –48, qtr. 2005.

[52] C.R. Lin and M. Gerla. Adaptive clustering for mobile wireless networks. *Selected Areas in Communications, IEEE Journal on*, 15(7) :1265 –1275, sep 1997.

[53] C C Chiang, H K Wu, W Liu, and M Gerla. *Routing in clustered multihop, mobile wireless networks with fading channel*, volume 97, pages 1–15. Citeseer, 1997.

[54] J.Y. Yu and P.H.J. Chong. 3hbac (3-hop between adjacent clusterheads) : a novel non-overlapping clustering algorithm for mobile ad hoc networks. In *Communications, Computers and signal Processing, 2003. PACRIM. 2003 IEEE Pacific Rim Conference on*, volume 1, pages 318 – 321 vol.1, aug. 2003.

[55] S. Basagni. Distributed clustering for ad hoc networks. In *Parallel Architectures, Algorithms, and Networks, 1999. (I-SPAN '99) Proceedings. Fourth International-Symposium on*, pages 310 –315, 1999.

[56] Mainak Chatterjee, Sajal K. Das, and Damla Turgut. A weight based distributed clustering algorithm for mobile ad hoc networks. In *Proceedings of the 7th International Conference on High Performance Computing*, HiPC '00, pages 511–521, London, UK, 2000. Springer-Verlag.

[57] M.R. Brust, A. Andronache, and S. Rothkugel. Waca : A hierarchical weighted clustering algorithm optimized for mobile hybrid networks. In *Wireless and Mobile Communications, 2007. ICWMC '07. Third International Conference on*, page 23, march 2007.

[58] Geng Chen, F.G. Nocetti, J.S. Gonzalez, and I. Stojmenovic. Connectivity based k-hop clustering in wireless networks. In *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on*, pages 2450 – 2459, jan. 2002.

[59] F.D. Tolba, D. Magoni, and P. Lorenz. Connectivity, energy and mobility driven clustering algorithm for mobile ad hoc networks. In *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, pages 2786 –2790, nov. 2007.

[60] Inc Cisco Systems. Cisco aironet 802.11a/b/g wireless cardbus adapter. Cisco public information, Cisco Systems, Inc, 2007.

[61] M.R. Brust, A. Andronache, S. Rothkugel, and Z. Benenson. Topology-based clusterhead candidate selection in wireless ad-hoc and sensor networks. In *Communication Systems Software and Middleware, 2007. COMSWARE 2007. 2nd International Conference on*, pages 1 –8, jan. 2007.

[62] Yi Wang, Hairong Chen, Xinyu Yang, and Deyun Zhang. Wachm : Weight based adaptive clustering for large scale heterogeneous manet. In *Communications and Information Technologies, 2007. ISCIT '07. International Symposium on*, pages 936 –941, oct. 2007.

[63] Supeng Leng, Yan Zhang, Hsiao-Hwa Chen, Liren Zhang, and Ke Liu. A novel k-hop compound metric based clustering scheme for ad hoc wireless networks. *Wireless Communications, IEEE Transactions on*, 8(1) :367 –375, jan. 2009.

[64] P. Basu, N. Khan, and T.D.C. Little. A mobility based metric for clustering in mobile ad hoc networks. In *Distributed Computing Systems Workshop, 2001 International Conference on*, pages 413 –418, apr 2001.

[65] Yan Zhang, Jim Mee Ng, and Chor Ping Low. A distributed group mobility adaptive clustering algorithm for mobile ad hoc networks. *Comput. Commun.*, 32 :189–202, January 2009.

[66] Zouhair El-Bazzal, M. Kadoch, B.L. Agba, F. Gagnon, and M. Bennani. A flexible weight based clustering algorithm in mobile ad hoc networks. In *Systems and Networks Communications, 2006. ICSNC '06. International Conference on*, page 50, oct. 2006.

[67] S.K. Dhurandher and G.V. Singh. Stable clustering with efficient routing in wireless ad hoc networks. In *Communication Systems Software and Middleware, 2007. COMSWARE 2007. 2nd International Conference on*, pages 1 –12, jan. 2007.

[68] Marco Aurélio Spohn and J. J. Garcia-Luna-Aceves. Bounded-distance multi-clusterhead formation in wireless ad hoc networks. *Ad Hoc Netw.*, 5 :504–530, May 2007.

[69] Giacomo Angione, Paolo Bellavista, Antonio Corradi, and Eugenio Magistretti. A k-hop clustering protocol for dense mobile ad-hoc networks. *Distributed Computing Systems Workshops, International Conference on*, 0 :10, 2006.

[70] Khac Tiep Mai, Dongkun Shin, and Hyunseung Choo. Connectivity-based clustering with stretching technique in manets. In *Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication*, ICUIMC '09, pages 200–206, New York, NY, USA, 2009. ACM.

[71] A.D. Amis, R. Prakash, T.H.P. Vuong, and D.T. Huynh. Max-min d-cluster formation in wireless ad hoc networks. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 1, pages 32 –41 vol.1, 2000.

[72] Marius Portmann and Asad Amir Pirzada. Wireless mesh networks for public safety and crisis management applications. *IEEE Internet Computing*, 12 :18–25, 2008.

[73] S. Carson and J. Macker. Rfc 2501 - mobile ad hoc networking (manet) : Routing protocol performance issues and evaluation considerations, 1999.

[74] Mark A. Weissberger. An initial critical summary of models for predicting the attenuation of radio waves by trees. *Department of Defense, Electromagnetic Compatibility Analysis Center Annapolis, Maryland 21402*, 1982.

[75] J.D. Parsons and A.M.D. Turkmani. Performance evaluation of m-branch selection diversity with continuous phase modulation systems in rayleigh and log-normal fading. In *Methods of Combating Multipaths, IEE Colloquium on*, pages 11/1 –11/6, jan 1990.

[76] Per Johansson, Tony Larsson, Nicklas Hedman, Bartosz Mielczarek, and Mikael Degermark. Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, MobiCom '99, pages 195–206, New York, NY, USA, 1999. ACM.

[77] F. Bai, Narayanan Sadagopan, and A. Helmy. Important : a framework to systematically analyze the impact of mobility on performance of routing protocols for adhoc networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 2, pages 825 – 835 vol.2, march-3 april 2003.

[78] Einstein A. Über die von der molekularkinetischen theorie der wärme geforderte bewegung von in ruhenden flüssigkeiten suspendierten teilchen. *Annalen der Physik 322*, 8 :549–560, 1905.

[79] B. Liang and Z.J. Haas. Predictive distance-based mobility management for pcs networks. In *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE*

Computer and Communications Societies. Proceedings. IEEE, volume 3, pages 1377 –1384 vol.3, mar 1999.

[80] C. A. V. Campos and L. F. M. de Moraes. A markovian model representation of individual mobility scenarios in ad hoc networks and its evaluation. *EURASIP J. Wirel. Commun. Netw.*, 2007 :35–35, January 2007.

[81] Stuart Kurkowski, Tracy Camp, and William Navidi. Two standards for rigorous manet routing protocol evaluation. In *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, pages 256 –266, oct. 2006.

[82] J. Yoon, M. Liu, and B. Noble. Random waypoint considered harmful. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 2, pages 1312 – 1321 vol.2, march-3 april 2003.

[83] Tracy Camp, Jeff Boleng, and Vanessa Davies. A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing (WCMC) : Special issue on mobile ad hoc networking : Resarch, trands and applications*, 2 :483–502, 2002.

[84] Christian Bettstetter, Hannes Hartenstein, and Xavier Pérez-Costa. Stochastic properties of the random waypoint mobility model. *Wirel. Netw.*, 10 :555–567, September 2004.

[85] Christian Bettstetter. Mobility modeling in wireless networks : categorization, smooth movement, and border effects. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5 :55–66, July 2001.

[86] Z.J. Haas. A new routing protocol for the reconfigurable wireless networks. In *Universal Personal Communications Record, 1997. Conference Record., 1997 IEEE 6th International Conference on*, volume 2, pages 562 –566 vol.2, oct 1997.

[87] M. Sanchez and P. Manzoni. A java-based ad hoc networks simulator. In *Proceedings of the SCS Western Multiconference Web-based Simulation Track*, 1999.

[88] J.M. Ng and Y. Zhang. A mobility model with group partitioning for wireless ad hoc networks. In *Information Technology and Applications, 2005. ICITA 2005. Third International Conference on*, volume 2, pages 289 –294, july 2005.

[89] Xiaoyan Hong, Mario Gerla, Guangyu Pei, and Ching chuan Chiang. A group mobility model for ad hoc wireless networks. In *Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, pages 53–60, 1999.

[90] Jing Tian, Joerg Haehner, Christian Becker, Illya Stepanov, and Kurt Rothermel. Graph-based mobility model for mobile ad hoc network simulation. In *Proceedings of*

the 35th Annual Simulation Symposium, pages 337–, Washington, DC, USA, 2002. IEEE Computer Society.

[91] Amit Jardosh, Elizabeth M. Belding-Royer, Kevin C. Almeroth, and Subhash Suri. Towards realistic mobility models for mobile ad hoc networks. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, MobiCom '03, pages 217–229, New York, NY, USA, 2003. ACM.

[92] Mark de Berg, Otfried Cheong, Mark van Kreveld, and Mark Overmars. *Computational Geometry Algorithms and Applications*. Springer-Verlag, 2008.

[93] Nils Aschenbruck, Elmar Gerhards-Padilla, Michael Gerharz, Matthias Frank, and Peter Martini. Modelling mobility in disaster area scenarios. In *Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, MSWiM '07, pages 4–12, New York, NY, USA, 2007. ACM.

[94] Ying Huang, Wenbo He, K. Nahrstedt, and W.C. Lee. Corps : Event-driven mobility model for first responders in incident scene. In *Military Communications Conference, 2008. MILCOM 2008. IEEE*, pages 1 –7, nov. 2008.

[95] Giraldo Rodriguez Carlos. *MANET routing assisted by satellites*. PhD thesis, Télécom Bretagne/Institut Télécom, 2011.

[96] Direction de la défense et de la sécurité civiles Sous direction des sapeurs-pompiers et des acteurs du secours. Guide national de référence- techniques professionnelles -manoeuvres feux de forêt. Technical report, Ministère de l'intérieur, 2008.

[97] C. Lampin-Cabaret, M. Jappiot, N. Alibert, R. Manlay, and R. Guillande. Prototype d'une échelle d'intensité pour le phénomène « incendie de forêts». *Ingénieries n°31*, pages 49–56, 2002.

[98] E. Sakhaee and A. Jamalipour. A new stable clustering scheme for pseudo-linear highly mobile ad hoc networks. In *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, pages 1169 –1173, nov. 2007.

[99] Yong Li and Ping Wang. A load balance k-hop clustering algorithm for ad hoc networks. In *Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09. 5th International Conference on*, pages 1 –4, sept. 2009.

[100] G. Venkataraman, S. Emmanuel, and S. Thambipillai. A novel distributed cluster maintenance technique for high mobility ad-hoc networks. In *Wireless Communication Systems, 2004, 1st International Symposium on*, pages 225 – 229, sept. 2004.

[101] Bo Xing, M. Deshpande, S. Mehrotra, and N. Venkatasubramanian. Gateway designation for timely communications in instant mesh networks. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*, pages 564 –569, 29 2010-april 2 2010.

[102] P. Bellavista and E. Magistretti. How node mobility affects k-hop cluster quality in mobile ad hoc networks : A quantitative evaluation. In *Computers and Communications, 2008. ISCC 2008. IEEE Symposium on*, pages 750 –756, july 2008.

[103] T. Murata. Petri nets : Properties, analysis and applications. *Proceedings of the IEEE*, 77(4) :541 –580, apr 1989.

[104] Imperial College London Department of Computing. The platform independent petri net editor pipe. `http://pipe2.sourceforge.net`.

[105] René David and Hassane Alla. *Discrete, Continuous, and Hybrid Petri Nets*, chapter Chapter 2 : Properties of Petri Nets. Springer, 2010.

[106] Michele Luglio, Cristiano Monti, Cesare Roseti, Antonio Saitto, and Michael Segal. Interworking between manet and satellite systems for emergency applications. *International Journal of Satellite Communications and Networking*, 25(5) :551–558, 2007.

[107] Bing He, Bin Xie, and Dharma P. Agrawal. Optimizing the internet gateway deployment in a wireless mesh network. In *IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems*, pages 1–9, october 2007.

[108] Wenjia Wu, Junzhou Luo, and Ming Yang. Gateway placement optimization for load balancing in wireless mesh networks. *13th International Conference on Computer Supported Cooperative Work in Design*, pages 408–413, 2009.

[109] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristic for k-median and facility location problems. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, STOC '01, pages 21–29, New York, NY, USA, 2001. ACM.

[110] Michael Patriksson. On the applicability and solution of bilevel optimization models in transportation science : A study on the existence, stability and computation of optimal solutions to stochastic mathematical programs with equilibrium constraints. *Transportation Research*, 42(10) :843 – 860, 2008.

[111] Shaw J Bialas W, Karwan M. a parametric complementarity pivot approach for two-level linear programming, technical report , state university of new york at buffalo. *Operations Research Program*, 1980.

[112] Lasdon LS Kolstad CD. Derivative estimation and computational experience with large bilevel mathematical programs. *Journal of Optimization Theory and Applications*, 65 :485 – 499, 1990.

[113] Yo Ishizuka and Eitaro Aiyoshi. Double penalty method for bilevel optimization problems. *Annals of Operations Research*, 34 :73–88, 1992. 10.1007/BF02098173.

[114] Loukas Dimitriou, Theodore Tsekeris, and Antony Stathopoulos. Evolutionary combinatorial programming for discrete road network design with reliability requirements. In Mario Giacobini, editor, *Applications of Evolutionary Computing*, volume 4448 of *Lecture Notes in Computer Science*, pages 678–687. Springer Berlin / Heidelberg, 2007.

[115] H. Schittenhelm. On the integration of an effective assignment algorithm with path and path-flow management in a combined trip distribution and traffic assignment algorithm. *8th PTRC Summer Annual Meeting*, 1990.

[116] John Glen Wardrop. Some theoretical aspects of road traffic research. *Proceedings, Institute of Civil Engineers*, 1 :325–362, 1952.

[117] Leblanc L.J. An algorithm for the discrete network design problem. *Transportation Science*, 9(3) :183–199, 1975.

[118] MAGNANTI T. L. and WONG R. T. Network design and transportation planning : models and algorithms. *Institute for Operations Research and the Management Sciences*, 18(1) :1–55, 1984.

[119] Ziyou Gao, Jianjun Wu, and Huijun Sun. Solution algorithm for the bi-level discrete network design problem. *Transportation Research Part B : Methodological*, 39(6) :479 – 495, 2005.

[120] Hossain Poorzahedy and Omid M. Rouhani. Hybrid meta-heuristic algorithms for solving network design problem. *European Journal of Operational Research*, 182(2) :578 – 596, 2007.

[121] Liang Xu and Ziyou Gao. Bi-objective urban road transportation discrete network design problem under demand and supply uncertainty. In *Automation and Logistics, 2008. ICAL 2008. IEEE International Conference on*, pages 1951 –1955, sept. 2008.

[122] John H. Holland. Outline for a logical theory of adaptive systems. *J. ACM*, 9 :297–314, July 1962.

[123] M. Srinivas and L.M. Patnaik. Genetic algorithms : a survey. *Computer*, 27(6) :17 –26, jun 1994.

[124] D.E Goldberg. Genetic algorithms and walsh functions. part 1 and 2. *Complex Systems*, 3 :129–171, 1989.

[125] David E. Goldberg and Kalyanmoy Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms*, pages 69–93. Morgan Kaufmann, 1991.

[126] Enrique Alba and Bernabe Dorronsoro. *Cellular Genetic Algorithms*. Operations Research/Computer Science Interfaces Series. Springer-Verlag, 2008.

[127] Mitchell Potter and Kenneth De Jong. A cooperative coevolutionary approach to function optimization. In *Parallel Problem Solving from Nature - PPSN III*, volume 866 of *Lecture Notes in Computer Science*, pages 249–257. Springer Berlin / Heidelberg, 1994.